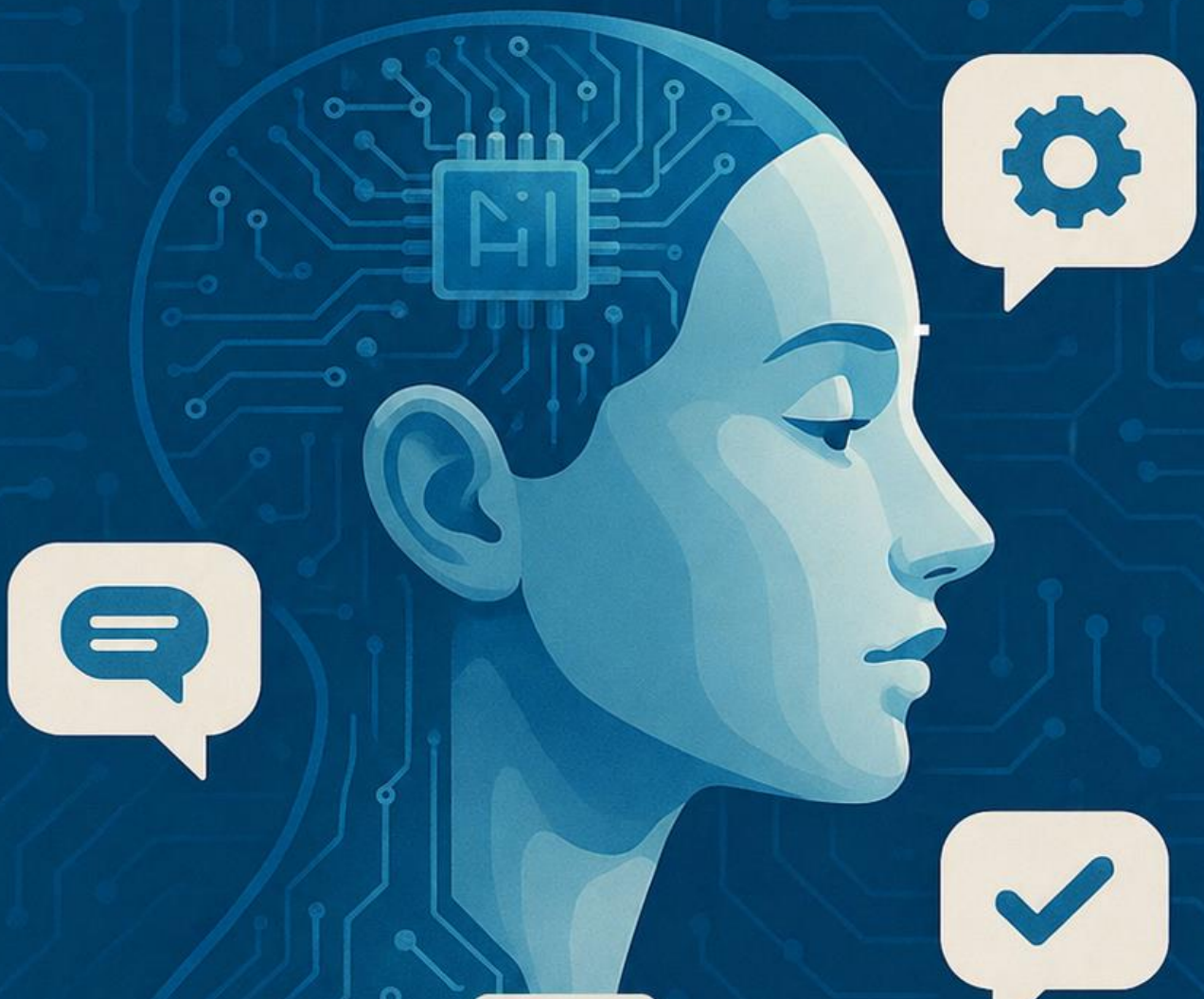
एआई असिस्टेंट

कक्षा 10 के लिए मॉड्यूल



एन सी ई आर टी
NCERT

पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान
(भारत सरकार के शिक्षा मंत्रालय के अधीन रा.शै.अ.प्र.प. की घटक इकाई)
श्यामला हिल्स, भोपाल– 462002, मध्य प्रदेश, भारत
http://www.psscive.ac.in

एआई असिस्टेंट

(Artificial Intelligence Assistant)

(कार्य-भूमिका)

(योग्यता पैक — संदर्भ सं. NIE/SSC/Q1003)

कार्य क्षेत्र — सूचना प्रौद्योगिकी सूचना प्रौद्योगिकी सक्षम सेवाएँ

(आई.टी. – आई.टी.ई.एस.)

कक्षा 10 के लिए मॉड्यूल

© PSSCIVE Draft Study Material Not to be Published

विद्यया ऽमृतमश्नुते

एन सी ई आर टी
NCERT

पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान

(भारत सरकार के शिक्षा मंत्रालय के अधीन रा.शै.अ.प्र.प. की घटक मॉड्यूल)

श्यामला हिल्स, भोपाल— 462002, मध्य प्रदेश, भारत

http//www.psscive.ac.in

© पं.सुं.श. केंद्रीय व्यावसायिक शिक्षा संस्थान, भोपाल 2025

प्रकाशक की पूर्व अनुमति के बिना इस प्रकाशन के किसी भी भाग को किसी भी रूप में या किसी भी माध्यम से, इलेक्ट्रॉनिक, यांत्रिक, फोटोकॉपी, रिकॉर्डिंग या अन्य किसी भी माध्यम से पुनरुत्पादित, पुनर्प्राप्ति प्रणाली में संग्रहीत या प्रेषित नहीं किया जा सकता है।

© PSSCIVE Draft Study Material Not be Published

आमुख

व्यावसायिक शिक्षा एक गतिशील और विकासशील क्षेत्र है और यह सुनिश्चित करना अत्यंत महत्वपूर्ण है कि प्रत्येक विद्यार्थी के पास गुणवत्तापूर्ण शिक्षण सामग्री उपलब्ध हो। पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान (पी.एस.एस.सी.आई.वी.ई.) की व्यापक और समावेशी अध्ययन सामग्री तैयार करने की यात्रा कठिन और समय लेने वाली है जिसके लिए गहन शोध, विशेषज्ञ परामर्श और राष्ट्रीय शैक्षिक अनुसंधान और प्रशिक्षण परिषद् (रा.शै.अ.प्र.प.) द्वारा प्रकाशन की आवश्यकता है। हालाँकि, अंतिम अध्ययन सामग्री की अनुपस्थिति हमारे विद्यार्थियों की शैक्षिक प्रगति में बाधा नहीं बननी चाहिए। इस आवश्यकता को देखते हुए हम प्रारूप अध्ययन सामग्री प्रस्तुत करते हैं, जो एक अनंतिम लेकिन व्यापक मार्गदर्शिका है, जिसे शिक्षण और सीखने के बीच का अंतर दूर करने के लिए डिजाइन किया गया है, जब तक कि अध्ययन सामग्री का आधिकारिक संस्करण रा.शै.अ.प्र.प. द्वारा उपलब्ध नहीं करा दिया जाता। प्रारूप अध्ययन सामग्री शिक्षकों और विद्यार्थियों के लिए अंतरिम अवधि में उपयोग करने के लिए सामग्री का एक संरचित और सुलभ सेट प्रदान करती है। सामग्री को निर्धारित पाठ्यक्रम के साथ संरेखित किया गया है ताकि यह सुनिश्चित किया जा सके कि विद्यार्थी अपने सीखने के उद्देश्यों के साथ सत्य रास्ते पर बने रहें।

मॉड्यूल की विषयवस्तु शिक्षा में निरंतरता बनाए रखने और व्यावसायिक शिक्षा में शिक्षण-अधिगम की गति को बनाए रखने के लिए तैयार की गई है। इसमें पाठ्यक्रम और शैक्षिक मानकों के अनुरूप आवश्यक अवधारणाएँ और कौशल शामिल हैं। हम उन शिक्षाविदों, व्यावसायिक शिक्षकों, विषय विशेषज्ञों, उद्योग विशेषज्ञों, शैक्षणिक सलाहकारों और अन्य सभी लोगो के प्रति आभार व्यक्त करते हैं जिन्होंने इस प्रारूप अध्ययन सामग्री के निर्माण में अपनी विशेषज्ञता और अंतर्दृष्टि प्रदान की।

शिक्षकों को अध्ययन सामग्री के प्रारूप मॉड्यूल को एक मार्गदर्शक के रूप में उपयोग करने और अपने शिक्षण को अतिरिक्त संसाधनों और गतिविधियों से पूरक बनाने के लिए प्रोत्साहन दिया जाता है जो उनके विद्यार्थियों की विशिष्ट शिक्षण शैलियों और आवश्यकताओं को पूरा करते हैं। सहयोग और प्रतिक्रिया महत्वपूर्ण हैं; इसलिए, हम अध्ययन सामग्री की विषय-वस्तु में सुधार के लिए विशेष रूप से शिक्षकों द्वारा, सुझावों का स्वागत करते हैं।

यह सामग्री कॉपीराइट के अधीन है और इसे रा.शै.अ.प्र.प.- पी.एस.एस.सी.आई.वी.ई. की अनुमति के बिना मुद्रित नहीं किया जाना चाहिए।

भोपाल

अगस्त 2025

दीपक पालीवाल

संयुक्त निदेशक

पं.सुं.श. केंद्रीय व्यावसायिक शिक्षा संस्थान (पी.एस.एस.सी.आई.वी.ई.)

राष्ट्रीय शैक्षिक अनुसंधान और प्रशिक्षण परिषद्

© PSSCIVE Draft Study Material Not to be Published

माड्यूल विकास दल

सदस्य

दीपक दि. शुद्धलवार, प्राध्यापक (सी.एस.ई.), अभियांत्रिकी एवं प्रौद्योगिकी विभाग, पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान, रा.शै.अ.प्र.प., श्यामला हिल्स, भोपाल, मध्यप्रदेश।

प्रकाश खनाले, वरिष्ठ समन्वयक, स्कूल ऑफ कंप्यूटर साइंस, वाईसीएमओयू, नासिक, पूर्व प्राध्यापक और विभागाध्यक्ष, कंप्यूटर विज्ञान विभाग, डीएसएम कॉलेज, परभणी, महाराष्ट्र

अक्षय आर्या, सहायक प्रोफेसर, आर्टिफिशियल इंटेलिजेंस एवं डेटा साइंस विभाग, आर्य कॉलेज ऑफ इंजीनियरिंग, मुख्य परिसर, एसपी 40, कुकस, जयपुर (राजस्थान)

कुलदीप मालविया, (एस.आर.ए), इंजीनियरिंग एवं प्रौद्योगिकी विभाग, पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान, रा.शै.अ.प्र.प., श्यामला हिल्स, भोपाल, मध्यप्रदेश।

अनुवाद, संपादन एवं समीक्षा

मुकेश कुमार चौरासे, सहायक प्राध्यापक, माखनलाल चतुर्वेदी राष्ट्रीय पत्रकारिता एवं संचार विश्वविद्यालय, भोपाल, मध्य प्रदेश

अरविंद सिंह राजपूत, मुकेश कुमार चौरासे, सह प्राध्यापक (सी.एस.ई.), माखनलाल चतुर्वेदी राष्ट्रीय पत्रकारिता एवं संचार विश्वविद्यालय, भोपाल, मध्य प्रदेश

ले-आउट और कंपोजिंग

राजेश कहार, डीटीपी ऑपरेटर, पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान, रा.शै.अ.प्र.प., श्यामला हिल्स, भोपाल, मध्यप्रदेश।

अनुवाद कार्यक्रम समन्वयक

रजनीश, सहायक पुस्तकालयाध्यक्ष, पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान, रा.शै.अ.प्र.प., श्यामला हिल्स, भोपाल, मध्यप्रदेश।

राज्य समन्वयक

विपिन कुमार जैन, सह प्राध्यापक एवं विभागाध्यक्ष, मानविकी, विज्ञान, शिक्षा और अनुसंधान, पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान, रा.शै.अ.प्र.प., श्यामला हिल्स, भोपाल, मध्यप्रदेश।

कार्यक्रम समन्वयक (हिन्दी अनुवाद)

दीपक दि. शुद्धलवार, प्राध्यापक (सी.एस.ई.), अभियांत्रिकी एवं प्रौद्योगिकी विभाग, पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान, रा.शै.अ.प्र.प., श्यामला हिल्स, भोपाल, मध्यप्रदेश।

© PSSCIVE Draft Study Material Not be Published

विषय सूची

क्र. सं.	शीर्षक	पृष्ठ सं.
1.	मॉड्यूल 1— पायथन प्रोग्रामिंग	1
	सत्र 1— कंट्रोल स्ट्रक्चर	2
	अपनी प्रगति जांचें	30
	सत्र 2— पायथन में फंक्शन	33
	अपनी प्रगति जांचें	67
2.	मॉड्यूल 2— डेटा साइंस	74
	सत्र 1— NumPy का परिचय	75
	अपनी प्रगति जांचें	91
	सत्र 2— NumPy का उपयोग करके एरे हेरफेर	93
	अपनी प्रगति जांचें	103
	सत्र 3— NumPy का उपयोग करके एरे संगणना	105
	अपनी प्रगति जांचें	114
3.	मॉड्यूल 3— डेटा विश्लेषण	117
	सत्र 1— पांडस का परिचय	118
	अपनी प्रगति जांचें	122
	सत्र 2— पांडस सीरीज	124
	अपनी प्रगति जांचें	132
	सत्र 3— पांडस डेटाफ्रेम	135
	अपनी प्रगति जांचें	141
	सत्र 4— मैटप्लोटलिब का उपयोग करके डेटा विज़ुअलाइज़ेशन	144
	अपनी प्रगति जांचें	148
4.	मॉड्यूल 4— न्यूरल नेटवर्क	151
	सत्र 1— न्यूरल नेटवर्क का परिचय	152
	अपनी प्रगति जांचें	158
	सत्र 2— न्यूरॉन के अंदर: सक्रियण फ़ंक्शन	160
	अपनी प्रगति जांचें	165
	सत्र 3— क्रियाशील न्यूरल नेटवर्क: वास्तविक दुनिया के अनुप्रयोग	166
	अपनी प्रगति जांचें	169
	सत्र 4— पायथन लाइब्रेरी के साथ न्यूरल नेटवर्क का निर्माण	170
	अपनी प्रगति जांचें	175

© PSSCIVE Draft Study Material Not be Published

क्र. सं.	शीर्षक	पृष्ठ सं.
	सत्र 5— डेटा तैयार करना और अपने मॉडल को प्रशिक्षित करना	176
	अपनी प्रगति जांचें	181
	सत्र 6— वर्गीकरण मॉडल का निर्माण	182
	अपनी प्रगति जांचें	186
5.	मॉड्यूल 5— ए आई प्रोजेक्ट	188
	सत्र 1— ए आई प्रोजेक्ट साइकिल	189
	अपनी प्रगति जांचें	194
	सत्र 2— प्रोजेक्ट वर्क	197
	अपनी प्रगति जांचें	203
	सत्र 3— सैम्पल प्रोजेक्ट	205
6.	शब्दावली	211
7.	उत्तर कुंजी	213

© PSSCIVE Draft Study Material Not be Published

मॉड्यूल 1. पायथन प्रोग्रामिंग

(Python Programming)

मॉड्यूल का संक्षिप्त परिचय (Module Overview)

हमारे दैनिक जीवन में कुछ ऐसी परिस्थितियाँ होती हैं, जहाँ किसी कार्य को पूरा करने के लिए हमें निश्चित क्रम में कुछ चरणों का पालन करना पड़ता है, जबकि कुछ अन्य परिस्थितियों में कार्य को पूरा करने के लिए हमारे पास चरणों के चुनाव की स्वतंत्रता होती है। उदाहरण के लिए, हवाई यात्रा में प्रत्येक यात्री को विमान में सवार होने के लिए निम्नलिखित चरणों का पालन करना होता है।

1. हवाई अड्डे के मुख्य द्वार पर अपना टिकट और पहचान पत्र दिखाएं।
2. अपने सामान की जाँच (स्कैनिंग) कराएं।
3. अपना बोर्डिंग पास प्राप्त करें।
4. सुरक्षा जाँच से गुजरें।
5. अंत में, विमान में सवार हों।

यात्रियों के पास विमान में सवार होने के लिए ऊपर दिए गए चरणों के क्रम को बदलने का विकल्प नहीं होता है। दूसरी ओर, यात्री अपना बोर्डिंग पास एयरलाइन के चेक-इन काउंटर से ले सकते हैं या पास में लगे इलेक्ट्रॉनिक कियोस्क से भी प्राप्त कर सकते हैं, जैसा कि चित्र 1.1 में दिखाया गया है।

इसी प्रकार, प्रोग्रामिंग में सामान्यतः कथनों (स्टेटमेंट्स) का निष्पादन शुरू से अंत तक उसी क्रम में होता है, जिस क्रम में वे लिखे गए होते हैं। लेकिन प्रोग्रामिंग में भी कुछ ऐसी स्थितियाँ आती हैं, जहाँ कथनों के सामान्य निष्पादन क्रम को बदलना आवश्यक हो जाता है। इस इकाई में हम निर्णय और बचाव के साथ कंट्रोल फ्लो पर चर्चा करेंगे।

चित्र 1.1: विमान में चढ़ने के चरण

अधिगम के परिणाम (Learning Outcomes)

इस मॉड्यूल को पूरा करने के बाद, आप सक्षम होंगे:

• प्रोग्रामिंग में नियंत्रण प्रवाह (कंट्रोल फ्लो) की अवधारणा को समझेंगे।

• प्रोग्राम किस प्रकार क्रमिक (सीक्वेंशियल) रूप से निष्पादित होंगे, इसे समझेंगे।

Study Material Not be Published

• उन परिस्थितियों की पहचान करेंगे जहाँ निर्णय-निर्माण और पुनरावृत्ति की आवश्यकता होती है।

• पायथन में नियंत्रण संरचनाओं (Conditional और Looping Statements) का प्रयोग करना समझेंगे।

• फ़ंक्शन्स और कंट्रोल स्टेटमेंट्स का उपयोग करके सरल पायथन प्रोग्राम विकसित करना समझेंगे।

मॉड्यूल संरचना (Module Structure)

सत्र 1. कंट्रोल स्ट्रक्चर (Control Structures)

सत्र 2. पायथन में फंक्शन (Functions in Python)

सत्र 1. कंट्रोल स्ट्रक्चर्स (Control Structures)

प्रोग्रामों में स्टेटमेंट्स की एक श्रृंखला ऊपर से नीचे क्रम में निष्पादित होती है। किसी प्रोग्राम में स्टेटमेंट्स कथनों के निष्पादन के क्रम को फ्लो ऑफ कंट्रोल कहा जाता है। नियंत्रण प्रवाह को नियंत्रण संरचनाओं (कंट्रोल स्ट्रक्चर्स) के माध्यम से लागू किया जा सकता है।

जब प्रोग्राम को विभिन्न परिस्थितियों के अनुसार कुछ निर्णय लेने की आवश्यकता होती है, तब इसे कंट्रोल फ्लो स्टेटमेंट्स के द्वारा पूरा किया जाता है। पायथन दो प्रकार की नियंत्रण संरचनाओं का सपोर्ट करता है— सेलेक्शन और रिपीटिशन। आइए, इन दोनों प्रकार के कंट्रोल स्ट्रक्चर्स पर क्रमशः विस्तार से चर्चा करते है।

पायथन में तीन कंट्रोल फ्लो स्टेटमेंट होते हैं —if, for और while । इनमें if और for सेलेक्शन प्रकार के होते हैं, जबकि while रिपीटिशन प्रकार का होता है।

इस सत्र में आप कंट्रोल स्ट्रक्चर्स, उनकी सिंटैक्स और पायथन प्रोग्रामिंग में उनके उपयोग को समझेंगे। साथ ही break (ब्रेक) और continue (कंटीन्यू) स्टेटमेंट्स पर भी चर्चा करेंगे।

1.1 If स्टेटमेंट

वास्तविक जीवन में जब हमें कुछ निर्णय लेने होते हैं, तो उन्हीं निर्णयों के आधार पर हम यह तय करते हैं कि आगे क्या करना है। इसी प्रकार की स्थितियाँ प्रोग्रामिंग में भी उत्पन्न होती हैं, जहाँ हमें कुछ निर्णय लेने होते हैं और उन्हीं के आधार पर हम अगले कोड ब्लॉक का निष्पादन करते हैं।

प्रोग्रामिंग भाषाओं में निर्णय-निर्माण से संबंधित स्टेटमेंट्स प्रोग्राम के निष्पादन के प्रवाह की दिशा तय करते हैं। पायथन में उपलब्ध निर्णय-निर्माण स्टेटमेंट्स निम्नलिखित हैं:

- if statement
- if...else statements
- if-elif ladder

If स्टेटमेंट

if स्टेटमेंट का उपयोग किसी कंडीशन की जाँच करने के लिए किया जाता है: यदि कंडीशन true होती है, तो हम स्टेटमेंट्स के एक ब्लॉक (जिसे if-ब्लॉक कहा जाता है) को रन करते हैं।

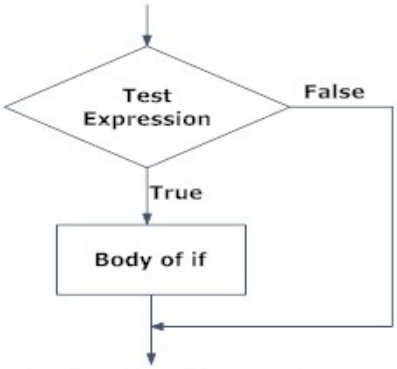
if स्टेटमेंट का सिंटैक्स

```
if test expression:
        statement(s)
```

प्रोग्राम टेस्ट एक्सप्रेशन (परीक्षण अभिव्यक्ति) का मूल्यांकन करता है और केवल स्टेटमेंट का निष्पादन करता है जब यह एक्सप्रेशन True होती है। यदि टेक्स्ट एक्सप्रेशन False होती है, तो स्टेटमेंट का निष्पादन नहीं किया जाता।

पायथन में if स्टेटमेंट का मुख्य भाग इंडेंटेशन द्वारा दर्शाया जाता है। यह भाग इंडेंटेशन से शुरू होता है और पहली अइंडेंटेड वाली लाइन इसके अंत को दर्शाती है। पायथन में शून्य के अलावा अन्य सभी मानों को True माना जाता है, जबकि None और 0 को False के रूप में समझा जाता है।

if स्टेटमेंट का फ़्लोचार्ट



if स्टेटमेंट का उदाहरण

आइए कंट्रोल स्ट्रक्चर को समझने के लिए एक उदाहरण लेते हैं।

```
File  Edit  Format  Run  Options  Window  Help
# Program to check the eligibility of vote using if statement
age = int(input("Enter your age : "))
if age >= 18:
     print("You are ELIGIBLE to vote")
                                        Ln: 4  Col: 38
```

जब आप प्रोग्राम को तीन अलग-अलग इनपुट आयु 20, 18 और 17 के साथ रन करते हैं, तो आउटपुट इस प्रकार होता है:

```
File  Edit  Shell  Debug  Options  Window  Help
    ==========================
    Enter your age : 20
    You are ELIGIBLE to vote
>>>
    ==========================
    Enter your age : 18
    You are ELIGIBLE to vote
>>>
    ==========================
    Enter your age : 16
>>>
```

उपरोक्त उदाहरण में, यदि उपयोगकर्ता द्वारा दर्ज की गई आयु 18 से अधिक है, तो संदेश “You are ELIGIBLE to vote” प्रिंट होता है। यदि स्टेटमेंट True होती है, तो इंडेंट किए गए स्टेटमेंट का निष्पादन किया जाता है, अन्यथा नहीं।

इंडेंटेशन यह दर्शाता है कि स्टेटमेंट का निष्पादन कंडीशन पर निर्भर करता है। if स्टेटमेंट के अंतर्गत एक ब्लॉक में आने वाले स्टेटमेंट्स की संख्या की कोई सीमा नहीं होती है।

if...else स्टेटमेंट

if स्टेटमेंट का एक रूप if...else स्टेटमेंट कहलाता है, जो दो वैकल्पिक पाथ लिखने की अनुमति देता है। कंडीशन यह निर्धारित करती है कि इनमें से कौन-सा मार्ग निष्पादित (execute) होगा।

if...else स्टेटमेंट का सिंटैक्स

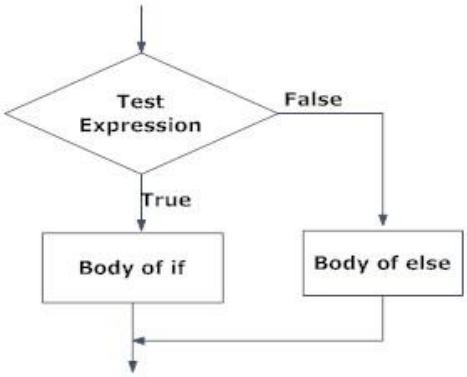
if....else स्टेटमेंट का सिंटैक्स निम्नलिखित है।

```
if test expression:
    Body of if
else:
    Body of else
```

if...else स्टेटमेंट टेस्ट एक्सप्रेशन का मूल्यांकन करता है और केवल तब if वाले body को निष्पादित करता है जब कंडीशन True होती है।

यदि कंडीशन False होती है, तो else वाले भाग का निष्पादन किया जाता है। ब्लॉक्स को अलग करने के लिए इंडेंटेशन का उपयोग किया जाता है।

if...else स्टेटमेंट का फ़्लोचार्ट



if...else स्टेटमेंट का उदाहरण

स्टेटमेंट

आइए अब if else संरचना के उपयोग को स्पष्ट करने के लिए उपरोक्त कोड को संशोधित करें।

File Edit Format Run Options Window Help

```
# Program to check the eligibility of vote using if statement
age = int(input("Enter your age : "))
if age >= 18:
     print("You are ELIGIBLE to vote")
else:
    print("You are NOT ELIGIBLE to vote")
```

Ln: 6 Col: 4

जब आप प्रोग्राम को तीन अलग-अलग इनपुट आयु 18, 20, और 17 के साथ रन करते हैं, तो आउटपुट इस प्रकार होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
    ========================== RESTART:
    Enter your age : 18
    You are ELIGIBLE to vote
>>>
    ========================== RESTART:
    Enter your age : 20
    You are ELIGIBLE to vote
>>>
    ========================== RESTART:
    Enter your age : 17
    You are NOT ELIGIBLE to vote
>>>
```

उपरोक्त उदाहरण में, यदि व्यक्ति द्वारा दर्ज की गई आयु 18 वर्ष या उससे अधिक है, तो वह वोट कर सकता है। अन्यथा, वह व्यक्ति वोट करने के लिए पात्र नहीं है।

if...elif...else स्टेटमेंट

ऐसी स्थिति हो सकती है जब आपको कई कंडीशनो की जाँच करनी हो और ये सभी कंडीशनो एक-दूसरे से स्वतंत्र हों। ऐसे कंडीशन में आप elif स्टेटमेंट का उपयोग कर सकते हैं, जिससे if कंडीशन के बाद या if और else कंट्रोल स्ट्रक्चर के बीच कई कंडीशनल एक्सप्रेशन शामिल की जा सकती हैं।

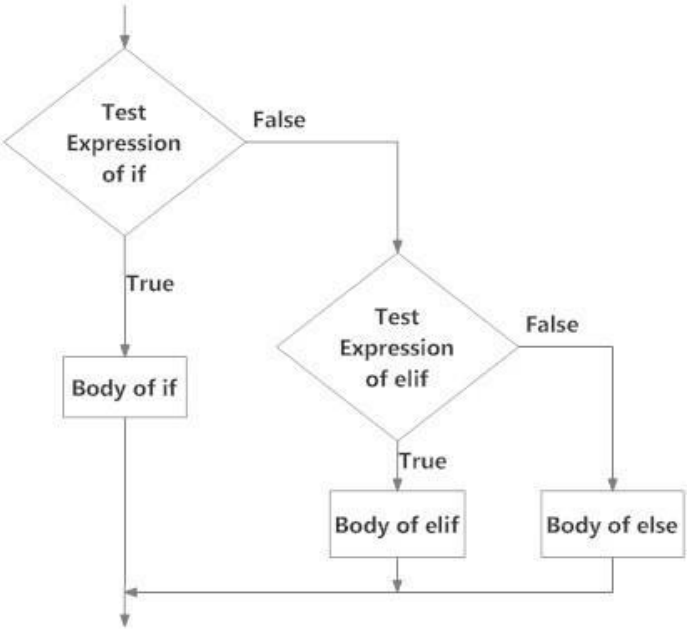
if...elif...else स्टेटमेंट का सिंटैक्स

if … elif … else का उपयोग करते हुए एक सेलेक्शन स्ट्रक्चर का सिंटैक्स नीचे दिया गया है।

```
if test expression:
    Body of if
elif test expression:
    Body of elif
else:
    Body of else
```

elif का अर्थ else if का संक्षिप्त रूप है। यह हमें कई एक्सप्रेशन की जाँच करने की अनुमति देता है। यदि if की कंडीशन False होती है, तो यह अगले elif ब्लॉक की कंडीशन को जाँचता है और इसी प्रकार आगे बढ़ता है। यदि सभी कंडीशन असत्य होती हैं, तो else का भाग निष्पादित किया जाता है। कई if...elif...else ब्लॉकों में से केवल एक ही ब्लॉक कंडीशन के अनुसार निष्पादित होता है। if ब्लॉक के साथ केवल एक else ब्लॉक हो सकता है, लेकिन इसके साथ कई elif ब्लॉक हो सकते हैं।

if...elif...else स्टेटमेंट का फ़्लोचार्ट



if...elif...else स्टेटमेंट का उदाहरण

आइए अब if…elif…else संरचना के उपयोग को स्पष्ट करने के लिए उपरोक्त कोड को संशोधित करें।

```
File  Edit  Format  Run  Options  Window  Help
# Program to check the eligibility of vote
# using if elif else statement
age = int(input("Enter your age : "))
if age == 18:
     print("You become just ELIGIBLE to vote")
elif age > 18:
     print ("You are ELIGIBLE to vote")
else:
     print ("You are NOT ELIGIBLE to vote")
                                        Ln: 5  Col: 28
```

जब आप प्रोग्राम को तीन अलग-अलग इनपुट आयु 18,19,17 के साथ रन करते हैं, तो आउटपुट इस प्रकार होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
    Enter your age : 18
    You become just ELIGIBLE to vote
>>>
    ========================== RESTART:
    Enter your age : 19
    You are ELIGIBLE to vote
>>>
    ========================== RESTART:
    Enter your age : 17
    You are NOT ELIGIBLE to vote
>>>
```

Nested if statements (नेस्टेड इफ स्टेटमेंट्स)

हम एक if...elif...else स्टेटमेंट के अंदर एक और if...elif...else स्टेटमेंट रख सकते हैं। इसे कंप्यूटर प्रोग्रामिंग में नेस्टिंग कहा जाता है।

इनमें से किसी भी संख्या में स्टेटमेंट्स को एक दूसरे के अंदर नेस्ट किया जा सकता है। नेस्टिंग के स्तर का पता लगाने का एकमात्र तरीका इंडेंटेशन है। यह भ्रमित करने वाला हो सकता है, इसलिए यदि संभव हो तो इससे बचना चाहिए।

Nested if statements (नेस्टेड इफ स्टेटमेंट्स) का उदाहरण

आइए अब if…elif…else संरचना के उपयोग को स्पष्ट करने के लिए उपरोक्त कोड को संशोधित करें।

```
File Edit Format Run Options Window Help
# Program to check the eligibility of vote
# using nested if statement
age = int(input("Enter your age : "))
if age >= 18:
    if age == 18:
        print("You become just ELIGIBLE to vote")
    else:
        print ("You are ELIGIBLE to vote")
else:
    print ("You are NOT ELIGIBLE to vote")
Ln: 5 Col: 0
```

जब आप प्रोग्राम को तीन अलग-अलग इनपुट आयु 18,19,17 के साथ रन करते हैं, तो आउटपुट इस प्रकार होगा:

```
File Edit Shell Debug Options Window Help
    Enter your age : 18
    You become just ELIGIBLE to vote
>>>
    ========================== RESTART:
    Enter your age : 19
    You are ELIGIBLE to vote
>>>
    ========================== RESTART:
    Enter your age : 17
    You are NOT ELIGIBLE to vote
>>>
```

उदाहरण: यह जाँचने के लिए कि कोई संख्या धनात्मक, ऋणात्मक या शून्य है, नेस्टेड if का उपयोग करते हुए एक प्रोग्राम लिखें।

```
File Edit Format Run Options Window Help
# Program to check the number is positive,
# negative or zero using nested if statement
number = int(input("Enter the number : "))
if number > 0:
    print("The number is positive")
elif number < 0:
    print("The number is negative")
else:
    print("The number is zero")
Ln: 3 Col: 0
```

जब आप प्रोग्राम को चार अलग-अलग इनपुट संख्याओं के साथ रन करते हैं, तो आउटपुट इस प्रकार होगा:

```
File Edit Shell Debug Options Window Help
    Enter the number : 11
    The number is positive
>>>
    ================= RESTART:
    Enter the number : -5
    The number is negative
>>>
    ================= RESTART:
    Enter the number : 0
    The number is zero
>>>
```

उदाहरण: सड़क क्रॉसिंग पर सिग्नल के रंग के अनुसार उपयुक्त संदेश प्रदर्शित करने के लिए एक प्रोग्राम लिखें।

```
File Edit Format Run Options Window Help
# Program to display the message as per
# colour of signal at the road crossing
signal = input("Enter the colour : ")
if signal == "RED" or signal == "red":
    print("STOP")
elif signal == "ORANGE" or signal == "orange
    print("Go Slow")
elif signal == "GREEN" or signal == "green":
    print("Go Ahed!")
else:
    print("Wait for the Signal to glow")
                                    Ln: 10 Col: 5
```

जब आप प्रोग्राम को चार अलग-अलग इनपुट संख्याओं के साथ रन करते हैं, तो आउटपुट इस प्रकार होगा:

```
File Edit Shell Debug Options Window Help
    Enter the colour : red
    STOP
>>>
    ========================== |
    Enter the colour : ORANGE
    Go Slow
>>>
    ========================== |
    Enter the colour : GREEN
    Go Ahed!
>>>
    ========================== |
    Enter the colour :
    Wait for the Signal to glow
>>>
```

elif की संख्या जाँची जाने वाली कंडीशनो की संख्या पर निर्भर करती है। यदि पहली कंडीशन false है, तो अगली कंडीशन की जाँच की जाती है और इसी तरह आगे बढ़ते है। यदि कोई एक कंडीशन सही है, तो संबंधित इंडेंटेड ब्लॉक निष्पादित होता है और if स्टेटमेंट समाप्त हो जाता है।

उदाहरण: आइए दो संख्याओं पर बुनियादी अंकगणितीय संक्रियाएँ (arithmetic operations) करने के लिए एक साधारण कैलकुलेटर बनाने का प्रोग्राम लिखें। प्रोग्राम को ऐसे करना चाहिए:

- उपयोगकर्ता से दो संख्याएँ स्वीकार करें।
- उपयोगकर्ता को इनमें से कोई ऑपरेटर (+, -, *, /) इनपुट करने के लिए कहे।

- यदि उपयोगकर्ता कुछ और दर्ज करता है तो एक त्रुटि (error) संदेश प्रदर्शित हो।
- ऑपरेटर "-" के मामले में केवल धनात्मक अंतर (positive difference) प्रदर्शित करें।
- यदि उपयोगकर्ता दूसरी संख्या 0 दर्ज (enter) करता है और ऑपरेटर '/' दर्ज किया गया है तो एक संदेश "Please enter a value other than 0" प्रदर्शित करें।

File Edit Format Run Options Window Help

```
# Program of Calculator to perform
# Four basic arithmatic operations
result = 0
val1 = float(input("Enter value 1: "))
val2 = float(input("Enter value 2: "))
op = input("Enter the Operator (+,-,*,/): ")
if op == "+":
     result = val1 + val2
elif op == "-":
     if val1 > val2:
          result = val1 - val2
     else:
          result = val2 - val1
elif op == "*":
     result = val1 * val2
elif op == "/":
     if val2 == 0:
          print("Error ! Division by Zero")
     else:
          result = val1/val2
else:
     print("Wrong input, Program terminated")
print("The Result is : ", result)
```

Ln: 26 Col: 0

उपरोक्त प्रोग्राम में, ऑपरेटरों "-" और "/" के लिए, elif ब्लॉक के भीतर एक if...else शर्त मौजूद है। इसे नेस्टेड if कहा जाता है। if...else स्टेटमेंट्स के अंदर नेस्टिंग के कई स्तर हो सकते हैं।

जैसा कि दर्शाया गया है, if elif संरचना का उपयोग करना संभव है। पायथन के नवीनतम संस्करण में, एक अधिक शक्तिशाली और लचीला निर्माण उपलब्ध है जिसे स्ट्रक्चरल पैटर्न मैचिंग कहा जाता है। इसका उपयोग एक साधारण स्विच स्टेटमेंट के रूप में किया जा सकता है, लेकिन यह और भी बहुत कुछ करने में सक्षम है।

© PSS

```
File  Edit  Shell  Debug  Options  Window  Help
    Enter value 1: 45
    Enter value 2: 68
    Enter the Operator (+,-,*,/): +
    The Result is :  113.0
>>>
    ========================== RESTART:
    Enter value 1: 78
    Enter value 2: 89
    Enter the Operator (+,-,*,/): -
    The Result is :  11.0
>>>
    ========================== RESTART:
    Enter value 1: 85
    Enter value 2: 45
    Enter the Operator (+,-,*,/): -
    The Result is :  40.0
>>>
    ========================== RESTART:
    Enter value 1: 12
    Enter value 2: 5
    Enter the Operator (+,-,*,/): *
    The Result is :  60.0
>>>
    ========================== RESTART:
    Enter value 1: 90
    Enter value 2: 0
    Enter the Operator (+,-,*,/): /
    Error ! Division by Zero
    The Result is :  0
>>>
    ========================== RESTART:
    Enter value 1: 50
    Enter value 2: 10
    Enter the Operator (+,-,*,/): /
    The Result is :  5.0
```

स्ट्रक्चरल पैटर्न मैचिंग (Structural pattern matching)

स्ट्रक्चरल पैटर्न मैचिंग पायथन में match…case स्टेटमेंट और पैटर्न सिंटैक्स को प्रस्तुत करता है। match…case स्टेटमेंट अन्य प्रोग्रामिंग भाषाओं के switch...case की तरह ही मूल संरचना का पालन करता है। यह किसी ऑब्जेक्ट को लेता है, उसे एक या अधिक मैच पैटर्न से तुलना करता है और यदि मैच करता है तो संबंधित एक्शन करता है।

पायथन मैचिंग करने के लिए केसों की सूची को ऊपर से नीचे तक क्रम में जाँचता है। जैसे ही पहला मिलान मिलता है, पायथन संबंधित case ब्लॉक के स्टेटमेंट्स को निष्पादित करता है, फिर मैच ब्लॉक के अंत तक पहुँचकर प्रोग्राम के शेष भाग को जारी रखता है। केसों के बीच कोई fall-through नहीं होता, लेकिन आप अपनी लॉजिक को इस प्रकार डिज़ाइन कर सकते हैं कि एक ही case ब्लॉक में कई संभावित केसों को संभाला जा सके। आइए, स्ट्रक्चरल पैटर्न मैचिंग के उपयोग को समझने के लिए एक उदाहरण देखते हैं।

उदाहरण: स्ट्रक्चरल पैटर्न मैचिंग के उपयोग को दर्शाने के लिए एक प्रोग्राम लिखिए।

```
File  Edit  Format  Run  Options  Window  Help
# Program to illustrate the
# structural pattern matching
digit=int(input("Enter any digit 0-9 :"))
match digit:
     case 0: print("Zero")
     case 1: print("One")
     case 2: print("Two")
     case 3: print("Three")
     case 4: print("Four")
     case 5: print("Five")
     case 6: print("Six")
     case 7: print("Seven")
     case 8: print("Eight")
     case 9: print("Nine")
     case _: print("Invalid Input")
                                        Ln: 8  Col: 10
```

उपरोक्त प्रोग्राम में, उपयोगकर्ता द्वारा दर्ज किए गए मान वेरिएबल digits में संग्रहीत किए जाते हैं। case के बाद लिखे गए मानों की एक-एक करके digit के मान से तुलना की जाएगी। यदि digit = 0 है जैसा कि case 0 में निर्दिष्ट है, तो zero प्रिंट करने का स्टेटमेंट निष्पादित होगा। यदि digit 0 के बराबर नहीं है, तो नियंत्रण case 1 के रूप में निर्दिष्ट अगले case स्टेटमेंट पर चला जाएगा। यदि digit =1 है, तो one प्रिंट करने का स्टेटमेंट निष्पादित होगा। इसी तरह, यह case 2 से case 9 तक चलता रहेगा। अंतिम case के लिए, यदि digit 0 से 9 तक के अंकों के अलावा कुछ और है, तो यह "Invalid input" प्रिंट करेगा।

```
File  Edit  Shell  Debug  Options  Window  Help
    Enter any digit 0-9 :0
    Zero
>>>
    ==========================
    Enter any digit 0-9 :7
    Seven
>>>
    ==========================
    Enter any digit 0-9 :11
    Invalid Input
>>>
```

असाइनमेंट

1. एक प्रोग्राम लिखिए जो यह जाँच करे कि कोई संख्या 7 से विभाज्य है या नहीं।
2. एक प्रोग्राम लिखिए जो यह जाँच करे कि कोई अक्षर स्वर है या व्यंजन है।
3. एक प्रोग्राम लिखिए जो महीने का नंबर इनपुट करें और महीने का नाम प्रिंट करें।
4. उपयोगकर्ता द्वारा दी गई भुजाओं की लंबाई के आधार पर यह जाँच करने के लिए एक प्रोग्राम लिखिए कि त्रिभुज समबाहु, समद्विबाहु या विषमबाहु है।

1.3 इंडेंटेशन (Indentation)

अधिकांश प्रोग्रामिंग भाषाओं में किसी ब्लॉक के भीतर के स्टेटमेंट्स को कर्ली ब्रैकेट्स {} के अंदर रखा जाता है। हालांकि, पायथन ब्लॉक तथा नेस्टेड ब्लॉक संरचनाओं के लिए इंडेंटेशन का उपयोग करता है। किसी स्टेटमेंट की शुरुआत में आने वाले स्पेस और टैब को इंडेंटेशन कहा जाता है।

पायथन में समान स्तर का इंडेंटेशन स्टेटमेंट्स को एक ही कोड ब्लॉक में जोड़ता है। इंटरप्रेटर इंडेंटेशन स्तरों की बहुत सख्ती से जाँच करता है और यदि इंडेंटेशन सही नहीं होता है, तो सिंटैक्स एरर देता है। सामान्यतः प्रत्येक इंडेंटेशन स्तर के लिए एक टैब का उपयोग करना प्रचलित है।

निम्नलिखित प्रोग्राम में if-else स्टेटमेंट के दो ब्लॉक हैं और प्रत्येक ब्लॉक के स्टेटमेंट्स को समान संख्या के स्पेसेस या टैब्स के साथ इंडेंट किया गया है।

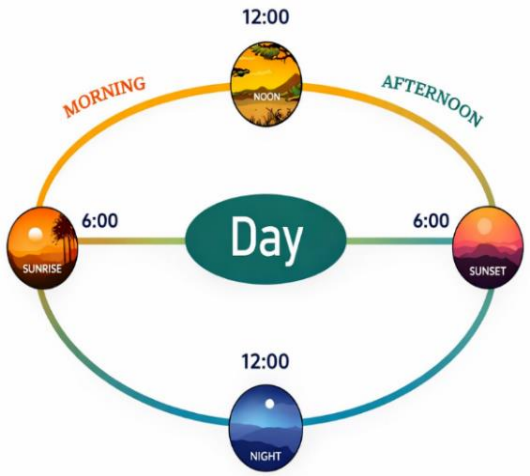
```
File  Edit  Format  Run  Options  Window  Help
# Programme to find larger of two numbers
# to illustrate the identation of block

num1=int(input("Enter first number  : "))
num2=int(input("Enter second number : "))

if num1 > num2:
     print("First number is larger")
     print("Bye")
else:
     print("Second number is larger")
     print("Bye Bye")
                                    Ln: 4  Col: 36
```

उपरोक्त प्रोग्राम में, प्रोग्राम में लिए गए num1 और num2 के मानों के लिए शर्त num1 > num2 गलत है। इसलिए, else से जुड़े स्टेटमेंट्स का ब्लॉक निष्पादित होगा जैसा कि आउटपुट में दिखाया गया है।

```
File  Edit  Shell  Debug  Options  Window  Help
     Enter first number  : 25
     Enter second number : 23
     First number is larger
     Bye
>>>
     ==========================
     Enter first number  : 45
     Enter second number : 56
     Second number is larger
     Bye Bye
```

1.4 रिपीटिशन (Repetition)

कभी-कभी हमें कार्यों को दोहराने की आवश्यकता होती है, जैसे बिजली के बिल का भुगतान हर महीने करना। आइए एक उदाहरण लेते हैं जो प्रकृति एक पुनरावृत्ति प्रक्रिया को भी दर्शाता है। चित्र 1.3 दिन के चरणों जैसे सुबह, दोपहर और शाम को दर्शाता है। ये चरण प्रतिदिन उसी क्रम में दोहराए जाते हैं।



चित्र 1.3: दिन के चरण

एक अन्य उदाहरण जहाँ हम चरणों को दोहराते हैं, वह है कक्षा के प्रत्येक विद्यार्थी के लिए अलग-अलग औसत या माध्य अंकों की गणना करना। पहले, हम एक विद्यार्थी द्वारा 5 विषयों में प्राप्त अंकों के माध्य की गणना करने के चरणों को सूचीबद्ध करते हैं:

1. विद्यार्थी द्वारा पाँच विषयों में प्राप्त अंकों पढ़ें।
2. सभी 5 विषयों के अंकों का योग निकालें।
3. इस योग को 5 से विभाजित करें।
4. प्राप्त परिणाम को माध्य के रूप में संग्रहीत किया जाएगा।

उपरोक्त चरणों को कक्षा के प्रत्येक विद्यार्थी के लिए दोहराया जाएगा ताकि औसत अंक प्राप्त किए जा सकें। इस प्रकार के दोहराव को इटरेशन भी कहा जाता है।

किसी प्रोग्राम में स्टेटमेंट्स के एक समूह की repetition लूपिंग कंस्ट्रक्ट्स के माध्यम से संभव होती है। इसे और समझने के लिए, आइए पहले 5 प्राकृतिक संख्याओं को प्रिंट करने वाले निम्नलिखित प्रोग्राम को देखें।

```
File  Edit  Shell  Debug  Options  Window  Help
      Python 3.12.3 (main, Jar
      Type "help", "copyright'
>>>
      ========================
      First 5 natural numbers
      1
      2
      3
      4
      5
>>>
```

```
File Edit Format Run Options Window Help
# Program to print first
# five natural numbers
print("First 5 natural numbers")
print(1)
print(2)
print(3)
print(4)
print(5)
Ln: 9 Col: 0
```

उपरोक्त प्रोग्राम में 5 अलग-अलग प्राकृतिक संख्याओं को प्रिंट करने के लिए print () फ़ंक्शन का 5 बार उपयोग किया गया है। लेकिन यदि हमें पहले 100,000 प्राकृतिक संख्याओं को प्रिंट करना हो, तो 100,000 बार print स्टेटमेंट लिखना प्रभावी नहीं होगा। ऐसी स्थितियों में प्रोग्राम में loop या रिपीटिशन का उपयोग करना अधिक उपयुक्त होता है।

लूपिंग कंस्ट्रक्ट्स प्रोग्राम में किसी स्टेटमेंट्स के समूह को एक शर्त के आधार पर बार-बार निष्पादित करने की सुविधा प्रदान करती हैं। लूप के अंदर के स्टेटमेंट्स तब तक बार-बार चलते रहते हैं, जब तक दी गई लॉजिकल कंडीशन true बनी रहती है।

यह कंडीशन एक वेरिएबल के मान के आधार पर जाँची जाती है, जिसे लूप कंट्रोल वेरिएबल कहा जाता है। जब यह कंडीशन False हो जाती है, तब लूप terminate हो जाता है।

यह प्रोग्रामर की जिम्मेदारी होती है कि वह सुनिश्चित करे कि यह कंडीशन अंततः False हो जाए, ताकि लूप से बाहर निकलने की स्थिति बने और प्रोग्राम infinite loop में न फँसे। उदाहरण के लिए, यदि हमने कंडीशन count <= 100000 निर्धारित नहीं की होती, तो प्रोग्राम कभी समाप्त नहीं होता।

पायथन में दो प्रकार की लूपिंग कंस्ट्रक्ट्स होती हैं — for और while। आइए, इन लूपिंग संरचनाओं को विस्तार से समझें।

1.4.1 फॉर लूप (For Loop)

for स्टेटमेंट का उपयोग range of values या निश्चित संख्या की सीक्वेंस पर इटरेशन करने के लिए किया जाता है। for लूप श्रेणी के प्रत्येक आइटम के लिए एक-एक बार निष्पादित होता है। ये मान संख्यात्मक (numeric), स्ट्रिंग (string), सूची (list) या टपल अथवा ट्यूपल (tuple) हो सकते हैं।

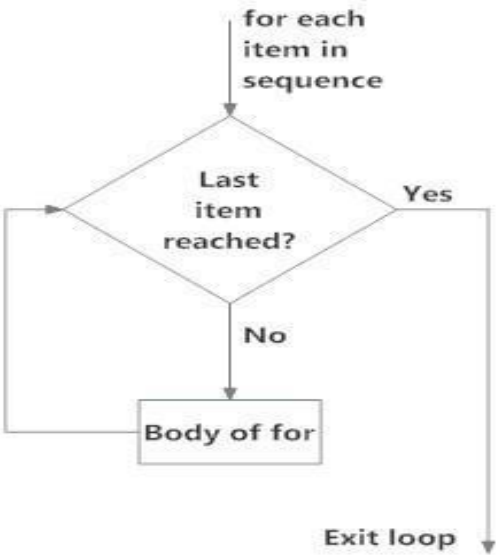
फॉर लूप का सिंटैक्स (Syntax of the For Loop)

```
for <control-variable> in <sequence/items in range>:
    <statements inside body of the loop>
```

लूप के प्रत्येक इटरेशन के साथ, कंट्रोल वेरिएबल यह जाँचता है कि रेंज के सभी मानों को traverse कर लिया गया है या नहीं। जब रेंज के सभी तत्व समाप्त हो जाते हैं, तब कंट्रोल for लूप के तुरंत बाद वाले स्टेटमेंट पर चला जाता है। for लूप का उपयोग करते समय पहले से ही यह ज्ञात होता है कि लूप कितनी बार निष्पादित होगा।

फॉर लूप का फ़्लोचार्ट (Flowchart of for Loop)

for लूप के निष्पादन को दर्शाने वाला फ़्लोचार्ट चित्र 1.4 में दिया गया है।



चित्र 1.4: for लूप का फ़्लो चार्ट

उदाहरण: एक सूची में संग्रहीत सभी संख्याओं का योग ज्ञात करने के लिए for लूप का उपयोग करते हुए एक प्रोग्राम लिखिए।

```
File  Edit  Format  Run  Options  Window  Help
# Program to find the sum of all numbers stored in a list
numbers = [6,5,3,7,9,1,11]  # List of numbers
sum = 0                     # variable to store the sum
for val in numbers:         # control variable to iterate
    sum = sum + val         # Adding the no. in the list sum
print("The sum is :",sum)   # Sum of all no. stored in litst
                                                    Ln: 8  Col: 0
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट यह होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
=========================
The sum is : 42
>>>
```

उपरोक्त प्रोग्राम में, numbers पूर्णांकों का एक अनुक्रम है। वेरिएबल val for लूप के विभिन्न इटरेशन में विभिन्न पूर्णांकों को दर्शाता है। इसलिए, यह सूची के सभी तत्वों का योग प्रिंट करता है।

उदाहरण: एक सूची में संग्रहीत संख्याओं को सम या विषम के रूप में जाँचने के लिए एक प्रोग्राम लिखिए।

```
File Edit Format Run Options Window Help
# Program to check the number as EVEN or ODD from the list
numbers = [6,5,8,7,4,1,11]          # List of numbers
for val in numbers:                 # control variable
    if (val % 2) ==0:               # Check for EVEN number
        print(val,'is EVEN number') # Number is EVEN
    else:
        print(val,'is ODD number')  # Number is ODD
Ln: 5 Col: 38
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट यह होगा:

```
File Edit Shell Debug Options Window Help
========================
6 is EVEN number
5 is ODD number
8 is EVEN number
7 is ODD number
4 is EVEN number
1 is ODD number
11 is ODD number
```

हमने उन प्रोग्रामों पर चर्चा की जो पायथन में for लूप का उपयोग करने के उदाहरण हैं। आइए यह भी देखें कि for लूप के साथ रेंज फ़ंक्शन का उपयोग कैसे किया जा सकता है।

रेंज() फ़ंक्शन

Range() पायथन में एक बिल्ट-इन फ़ंक्शन है। range () फ़ंक्शन का सिंटैक्स है:

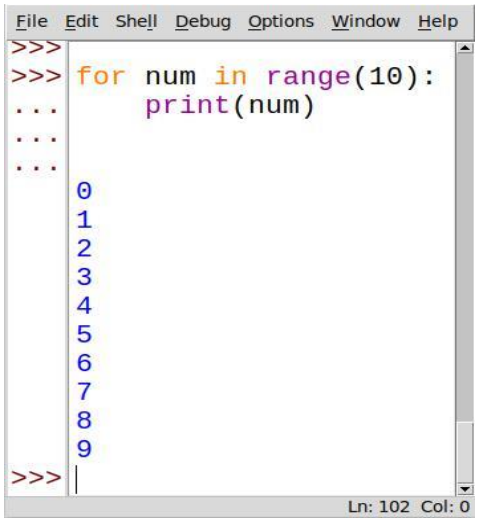
range (start, stop, step)

इसका उपयोग दिए गए स्टार्ट मान से स्टॉप मान तक, दिए गए स्टेप मान के अंतर के साथ, पूर्णांकों का एक अनुक्रम युक्त एक सूची बनाने के लिए किया जाता है। निम्नलिखित उदाहरण विभिन्न स्थितियों के लिए range() फ़ंक्शन के उपयोग को दर्शाते हैं।

```
File Edit Shell Debug Options Window Help
>>> # Illustrate the range() function
>>> # Print the single digit numbers
>>> list(range(0,10,1))
    [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> # Print the sigle digit odd numbers
>>> list(range(1,10,2))
    [1, 3, 5, 7, 9]
>>> # Print the single digit even numbers
>>> list(range(2,10,2))
    [2, 4, 6, 8]
>>> # Print the double digit numbers in step 10
>>> list(range(10,99,10))
    [10, 20, 30, 40, 50, 60, 70, 80, 90]
>>> # Print the sigle digit number in decresing order
>>> list(range(9,0,-1))
    [9, 8, 7, 6, 5, 4, 3, 2, 1]
>>> # When start and step not specified
>>> list(range(10))
    [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>>
Ln: 15 Col: 19
```

फ़ंक्शन range () में, start, stop और step पैरामीटर हैं। फ़ंक्शन्स पर अधिक जानकारी अगले अध्याय में शामिल है। start और step पैरामीटर वैकल्पिक हैं। यदि start मान निर्दिष्ट नहीं है, तो डिफ़ॉल्ट रूप से सूची 0 से शुरू होती है। यदि step भी निर्दिष्ट नहीं है, तो डिफ़ॉल्ट रूप से प्रत्येक इटरेशन में मान 1 से बढ़ता है। यह डिफ़ॉल्ट रूप से start=0 और step=1 लेगा।

range () फ़ंक्शन के सभी पैरामीटर पूर्णांक होने चाहिए। step पैरामीटर एक धनात्मक या ऋणात्मक पूर्णांक हो सकता है जिसमें शून्य को छोड़कर। range () फ़ंक्शन के पैरामीटर step का ऋणात्मक मान एक घटता हुआ क्रम उत्पन्न करेगा जैसा कि उपरोक्त उदाहरण में दर्शाया गया है। फ़ंक्शन range () का उपयोग अक्सर for लूप में संख्याओं का एक क्रम उत्पन्न करने के लिए किया जाता है जैसा कि नीचे दर्शाया गया है।



उपरोक्त पायथन कोड में range(10) फ़ंक्शन संख्याओं का एक sequence में [0, 1, 2, 3, 4, 5, 6, 7, 8, 9] उत्पन्न करता है। वेरिएबल num इस क्रम के तत्वों को for लूप की प्रत्येक इटरेशन में एक-एक करके ग्रहण करता है। हर इटरेशन में print(num) स्टेटमेंट निष्पादित होता है। इसलिए, इस क्रम के सभी तत्व आउटपुट में एक-एक करके प्रिंट हो जाते हैं, जैसा कि आउटपुट में दिखाया गया है।

असाइनमेंट

1. for लूप का उपयोग करते हुए दिए गए लिस्ट से विषम (odd) संख्याएँ प्रिंट करने के लिए एक प्रोग्राम लिखिए।
2. एक ऐसी लिस्ट प्रदर्शित करने के लिए एक प्रोग्राम लिखिए जो दी गई सूची के तत्वों के वर्गों को संग्रहीत करती है।
3. for लूप का उपयोग करते हुए एक प्रोग्राम लिखिए जो उपयोगकर्ता से 3 बार एक हॉबी (रुचि) पूछे, और प्रत्येक को हॉबीस (hobbies) नामक लिस्ट में जोड़ दे। अंत में हॉबीस लिस्ट के सभी तत्व प्रदर्शित कीजिए।

1.4.2 व्हाइल लूप (While Loop)

while स्टेटमेंट आपको किसी कंडीशन में true रहने तक स्टेटमेंट्स के एक ब्लॉक को बार-बार निष्पादित करने की अनुमति देता है। while लूप की कंट्रोल कंडीशन लूप के भीतर के किसी भी स्टेटमेंट के निष्पादन से पहले जाँची जाती है। प्रत्येक इटरेशन के बाद यह कंडीशन फिर से जाँची जाती है और जब तक कंडीशन सत्य रहती है, लूप चलता रहता है। जब यह कंडीशन false हो जाती है, तब लूप के बॉडी के स्टेटमेंट्स निष्पादित नहीं होते और नियंत्रण while लूप के तुरंत बाद वाले स्टेटमेंट पर चला जाता है।

यदि while लूप की शर्त शुरू में ही असत्य हो, तो लूप का बॉडी एक बार भी निष्पादित नहीं होता। while स्टेटमेंट एक प्रकार का लूपिंग स्टेटमेंट है। while स्टेटमेंट के साथ एक वैकल्पिक else क्लॉज भी हो सकता है।

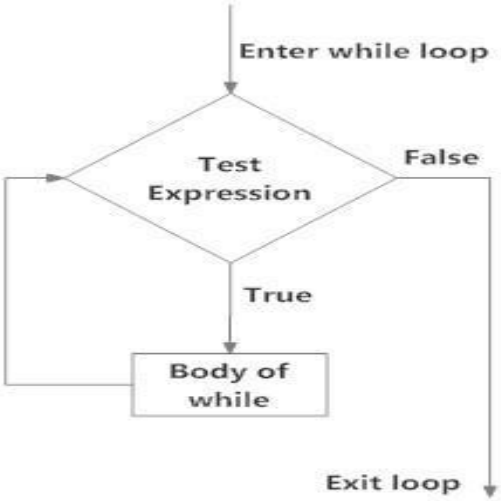
व्हाइल लूप का सिंटैक्स (Syntax of while Loop)

```
while test expression:
    body of while
```

while लूप के मुख्य भाग के भीतर लिखे गए स्टेटमेंट्स ऐसे होने चाहिए कि अंततः कंडीशन false हो जाए, अन्यथा infinite loop बन जाएगा, जिससे प्रोग्राम में logical error उत्पन्न होगी।

व्हाइल लूप का फ़्लोचार्ट (Flowchart of while Loop)

while लूप का फ़्लोचार्ट चित्र 1.5 में दिखाया गया है।



चित्र 1.5: while लूप का फ़्लोचार्ट

उदाहरण: निम्नलिखित प्रोग्राम while लूप के उपयोग से स्पष्ट करेंगे।

```
File Edit Format Run Options Window Help
# Program to add n natural numbers
# Input n number of natural no.from user
n = int(input("Enter n: "))
# initialize sum and counter
sum = 0
i = 1
while i <= n:
    sum = sum + i
    i = i+1
# update counter
# print the sum
print("Sum of",n, "natural numbers is: ", sum)
Ln: 12  Col: 38
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
    ========================= RESTART:
    Enter n: 5
    Sum of 5 natural numbers is:  15
>>>
    ========================= RESTART:
    Enter n: 10
    Sum of 10 natural numbers is:  55
>>>
```

असाइनमेंट

1. while लूप का उपयोग करके पहली 10 सम संख्याओं को प्रिंट करने के लिए एक प्रोग्राम लिखिए।
2. उपयोगकर्ता से प्राप्त किसी संख्या के अंकों का योग ज्ञात करने के लिए एक प्रोग्राम लिखिए।
3. while लूप का उपयोग करके उपयोगकर्ता द्वारा दी गई संख्या को रीवर्स करने का प्रोग्राम लिखिए।
4. यह जाँचने के लिए प्रोग्राम लिखिए कि दी गई संख्या पैलिंड्रोम है या नहीं।
5. यह जाँचने के लिए प्रोग्राम लिखिए कि दी गई संख्या आर्मस्ट्रांग है या नहीं।

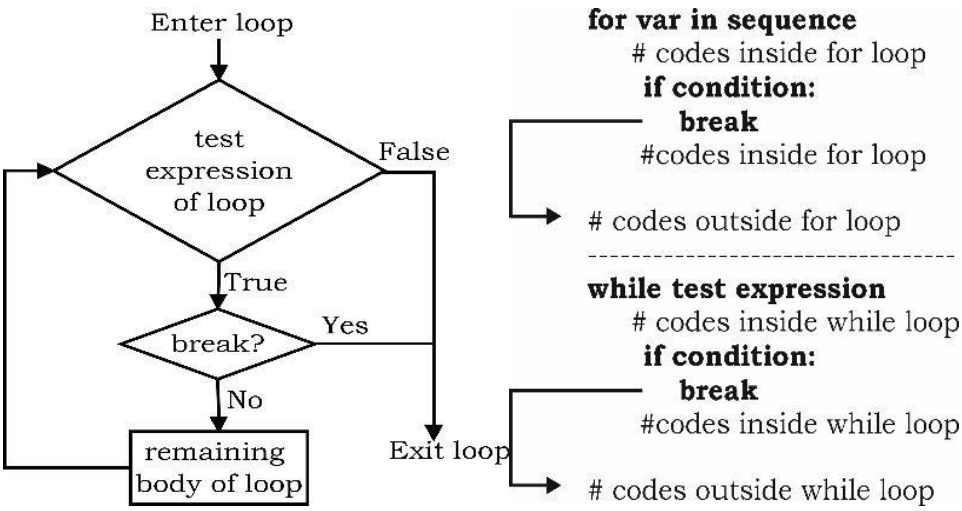
1.5 ब्रेक और कंटीन्यू स्टेटमेंट (Break and Continue Statement)

लूपिंग कंस्ट्रक्ट्स प्रोग्रामरों को कार्यों को कुशलतापूर्वक repetition की अनुमति देती हैं। कुछ स्थितियों में, जब कोई विशेष कंडीशन उत्पन्न होती है, तो हम लूप से बाहर निकलना चाहते हैं या लूप को जारी रखने से पहले उसके कुछ स्टेटमेंट्स को छोड़ना चाहते हैं।

इन आवश्यकताओं को क्रमशः break और continue स्टेटमेंट्स का उपयोग करके पूरा किया जा सकता है। पायथन प्रोग्राम के flow of execution को नियंत्रित करने में प्रोग्रामर को अधिक flexibility प्रदान करने के लिए ये स्टेटमेंट्स उपलब्ध कराता है।

1.5.1 ब्रेक स्टेटमेंट (Break Statement)

Break स्टेटमेंट current लूप को समाप्त कर देता है और उस लूप के बाद कंट्रोल करने वाले वाले स्टेटमेंट पर स्थानांतरित कर देता है, जहाँ से प्रोग्राम का निष्पादन पुनः शुरू होता है।



चित्र 11.6: लूप में ब्रेक स्टेटमेंट के उपयोग के लिए फ़्लोचार्ट

आइए, for लूप में break के उपयोग को दर्शाने के लिए एक प्रोग्राम लेते हैं।

```
# Program to demonstrate use use of break statement
```

```
num = 0
for num in range(10):
    num = num +1
    if num == 8:
        break
print('Number has value'+str(num))
print('Encountered break !! Out of loop')
```

```
File  Edit  Format  Run  Options  Window  Help
# Program to demonstrate use use of break statement
num = 0
for num in range(10):
    num = num +1
    if num == 8:
        break
print('Number has value'+str(num))
print('Encountered break !! Out of loop')
                                        Ln: 11  Col: 0
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
    ========================= RESTART
    Number has value8
    Encountered break !! Out of loop
>>> |
```

उपरोक्त प्रोग्राम में, जब num का मान 8 हो जाता है, तब break स्टेटमेंट निष्पादित होता है और for लूप समाप्त हो जाता है। अब एक अन्य उदाहरण पर विचार करें, जिसमें while लूप का उपयोग करके सभी धनात्मक (positive) संख्याओं का योग निकाला जाता है। जब उपयोगकर्ता द्वारा दर्ज की गई संख्या ऋणात्मक (negative) हो, तो आगे इनपुट लेना बंद कर दिया जाता है और योग प्रदर्शित किया जाता है।

```
# Find the sum of all positive numbers by taking the
# input using a while loop. Stop taking input when the
# number entered is negative using break statement
sum=0
print("Enter the number :")
while True:
    num = int(input()) # typecast string to integer
    if (num<0):
        break
    sum = sum + num
```

```
print("Sum = ",sum)
```

```
File  Edit  Format  Run  Options  Window  Help
# Find the sum of all positive numbers by taking the
# input using while loop. Stop taking input when the
# number entered is negative using break statement
sum=0
print("Enter the number :")
while True:
    num = int(input()) # typecast string to integer
    if (num<0):
        break
    sum = sum + num
print("Sum = ",sum)
                                                Ln: 10  Col: 0
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
    Enter the number :
    1
    2
    3
    4
    5
    6
    7
    8
    9
    0
    -1
    Sum =  45
>>>
```

यह जाँचने के लिए निम्नलिखित उदाहरण पर विचार करें कि दर्ज की गई संख्या अभाज्य (prime) है या नहीं।

```
# Program to check the input number is prime or not
num=int(input("Enter the number to check :"))
flag=0 # Assign flag as 0
if num>1:
    for i in range(2,int(num/2)):
        if (num % i == 0):
            flag = 1
        # number is not prime
    if flag==1:
        print(num,"is not a prime number")
    else:
        print(num,"is a prime number")
else:
```

© PSSCIVE Draft Stu

```
    print("Number entered is <=1, execute again!")
```

```
File  Edit  Format  Run  Options  Window  Help
# Program to check the input number is prime or not
num=int(input("Enter the number to check :"))
flag=0 # Assign flag as 0
if num>1:
    for i in range(2,int(num/2)):
        if (num % i == 0):
            flag = 1
        # number is not prime
    if flag==1:
        print(num,"is not a prime number")
    else:
        print(num,"is a prime number")
else:
    print("Number entered is <=1, execute again!")
                                                Ln: 5  Col: 0
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
      Enter the number to check :0
      Number entered is <=1, execute again!
>>>
      ========================== RESTART: /I
      Enter the number to check :1
      Number entered is <=1, execute again!
>>>
      ========================== RESTART: /I
      Enter the number to check :2
      2 is a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :3
      3 is a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :4
      4 is a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :5
      5 is a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :6
      6 is not a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :7
      7 is a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :8
      8 is not a prime number
>>>
      ========================== RESTART: /I
      Enter the number to check :9
      9 is not a prime number
```

असाइनमेंट

1. किसी list में किसी विशेष item का location ज्ञात करने के लिए एक प्रोग्राम लिखिए।

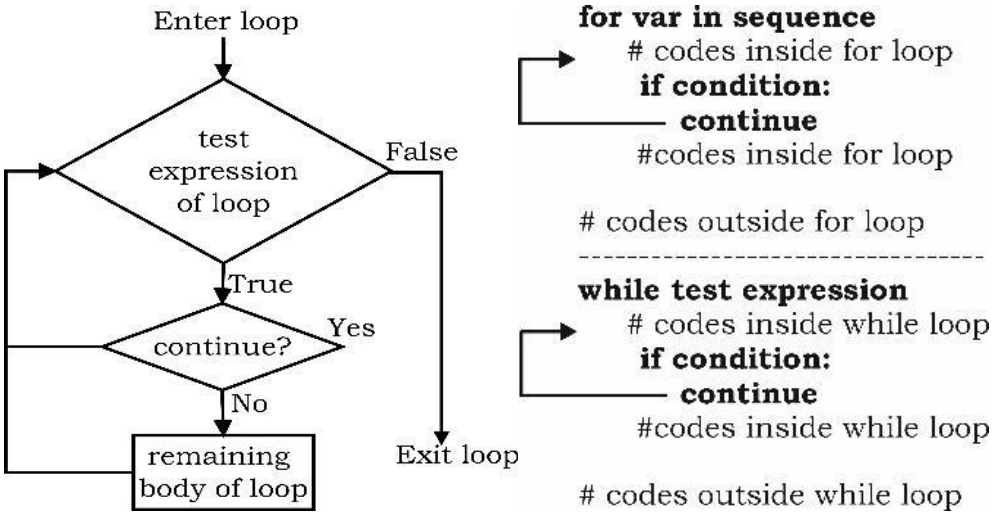
2. निम्नलिखित पायथन कोड का आउटपुट ज्ञात कीजिए:

```
for val in "string":
    if val == "i":
```

```
        break
  print(val)
print("The end")
```

1.5.2 कंटीन्यू स्टेटमेंट (Continue Statement)

जब continue स्टेटमेंट आता है, तब control वर्तमान इटरेशन के लिए लूप के अंदर बचे हुए स्टेटमेंट्स के निष्पादन को छोड़ देता है और अगली इटरेशन के लिए लूप की शुरुआत में चला जाता है। यदि लूप की शर्त (condition) अभी भी सत्य (true) है, तो लूप फिर से चलता है, अन्यथा नियंत्रण लूप के तुरंत बाद वाले स्टेटमेंट पर स्थानांतरित हो जाता है। चित्र 11.7 में continue स्टेटमेंट का फ्लोचार्ट दर्शाया गया है।



चित्र 11.7: continue स्टेटमेंट का फ़्लो चार्ट

उदाहरण: निम्नलिखित प्रोग्राम 0 से 9 तक सम और विषम संख्याओं को खोजने के लिए continue स्टेटमेंट के उपयोग को दर्शाता है।

```
# Demonstrate the use of continue statement
# Program to print ODD/EVEN numbers from 0-9
num = 0
print("ODD numbers from 0-9")
for num in range(9):
   num = num + 1
   if num%2 == 0:
      continue
   print("The next ODD number is :",num)
print('End of for loop')
print("EVEN numbers from 0-9")
```

© PSSCIVE Dr

```
for num in range(9):
    num = num + 1
    if num%2 != 0:
        continue
    print("The next EVEN number is:",num)
print('End of for loop')
```

```
File  Edit  Format  Run  Options  Window  Help
# Demonstrate the use of continue statement
# Program to print ODD/EVEN numbers from 0-9
num = 0
print("ODD numbers from 0-9")
for num in range(9):
    num = num + 1
    if num%2 == 0:
        continue
    print("The next ODD number is :",num)
print('End of for loop')
print("EVEN numbers from 0-9")
for num in range(9):
    num = num + 1
    if num%2 != 0:
        continue
    print("The next EVEN number is:",num)
print('End of for loop')
                                        Ln: 11  Col: 28
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
ODD numbers from 0-9
The next ODD number is : 1
The next ODD number is : 3
The next ODD number is : 5
The next ODD number is : 7
The next ODD number is : 9
End of for loop
EVEN numbers from 0-9
The next EVEN number is: 2
The next EVEN number is: 4
The next EVEN number is: 6
The next EVEN number is: 8
End of for loop
>>>
```

ध्यान दें कि आउटपुट में 3 का मान प्रिंट नहीं होता है, लेकिन continue स्टेटमेंट के बाद भी लूप चलता रहता है और for लूप समाप्त होने तक अन्य मानों को प्रिंट करता रहता है।

असाइनमेंट

1. 1 से 20 तक के मानों को प्रिंट करने के लिए एक प्रोग्राम लिखिए, जिसमें 3 के गुणज (multiples) शामिल न हों।
2. 1 से 50 तक सभी अभाज्य संख्याओं (prime numbers) को प्रिंट करने के लिए एक प्रोग्राम लिखिए।

1.6 नेस्टेड लूप्स (Nested Loops)

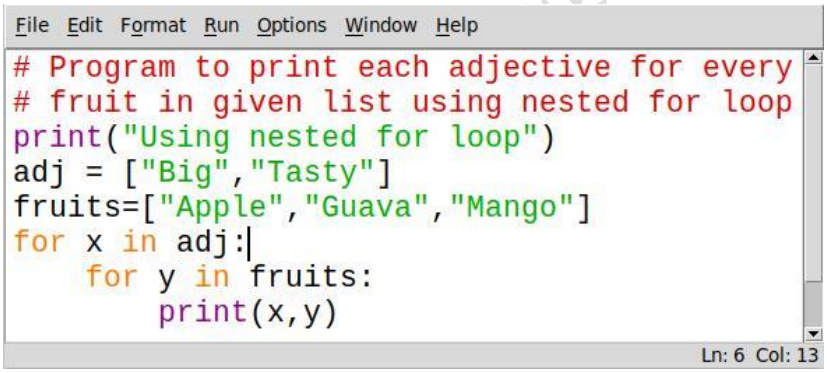
एक लूप के अंदर दूसरा लूप हो सकता है। एक लूप के अंदर स्थित दूसरे लूप को नेस्टेड लूप कहा जाता है। inner या outer लूप किसी भी प्रकार का हो सकता है, जैसे while लूप या for लूप। उदाहरण के लिए, बाहरी for लूप के अंदर while लूप हो सकता है और इसके विपरीत भी संभव है। बाहरी लूप में एक से अधिक आंतरिक लूप भी हो सकते हैं। लूप्स की इस श्रृंखला बनाने पर कोई सीमा नहीं होती।

नेस्टेड लूप्स का उपयोग प्रायः संख्याओं और स्टार पैटर्न प्रोग्राम्स में किया जाता है। ये बहु-आयामी (multidimensional) डेटा संरचनाओं के साथ कार्य करने में भी उपयोगी होते हैं, जैसे द्वि-आयामी एरे (two-dimensional arrays) को प्रिंट करना या ऐसी list पर इटरेट करना जिसमें एक नेस्टेड सूची शामिल हो।

उदाहरण: दी गई सूची में प्रत्येक फल के लिए प्रत्येक adjective प्रिंट करने के लिए नेस्टेड for लूप का उपयोग करते हुए एक प्रोग्राम लिखिए।

```
# Program to print each adjective for every
# fruit in given list using nested for loop
print("Using nested for loop")
adj = ["Big","Tasty"]
fruits=["Apple","Guava","Mango"]
for x in adj:
    for y in fruits:
        print(x,y)
```

dy Material Not be Publi



जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

उपयोगकर्ता द्वारा दिए गए पैटर्न को प्रिंट करने के लिए एक अन्य प्रोग्राम पर विचार कीजिए।

```
# Program to Generate the pattern
# of the number input by the user
# 1
# 1 2
# 1 2 3
# 1 2 3 4
# 1 2 3 4 5
num=int(input("Enter a number : "))
for i in range(1,num+1):
    for j in range(1,i+1):
        print(j,end=" ")
    print()
```

Not be Published

```
File  Edit  Format  Run  Options  Window  Help
# Program to Generate the pattern
# of the number input by the user
# 1
# 1 2
# 1 2 3
# 1 2 3 4
# 1 2 3 4 5
num=int(input("Enter a number : "))
for i in range(1,num+1):
    for j in range(1,i+1):
        print(j,end=" ")
    print()
                                    Ln: 7  Col: 0
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window
      Enter a number : 5
      1
      1 2
      1 2 3
      1 2 3 4
      1 2 3 4 5
>>>
```

1 से 10 के बीच अभाज्य संख्याओं को ज्ञात करने के लिए नेस्टेड लूप का उपयोग करने वाले एक अन्य प्रोग्राम पर विचार कीजिए।

```
# Program to find prime numbers
# between 0-9 using nested loop
```

```
print("Generate prime numbers in 0-9")
num=1
for i in range(0,9):
   j=2
   while(j<=(i/2)):
      if(i%j==0): # factor found
         break   # out of loop
      j+=1
   if(j>i/j):      # no factor found
      print(i, "is a prime number")
print("Bye Bye!!")
```

be Published

```
File  Edit  Format  Run  Options  Window  Help
# Program to find prime numbers
# between 0-9 using nested loop
print("Generate prime numbers in 0-9")
num=1
for i in range(0,9):
    j=2
    while(j<=(i/2)):
        if(i%j==0): # factor found
            break    # out of loop
        j+=1
    if(j>i/j):       # no factor found
        print(i, "is a prime number")
print("Bye Bye!!")
Ln: 3  Col: 33
```

जब आप प्रोग्राम रन करते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
Generate prime numbers in 0-9
0 is a prime number
1 is a prime number
2 is a prime number
3 is a prime number
5 is a prime number
7 is a prime number
Bye Bye!!
>>>
```

किसी दिए गए संख्या का फैक्टोरियल निकालने के लिए एक अन्य प्रोग्राम पर विचार कीजिए।

```
# Program to find the factorial of a given number
num=int(input("Enter a number: "))
```

```
fact=1
# Check the number for negative, positive or zero
if num<0:
    print("No factorial for -ve no.")
elif num==0:
    print("The factorial of 0 is 1")
else:
    for i in range(1,num+1):
        fact=fact*i
    print("Factorial of",num, "is",fact)
```

Published

```
File  Edit  Format  Run  Options  Window  Help
# Program to find the factorial of a given number
num=int(input("Enter a number: "))
fact=1
# Check the number for negative, positive or zero
if num<0:
    print("No factorial for -ve no.")
elif num==0:
    print("The factorial of 0 is 1")
else:
    for i in range(1,num+1):
        fact=fact*i
    print("Factorial of",num, "is",fact)
                                                Ln: 4  Col: 33
```

जब आप प्रोग्राम रन रते हैं, तो आउटपुट होगा:

```
File  Edit  Shell  Debug  Options  Window  Help
      Enter a number: 0
      The factorial of 0 is 1
>>>
      ==========================
      Enter a number: 1
      Factorial of 1 is 1
>>>
      ==========================
      Enter a number: 2
      Factorial of 2 is 2
>>>
      ==========================
      Enter a number: 5
      Factorial of 5 is 120
>>>
      ==========================
      Enter a number: -5
      No factorial for -ve no.
>>>
```

असाइनमेंट

1. नेस्टेड लूप का उपयोग करते हुए एक आयत (rectangle) बनाने के लिए प्रोग्राम लिखिए जिसमें 6 पंक्तियों (rows) और प्रत्येक पंक्ति में 20 * के साथ हों।
2. निम्नलिखित पैटर्न को प्रिंट करने के लिए एक प्रोग्राम लिखिए:

a.	b.	c.	d.	e.
1 2 2 3 3 3 4 4 4 4 5 5 5 5 5	1 2 1 3 2 1 4 3 2 1 5 4 3 2 1	A A B A B C A B C D A B C D E	1 1 2 1 2 3 1 2 3 4 1 2 3 4 5	5 4 3 2 1 4 3 2 1 3 2 1 2 1 1

सारांश

इस सत्र में विद्यार्थियों ने सीखा कि सामान्यतः एक प्रोग्राम चरण-दर-चरण (sequential execution) चलता है। लेकिन वास्तविक जीवन की स्थितियों (जैसे हवाई अड्डे की प्रक्रियाएँ) में कभी-कभी निर्णय लेने या कार्यों को बार-बार दोहराने की आवश्यकता होती है। इसी प्रकार, पायथन में:

• Conditional Statements निर्णय लेने में सहायता करते हैं।

• लूप्स कार्यों को कई बार दोहराने में मदद करते हैं।

ये Control structures प्रोग्रामरों को flow of execution को नियंत्रित करने और जटिल समस्याओं को कुशलतापूर्वक हल करने में सक्षम बनाती हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. पायथन दो प्रकार के कंट्रोल स्ट्रक्चर सपोर्ट करता है? (अ) परम्यूटेशन और कॉम्बिनेशन (ब) मॉड्यूल और लाइब्रेरी (स) बिल्ट-इन और यूजर-डिफाइंड (द) सेलेक्शन और रिपीटिशन
2. "प्रोग्रामिंग में निर्णय लेने या चयन (selection) की अवधारणा को निम्नलिखित में से किसकी सहायता से लागू किया जाता है? (अ) if else स्टेटमेंट (ब) for लूप (स) while लूप (द) फंक्शन
3. पायथन में समान स्तर का इंडेंटेशन स्टेटमेंट्स को निम्न में से किसमें जोड़ता है? (अ) सिंगल ब्लॉक ऑफ़ कोड (ब) सिंगल लूप (स) सिंगल फंक्शन (द) सिंगल कंट्रोल स्ट्रक्चर
4. इंडेंटेशन के प्रत्येक स्तर के लिए किस कुंजी का उपयोग होता है? (अ) \t (ब) \n (स) @ (द) #
5. किसी प्रोग्राम में स्टेटमेंट्स के एक सेट को दोहराना किसकी मदद से संभव है? (अ) if-else (ब) लूप्स (स) डेटा टाइप्स (द) फंक्शन

6. जब किसी लूप से जुड़ी कन्डिशन false हो जाती है, तो लूप क्या करता है? (अ) समाप्त हो जाता है (ब) चलता रहता है (स) फेल हो जाता है (द) एरर उत्पन्न करता है
7. किस फंक्शन का उपयोग दिए गए स्टार्ट मान से स्टॉप मान तक (स्टॉप मान को छोड़कर), दिए गए स्टेप वैल्यू के अंतर के साथ, पूर्णांकों का अनुक्रम युक्त एक लिस्ट बनाने के लिए किया जाता है? (अ) range() (ब) random() (स) maths() (द) int()
8. range() फंक्शन के start और step पैरामीटर कैसे होते हैं? (अ) वैकल्पिक (ब) अनिवार्य (स) डिफ़ॉल्ट (द) इन्वेलिड
9. रेंज() ऑफ़ द फंक्शन के सभी पैरामीटर क्या होने चाहिए? (अ) इन्टिजर (ब) फ्लोट (स) लिस्ट (द) टपल
10. स्टेप पैरामीटर ऑफ़ रेंज() फंक्शन एक धनात्मक या ऋणात्मक पूर्णांक हो सकता है, लेकिन यह क्या नहीं हो सकता? (अ) 0 (ब) 1 (स) -1 (द) -2
11. फंक्शन range(x) संख्याओं का यह क्रम [0,1,2,3,4,5,6,7,8,9,10] उत्पन्न करेगा। तो x का मान क्या होगा? (अ) 10 (ब) 11 (स) -10 (द) -11
12. निम्नलिखित में से किसे पायथन में फॉर लूप के रूप में उपयोग नहीं किया जाता है? (अ) for लूप (ब) while लूप (स) do-while लूप (द) इनमें से कोई नहीं
13. पायथन प्रोग्राम में, एक कंट्रोल स्ट्रक्चर है? (अ) प्रोग्राम-विशिष्ट डेटा स्ट्रक्चर को परिभाषित करता है। (ब) प्रोग्राम में स्टेटमेंट्स के निष्पादन के क्रम को निर्देशित करता है। (स) यह बताता है कि प्रोग्राम शुरू होने से पहले और उसके समाप्त होने के बाद क्या होगा। (द) उपरोक्त में से कोई नहीं।
14. निम्नलिखित लूप कितनी बार रन करेगा? (अ) 2 (ब) 3 (स) 1 (द) 0

```
i=2
while(i>0):
i=i-1
```

15. निम्नलिखित कोड का आउटपुट क्या होगा? (अ) 12 (ब) 1, 2 (स) 0 (द) Error

```
x = 12
for i in x:
print(i)
```

ब. रिक्त स्थान भरें

1. किसी प्रोग्राम में स्टेटमेंट्स के निष्पादन के क्रम को _____ कहा जाता है।
2. पायथन ब्लॉक और नेस्टेड ब्लॉक संरचनाओं के लिए _____ का उपयोग करता है।
3. इंटरप्रेटर इंडेंटेशन के स्तरों की बहुत सख्ती से जाँच करता है और यदि इंडेंटेशन सही नहीं है तो _____ एरर देता है।
4. _____ क्लॉज if के साथ-साथ लूप के साथ भी आ सकता है।

5. break स्टेटमेंट वर्तमान लूप को _____ कर देता है और उस लूप के बाद वाले स्टेटमेंट के निष्पादन को फिर से शुरू कर देता है।
6. जब continue स्टेटमेंट का सामना होता है, तो नियंत्रण वर्तमान इटरेशन के लिए लूप के मुख्य भाग के अंदर शेष स्टेटमेंट्स के निष्पादन को _____ देता है और अगले इटरेशन के लिए लूप की शुरुआत में कूद जाता है।
7. एक लूप के अंदर दूसरे लूप को _____ लूप कहा जाता है।
8. नेस्टेड लूप का सामान्य उपयोग विभिन्न _____ और _____ पैटर्न प्रिंट करना है।
9. _____ डेटा संरचनाओं के साथ काम करने के लिए, जैसे द्वि-आयामी सरणियाँ प्रिंट करना, आमतौर पर नेस्टेड लूप का उपयोग किया जाता है।
10. यह जाँचने के लिए स्टेटमेंट कि a, b के बराबर है या नहीं: if _____ है।

स. सत्य या असत्य बताएं

1. if स्टेटमेंट के तहत एक ब्लॉक में जितनी चाहें उतने स्टेटमेंट आ सकते हैं, इसकी कोई सीमा नहीं है।
2. लूप में स्टेटमेंट्स बार-बार तब तक चलते हैं जब तक कि कोई खास लॉजिकल शर्त असत्य (false) बनी रहती है।
3. range() पायथन में एक बिल्ट-इन फंक्शन है।
4. while स्टेटमेंट कोड के एक ब्लॉक को बार-बार तब तक चलाता है जब तक लूप की कंट्रोल शर्त सत्य (true) रहती है।
5. लूपों को एक के अंदर एक (नेस्टेड) करने की कोई सीमा नहीं है।
6. "break" कीवर्ड का इस्तेमाल करंट लूप से बाहर निकलने के लिए किया जा सकता है।
7. यदि कोई शर्त कभी असत्य (false) नहीं होती है, तो लूप अनंत लूप (infinite loop) बन जाता है।
8. यदि शर्त सही (true) है तो if ब्लॉक के स्टेटमेंट्स चलेंगे, अन्यथा else ब्लॉक के स्टेटमेंट्स चलेंगे।
9. पायथन में Do-while एक मान्य लूप है।
10. break और continue जंप स्टेटमेंट हैं।

द. लघु उत्तरीय प्रश्न

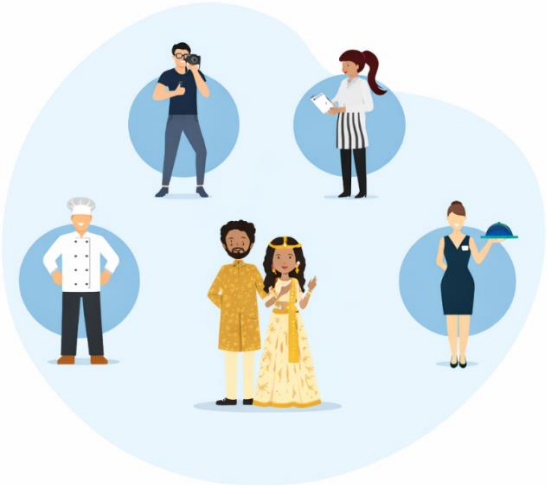
1. एक प्रोग्राम लिखिए जो उपयोगकर्ता से नाम और उम्र इनपुट लेकर यह संदेश दिखाए कि वह ड्राइविंग लाइसेंस के लिए आवेदन करने योग्य है या नहीं। पात्रता आयु 18 वर्ष है।
2. उपयोगकर्ता द्वारा दर्ज किए गए किसी संख्या का पहाड़ा (table) प्रिंट करने के लिए एक प्रोग्राम लिखिए।
3. एक प्रोग्राम लिखिए जो उपयोगकर्ता द्वारा दर्ज की गई पाँच संख्याओं में से न्यूनतम (minimum) और अधिकतम (maximum) संख्या प्रिंट करता है।

4. यह जाँच करने के लिए एक प्रोग्राम लिखिए कि उपयोगकर्ता द्वारा दर्ज किया गया वर्ष (year) लीप वर्ष (leap year) है या नहीं।
5. एक प्रोग्राम लिखिए जो निम्नलिखित अनुक्रम (sequence) उत्पन्न करे: –5, 10, –15, 20, –25… n पदों तक, जहाँ n उपयोगकर्ता द्वारा इनपुट किया गया एक पूर्णांक है।
6. 1 + 1/8 + 1/27 + ... + 1/n का योग ज्ञात करने के लिए एक प्रोग्राम लिखिए, जहाँ n उपयोगकर्ता द्वारा दर्ज किया गया है।
7. उपयोगकर्ता द्वारा इनपुट किए गए एक पूर्णांक संख्या के अंकों का योग ज्ञात करने के लिए एक प्रोग्राम लिखिए।
8. एक फंक्शन लिखिए जो जाँच करता है कि इनपुट संख्या पैलिंड्रोम (palindrome) है या नहीं। (पैलिंड्रोम वह संख्या होती है जो आगे और पीछे से पढ़ने पर एक जैसी लगती है, जैसे 121, 555)।
9. कीबोर्ड के माध्यम से दर्ज की गई संख्याओं की एक सूची (list) में से सबसे बड़ी संख्या ज्ञात करने के लिए एक प्रोग्राम लिखिए।
10. N संख्याएँ इनपुट करने और फिर उनमें से दूसरी सबसे बड़ी (second largest) संख्या प्रिंट करने के लिए एक प्रोग्राम लिखिए।

© PSSCIVE Draft Study Material

सत्र 2. पायथन में फ़ंक्शन (Functions in Python)

एक विवाह समारोह में केटरिंग, डेकोरेशन और अन्य कई कार्य करने होते हैं। सामान्यतः हम इन अलग-अलग कार्यों को विशेष समूहों या सेवा प्रदाताओं को सौंप देते हैं, ताकि काम सुचारु रूप से पूरा हो सके। उदाहरण के लिए, खान-पान का कार्य कैटरर्स को और सजावट का कार्य डेकोरेटर्स को दिया जाता है। इससे पूरे आयोजन को प्रबंधित करना आसान हो जाता है। इसी प्रकार, प्रोग्रामिंग में भी हम किसी विशेष कार्य को करने के लिए फंक्शन्स का उपयोग करते हैं, जिससे हमारा प्रोग्राम मॉड्यूलर और पढ़ने में आसान बनता है।



चित्र 2.1: विवाह समारोह में विभिन्न कार्य

पायथन फ़ंक्शन (कार्य)

फंक्शन कोड का एक ऐसा ब्लॉक होता है जो किसी विशेष कार्य को करता है। मान लीजिए हमें एक ऐसा प्रोग्राम बनाना है जो एक वृत्त बनाए और उसे रंगे। इस समस्या को हल करने के लिए हम दो फंक्शन बना सकते हैं:

1. वृत्त बनाने के लिए एक फंक्शन
2. आकृति को रंगने के लिए एक फंक्शन

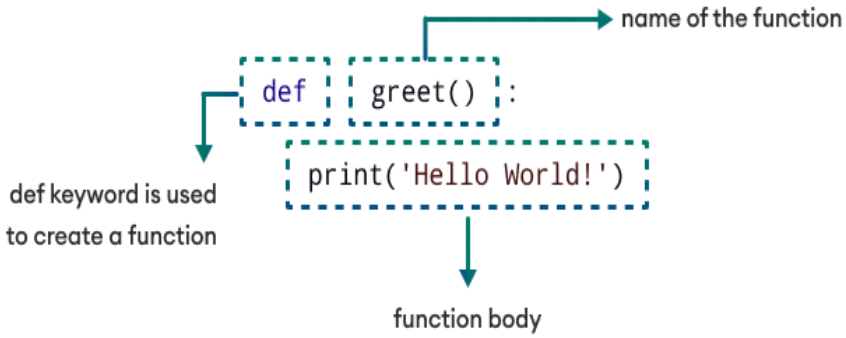
किसी जटिल समस्या को छोटे-छोटे भागों में विभाजित करने से हमारा प्रोग्राम समझने में आसान हो जाता है और उसका पुनः उपयोग भी किया जा सकता है।

फ़ंक्शन बनाना

आइए दूसरों का अभिवादन करने के लिए एक फ़ंक्शन greet() बनाएं।

```
def greet():
    print('Hello World!')
```

इस फ़ंक्शन के विभिन्न भाग इस प्रकार हैं:



यहाँ, Hello World! प्रिंट करने के लिए greet() नामक एक साधारण फ़ंक्शन बनाया गया है।

नोट: फ़ंक्शन लिखते समय, इंडेंटेशन को समझना महत्वपूर्ण है। इंडेंटेशन कोड की एक पंक्ति की शुरुआत में रिक्त स्थान होते हैं।

उपरोक्त कोड में, print() स्टेटमेंट को यह दिखाने के लिए इंडेंट किया गया है कि यह फ़ंक्शन के मुख्य भाग का हिस्सा है, जो फ़ंक्शन की परिभाषा को उसके मुख्य भाग से अलग करता है।

फ़ंक्शन को कॉल करना (Calling a Function)

उपरोक्त उदाहरण में 'greet()' नाम से एक फंक्शन declare किया गया है।

```
def greet():
    print('Hello World!')
```

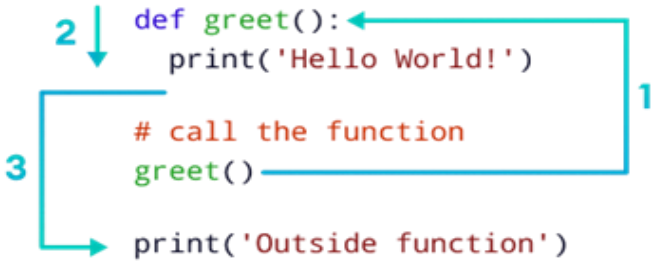
उपरोक्त कोड को रन करने पर कोई आउटपुट प्राप्त नहीं होगा। फंक्शन उसके अंदर लिखे गए कोड को निष्पादित करने के लिए बनाया जाता है। इसके निष्पादन के लिए फंक्शन को कॉल करना आवश्यक होता है। फंक्शन को कॉल करने के लिए केवल उसका नाम खोलने और बंद करने वाले कोष्ठक (parentheses) के साथ लिखना होता है, जैसा कि नीचे दिखाया गया है।

```
greet()
```

आइए निम्नलिखित कोड के साथ प्रदर्शित करें कि फ़ंक्शन को कैसे कॉल किया जाए:

```
def greet():
    print('Hello World!')
# call the function
greet()
print('Outside function')
```

उपरोक्त उदाहरण में, greet() नामक एक फ़ंक्शन बनाया गया है। नियंत्रण के प्रवाह को नीचे दर्शाया गया है:



जब 'greet()' फंक्शन को कॉल किया जाता है, तो प्रोग्राम का नियंत्रण फंक्शन की परिभाषा पर चला जाता है। फंक्शन के अंदर का पूरा कोड निष्पादित हो जाता है। इसके बाद प्रोग्राम का नियंत्रण फंक्शन कॉल के बाद वाले अगले स्टेटमेंट पर पहुँच जाता है।

पायथन फ़ंक्शन आर्गुमेंट (Python Function Arguments)

आर्गुमेंट फ़ंक्शन को दिए गए इनपुट होते हैं। आर्गुमेंट के मान को फ़ंक्शन में पास किया जा सकता है या स्वयं उल्लेख किया जा सकता है। प्रत्येक कॉल में अलग-अलग तर्क पास करना संभव है, जिससे फ़ंक्शन पुन: प्रयोज्य और गतिशील बन जाता है।

निम्नलिखित उदाहरण में, मान 'Diya' को greet() फ़ंक्शन के तर्क में पास किया गया है ताकि 'Hello Diya' के रूप में आउटपुट प्रदर्शित किया जा सके।

इस फ़ंक्शन को फिर से 'Deepak' नामक दूसरे तर्क के साथ कॉल किया जाता है, जैसा कि नीचे दिए गए कोड में दिखाया गया है।

```
def greet(name):
    print("Hello", name)
# pass argument
greet("Diya")
greet("Deepak")
```

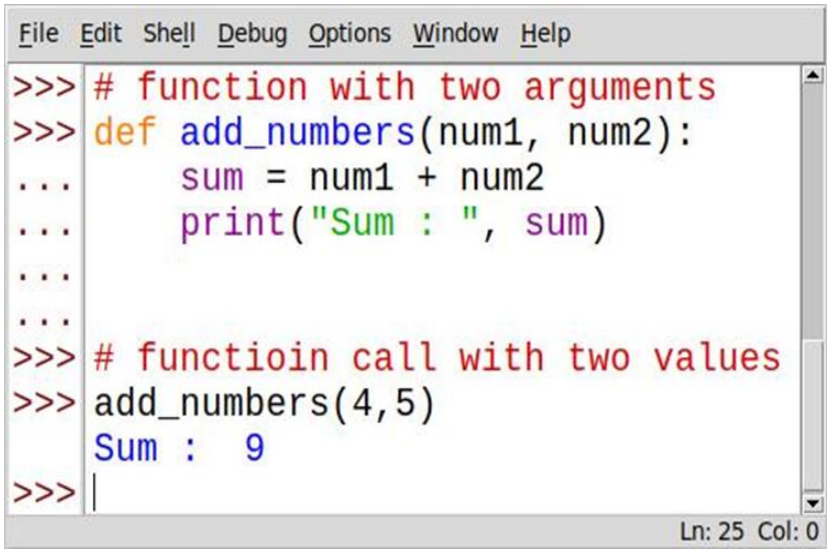
[यहाँ आउटपुट दिया गया है]

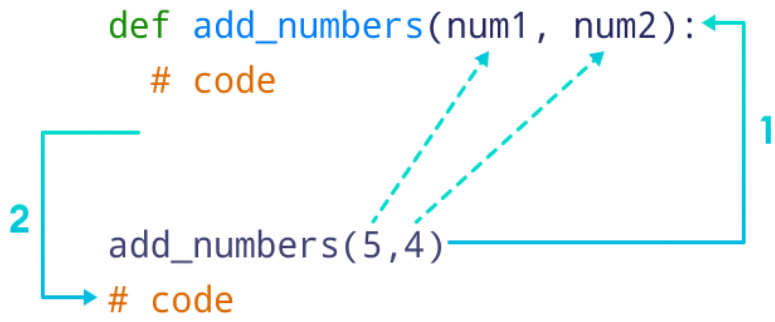
```
File  Edit  Shell  Debug  Options  Window  Help
>>> def greet(name):
...     print('Hello',name)
...
...
>>> # pass argument
>>> greet('Diya')
Hello Diya
>>> greet('Deepak')
Hello Deepak
>>>
                                    Ln: 9  Col: 0
```

दो संख्याओं को जोड़ने के लिए फ़ंक्शन का एक और उदाहरण देखिएं।

```
# function with two arguments
def add_numbers(num1, num2):
    sum = num1 + num2
    print("Sum : ", sum)

# functioin call with two values
add_numbers(4,5)
```

उपरोक्त उदाहरण में, add_numbers() नामक एक फ़ंक्शन तर्कों के साथ बनाया गया है: num1 और num2।





पैरामीटर और आर्गुमेंट (Parameters and Arguments)

पैरामीटर

पैरामीटर फ़ंक्शन परिभाषा में कोष्ठक के अंदर सूचीबद्ध चर होते हैं। वे उस डेटा के लिए प्लेसहोल्डर की तरह कार्य करते हैं जिसे फ़ंक्शन कॉल करने पर स्वीकार कर सकता है।

पैरामीटर को उस खाके (blueprint) के रूप में सोचें जो रूपरेखा तैयार करता है कि फ़ंक्शन किस प्रकार की जानकारी प्राप्त करने की अपेक्षा करता है।

```
def print_age(age):  # age is a parameter
    print(age)
```

इस उदाहरण में, print_age() फ़ंक्शन अपने इनपुट के रूप में age लेता है। हालाँकि, इस स्तर पर, वास्तविक मान निर्दिष्ट नहीं है। age पैरामीटर केवल एक विशिष्ट मान की प्रतीक्षा कर रहा एक प्लेसहोल्डर है जिसे फ़ंक्शन कॉल करने पर प्रदान किया जाना है।

आर्गुमेंट्स

आर्गुमेंट्स वास्तविक मान होते हैं जो फ़ंक्शन को कॉल करने पर पास किए जाते हैं। जब फ़ंक्शन निष्पादित होता है तो आर्गुमेंट्स पैरामीटर को प्रतिस्थापित कर देते हैं।

```
print_age(21)  # 21 is an argument
```

यहाँ, फ़ंक्शन कॉल के दौरान, तर्क 21 फ़ंक्शन को पास किया जाता है।

द रिटर्न स्टेटमेंट

किसी फ़ंक्शन का मान return स्टेटमेंट का उपयोग करके लौटाया जाता है।

नीचे दिए गए उदाहरण पर विचार कीजिए।

```
# function definition
def find_square(num):
    result = num * num
    return result

# function call
square = find_square(3)
print('Square:', square)
square = find_square(5)
print('Square:', square)
```

उपरोक्त उदाहरण में, find_square() नामक एक फ़ंक्शन बनाया गया है। फ़ंक्शन एक संख्या स्वीकार करता है और संख्या का वर्ग लौटाता है।

नोट: return स्टेटमेंट फ़ंक्शन के अंत को भी दर्शाता है। return के बाद कोई भी कोड निष्पादित नहीं होता है।

द पास स्टेटमेंट

pass स्टेटमेंट भविष्य के कोड के लिए एक प्लेसहोल्डर के रूप में कार्य करता है, खाली कोड ब्लॉकों के कारण होने वाली त्रुटियों को रोकता है। इसका उपयोग आमतौर पर वहाँ किया जाता है जहाँ कोड की योजना बनाई गई है लेकिन अभी तक लिखा जाना बाकी है।

© PSSCIVE Draft Study Material not to be republished

```
def future_function():
    pass

# this will execute without any action or error
future_function()
```

पायथन लाइब्रेरी फ़ंक्शन

पायथन कुछ बिल्ट-इन फ़ंक्शन प्रदान करता है जिनका उपयोग सीधे पायथन प्रोग्राम में किया जा सकता है। ऐसे मामलों में हमें फ़ंक्शन बनाने की आवश्यकता नहीं होती है, बस उन्हें कॉल करने की आवश्यकता होती है।

कुछ पायथन लाइब्रेरी फ़ंक्शन हैं:

```
print() - prints the string inside the quotation marks
sqrt() - returns the square root of a number
pow() - returns the power of a number
```

ये लाइब्रेरी फ़ंक्शन मॉड्यूल के अंदर परिभाषित होते हैं, और उनका उपयोग करने के लिए, आपको अपने प्रोग्राम के अंदर math मॉड्यूल को शामिल करना होगा। उदाहरण के लिए, sqrt() math मॉड्यूल के अंदर परिभाषित है।

उदाहरण: पायथन लाइब्रेरी फ़ंक्शन (Python Library Function)

उदाहरण: पायथन लाइब्रेरी फ़ंक्शन

निम्नलिखित उदाहरण से पायथन लाइब्रेरी फ़ंक्शन के उपयोग को समझते है।

```
import math

# sqrt computes the square root
square_root = math.sqrt(4)

print("Square Root of 4 is",square_root)

# pow() comptes the power
power = pow(2, 3)

print("2 to the power 3 is",power)
[यहाँ आउटपुट दिया गया है]
```

© PSSCIVE Draft Not be Pu

```
File Edit Shell Debug Options Window Help
>>> import math
>>> # sqrt computes the square root
>>> square_root = math.sqrt(4)
>>> print("Square Root of 4 is",square_root)
    Square Root of 4 is 2.0
>>> # pow() comptes the power
>>> power = pow(2, 3)
>>> print("2 to the power 3 is",power)
    2 to the power 3 is 8
>>>
                                    Ln: 65 Col: 21
```

पायथन फ़ंक्शन्स पर अधिक जानकारी

यूजर डिफाइंड फ़ंक्शन बनाम स्टैण्डर्ड लाइब्रेरी फ़ंक्शंस

पायथन में, फ़ंक्शन दो श्रेणियों में विभाजित हैं: उपयोगकर्ता-परिभाषित फ़ंक्शन (उपयोगकर्ता-परिभाषित फ़ंक्शन) और मानक लाइब्रेरी फ़ंक्शन (मानक लाइब्रेरी फ़ंक्शन)। ये दोनों कई तरह से भिन्न हैं:

यूजर डिफाइंड फ़ंक्शन

ये उपयोगकर्ता द्वारा बनाए गए फ़ंक्शन होते हैं। वे हमारी आवश्यकता के अनुसार किए जाने वाले विशिष्ट कार्यों के लिए डिज़ाइन किए गए हैं। वे पायथन के मानक टूलबॉक्स का हिस्सा नहीं हैं। उपयोगकर्ता के पास उन्हें अपनी आवश्यकताओं के अनुरूप बनाने की स्वतंत्रता है, जिससे कोड में एक व्यक्तिगत स्पर्श जुड़ जाता है।

स्टैण्डर्ड लाइब्रेरी फ़ंक्शंस

ये पायथन के पहले से तैयार (pre-packaged) उपहार हैं। ये पायथन में पहले से ही अंतर्निहित (built-in) होते हैं और उपयोग के लिए तैयार रहते हैं। ये फंक्शन्स गणित, फाइल संचालन, स्ट्रिंग्स के साथ कार्य करना आदि जैसे अनेक सामान्य कार्यों को कवर करते हैं। इन्हें पायथन समुदाय द्वारा परीक्षण किया गया है, जिससे इनकी कार्यकुशलता (efficiency) और विश्वसनीयता (reliability) सुनिश्चित होती है।

फ़ंक्शन में डिफ़ॉल्ट तर्क

पायथन फ़ंक्शन को डिफ़ॉल्ट तर्क मान रखने की अनुमति देता है। डिफ़ॉल्ट तर्कों का उपयोग तब किया जाता है जब फ़ंक्शन कॉल के दौरान इन पैरामीटरों के लिए कोई स्पष्ट मान पास नहीं किया जाता है।

आइए एक उदाहरण देखिएं।

```
def greet(name, message="Hello"):
    print(message, name)

# calling function with both arguments
greet("Students", "Good Morning")

# calling function with only one argument
greet("Diya")
```

[यहाँ आउटपुट दिया गया है]

```
File  Edit  Shell  Debug  Options  Window  Help
>>> def greet(name, message="Hello"):
...     print(message, name)
... # calling function with both arguments
...
>>> # calling function with both arguments
>>> greet("Students", "Good Morning")
    Good Morning Students
>>> # calling function with only one argument
>>> greet("Diya")
    Hello Diya
>>>
                                        Ln: 79  Col: 0
```

दो संख्याओं a और b में से अधिकतम ज्ञात करने की समस्या पर विचार करें। आइए इस समस्या के लिए एक सरल पायथन प्रोग्राम लेते हैं।

```
# Find Maximum of two numbers
# Without using function
a=10
b=20
if a>b:
    max=a
else:
    max=b
    print("Maximum = ",max)
```

```
File  Edit  Shell  Debug  Options  Window  Help
>>> # Find Maximum of two numbers
>>> # Without using function
>>> a=10
>>> b=20
>>> if a>b:
...     max=a
... else:
...     max=b
...     print("Maximum = ",max)
...
...
    Maximum =  20
>>>
                                        Ln: 11  Col: 0
```

उपरोक्त प्रोग्राम में मान (value) को प्रिंट करने के लिए एक बिल्ट-इन फंक्शन print() का उपयोग किया गया है। इस समस्या को हल करने का एक अन्य तरीका यह है कि प्रोग्राम को अलग-अलग कोड ब्लॉक्स में विभाजित किया जाए और प्रत्येक ब्लॉक को एक ऐसे फंक्शन में रखा जाए जो किसी विशेष कार्य को करने के लिए उपयोग किया जाता है।

किसी कंप्यूटर प्रोग्राम को अलग-अलग स्वतंत्र कोड ब्लॉक्स या विभिन्न नामों और विशिष्ट कार्यों वाले उप-समस्याओं (sub-problems) में विभाजित करने की प्रक्रिया को मॉड्यूलर प्रोग्रामिंग (modular programming) कहा जाता है।

ft Study Material Not be Pu

इस सत्र में आप फंक्शन्स की अवधारणा और उनके उपयोग के लाभों को समझेंगे। हम पायथन प्रोग्रामिंग में उपयोगकर्ता-परिभाषित फ़ंक्शन (यूज़र-डिफाइंड फंक्शन्स), निष्पादन का प्रवाह (flow of execution), चर (वेरिएबल) का दायरा (scope) और मानक लाइब्रेरी (standard libraries) पर चर्चा करेंगे।

1.2 फ़ंक्शन (Functions)

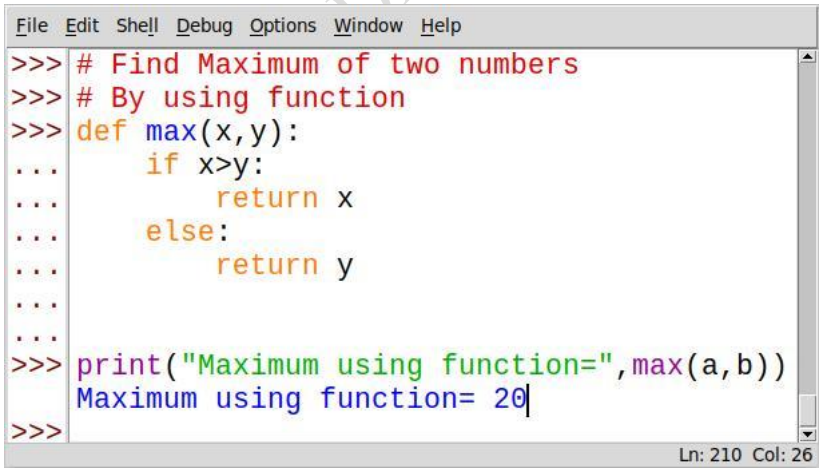
प्रोग्रामिंग में फंक्शन का उपयोग मॉड्यूलरिटी (modularity) और पुनः उपयोगिता (reusability) प्राप्त करने के एक महत्वपूर्ण साधन के रूप में किया जाता है। फंक्शन को निर्देशों (instructions) के एक नामित समूह के रूप में परिभाषित किया जा सकता है, जो कॉल (invoke) किए जाने पर एक विशिष्ट कार्य को पूरा करता है।

एक बार फंक्शन परिभाषित हो जाने के बाद, उसे प्रोग्राम के विभिन्न स्थानों से बार-बार कॉल किया जा सकता है, बिना हर बार उसका पूरा कोड लिखे। इसे किसी अन्य फंक्शन के अंदर से भी केवल फंक्शन का नाम लिखकर और आवश्यक पैरामीटर्स (यदि हों) पास करके कॉल किया जा सकता है। प्रोग्रामर कोड लिखते समय अपनी आवश्यकता के अनुसार जितने चाहें उतने फंक्शन्स परिभाषित कर सकता है।

उपरोक्त प्रोग्राम को यूज़र-डिफाइंड फंक्शन्स का उपयोग करते हुए नीचे दर्शाए अनुसार पुनः लिखा गया है।

```
# Find Maximum of two numbers
# By using function
def max(x,y):
    if x>y:
        return x
    else:
        return y
print("Maximum using function=",max(a,b))
```

Draft Study Material Not



ऊपर दिए गए कोड की तुलना करने पर, जिसमें अधिकतम मान (maximum) को फंक्शन का उपयोग करके और बिना फंक्शन के निकाला गया है, यह देखा जा सकता है कि फंक्शन का उपयोग करने वाला प्रोग्राम अधिक व्यवस्थित (organized) और पढ़ने में आसान होता है।

दो संख्याओं में से अधिकतम मान ज्ञात करने के लिए 'max()' नामक एक यूज़र-डिफाइंड फंक्शन का उपयोग किया जाता है। 'print()' फंक्शन पायथन का एक बिल्ट-इन फंक्शन है, जिसका उपयोग मान (value) को कंसोल पर प्रदर्शित करने के लिए किया जाता है।

इस प्रकार, पायथन में दो प्रकार के फंक्शन्स होते हैं— यूज़र-डिफाइंड फंक्शन्स और बिल्ट-इन फंक्शन्स।

फ़ंक्शन के लाभ

किसी प्रोग्राम में फ़ंक्शन का उपयोग करने के निम्नलिखित लाभ हैं:

- पठनीयता बढ़ाता है (Increases readability), विशेष रूप से लंबे कोड के लिए क्योंकि फ़ंक्शन का उपयोग करके प्रोग्राम बेहतर ढंग से व्यवस्थित और समझने में आसान होता है।
- कोड की लंबाई कम करता है (Reduces code length) क्योंकि एक ही कोड को किसी प्रोग्राम में कई स्थानों पर लिखने की आवश्यकता नहीं होती है। यह डिबगिंग को भी आसान बनाता है।
- पुनः उपयोगिता बढ़ाता है (Increases reusability), क्योंकि किसी फ़ंक्शन को किसी अन्य फ़ंक्शन या किसी अन्य प्रोग्राम से कॉल किया जा सकता है। इस प्रकार, हम पहले से परिभाषित फ़ंक्शनों का पुन: उपयोग या उन पर निर्माण कर सकते हैं और समान कोड को बार-बार लिखने से बचा जा सकता हैं।
- कार्य को टीम के सदस्यों के बीच आसानी से विभाजित कर इसे समानांतर (parallel) रूप से पूरा क्र सकते है।

2.3 यूज़र-डिफाइंड फंक्शन्स

फ़ंक्शन की पुन: उपयोगिता सुविधा का लाभ उठाते हुए, पायथन में मानक पुस्तकालय के तहत बड़ी संख्या में फ़ंक्शन पहले से ही उपलब्ध हैं। हम इन फ़ंक्शनों को परिभाषित किए बिना सीधे अपने प्रोग्राम में कॉल कर सकते हैं। हालाँकि, मानक लाइब्रेरी फ़ंक्शन के अलावा, हम प्रोग्राम लिखते समय अपने स्वयं के फ़ंक्शन परिभाषित कर सकते हैं। ऐसे फ़ंक्शन को उपयोगकर्ता-परिभाषित फ़ंक्शन कहा जाता है। इस प्रकार, प्रोग्रामर की आवश्यकता के अनुसार कुछ कार्यों को प्राप्त करने के लिए परिभाषित एक फ़ंक्शन को उपयोगकर्ता-परिभाषित फ़ंक्शन कहा जाता है।

2.3.1 उपयोगकर्ता-परिभाषित फ़ंक्शन बनाना (Creating User Defined Function)

एक फ़ंक्शन परिभाषा def (define के लिए संक्षिप्त) से शुरू होती है। उपयोगकर्ता-परिभाषित फ़ंक्शन बनाने का सिंटैक्स चित्र 2.2 में दिखाया गया है।

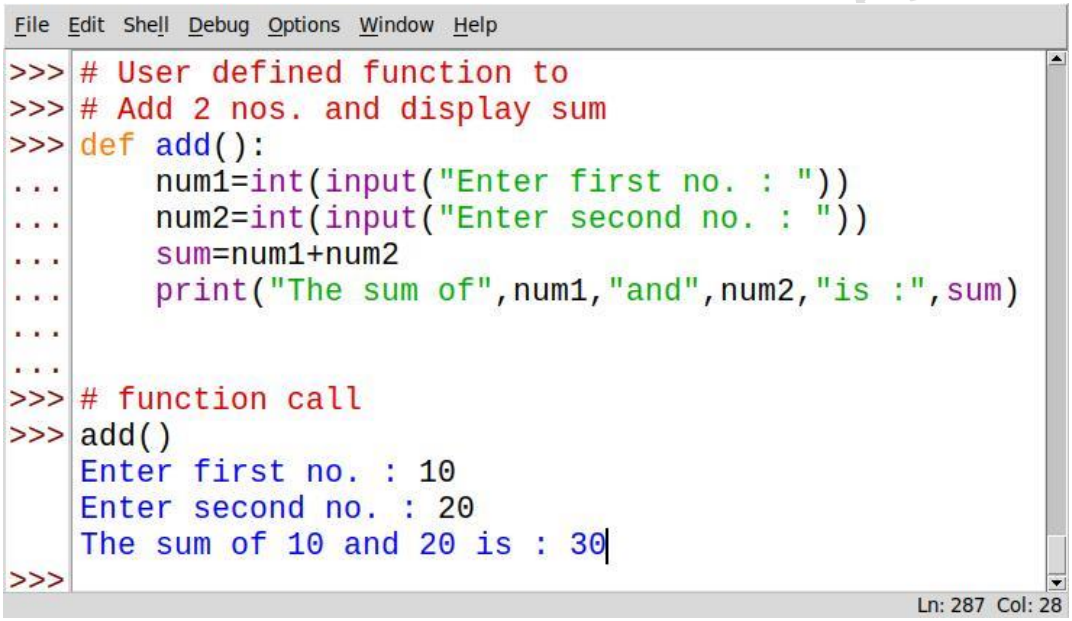
```
def<Function  name>  ([parameter 1, parameter 2,....]) :   [ Function Header ]

        set of instruction to be executed
        [return <value>]                  }  [ Function Body ( Should be intented
                                                within the function header) ]
```

चित्र 2.2: उपयोगकर्ता-परिभाषित फ़ंक्शन का सिंटैक्स

"[ ]" में संलग्न आइटम पैरामीटर कहलाते हैं और वे वैकल्पिक होते हैं। इसलिए, किसी फ़ंक्शन में पैरामीटर हो भी सकते हैं और नहीं भी। साथ ही, एक फ़ंक्शन मान लौटा भी सकता है और नहीं भी। फ़ंक्शन हेडर हमेशा एक कोलन (:) के साथ समाप्त होता है। फ़ंक्शन नाम अद्वितीय होने चाहिए। पहचानकर्ताओं के नामकरण के नियम फ़ंक्शन नामकरण पर भी लागू होते हैं। फ़ंक्शन इंडेंटेशन के बाहर के स्टेटमेंट्स को फ़ंक्शन का हिस्सा नहीं माना जाता है।

निम्नलिखित प्रोग्राम दर्शाता है कि दो संख्याओं को जोड़ने और उनके योग को परिणाम के रूप में प्रदर्शित करने के लिए उपयोगकर्ता-परिभाषित फ़ंक्शन कैसे लिखा जाता है।

```
# User defined function to add 2 nos. and display sum
def add():
    num1=int(input("Enter first no. : "))
    num2=int(input("Enter second no. : "))
    sum=num1+num2
    print("The sum of",num1,"and",num2,"is :",sum)

# function call
add()
```



उपरोक्त प्रोग्राम में, add() एक उपयोगकर्ता-परिभाषित फ़ंक्शन है जो इनपुट के रूप में दो पूर्णांक संख्याएँ लेता है, दो संख्याओं के योग की गणना करता है और उसे प्रदर्शित करता है। इस फ़ंक्शन को निष्पादित करने के लिए, इस फ़ंक्शन को फ़ंक्शन नाम () का उपयोग करके कॉल करना आवश्यक है। जैसा कि दिखाया गया है, फ़ंक्शन add को कोड की अंतिम पंक्ति में कॉल किया जा रहा है।

def कीवर्ड का उपयोग फ़ंक्शन को परिभाषित करने के लिए किया जाता है और फ़ंक्शन का नाम def कीवर्ड के बाद लिखा जाता है। निष्पादन के बाद, यह एक नया फ़ंक्शन ऑब्जेक्ट बनाता है और उसे एक नया नाम प्रदान करता है। इसका उपयोग if else जैसी किसी भी कंट्रोल स्ट्रक्चर के अंदर किया जा सकता है।

निम्नलिखित उदाहरण से यह स्पष्ट होता है।

```
# Using function inside the if...else
def test():
    a=int(input("Enter a value : "))
    if a>0:
        print("Positive")
    else:
        print("Negative")

# Function call
test()
```

```
File  Edit  Shell  Debug  Options  Window  Help
>>> # Using function inside the if...else
>>> def test():
...     a=int(input("Enter a value : "))
...     if a>0:
...         print("Positive")
...     else:
...         print("Negative")
...
...
>>> # Function call
>>> test()
    Enter a value : 11
    Positive
>>> test()
    Enter a value : -9
    Negative
>>>
                                    Ln: 313  Col: 0
```

असाइनमेंट 2.1

1. यूज़र-डिफाइंड फंक्शन का उपयोग करके तीन संख्याओं में से अधिकतम (maximum) ज्ञात करने के लिए एक प्रोग्राम लिखिए।
2. यूज़र-डिफाइंड फंक्शन का उपयोग करके किसी संख्या का फैक्टोरियल (factorial) निकालने के लिए एक प्रोग्राम लिखिए।
3. यूज़र-डिफाइंड फंक्शन का उपयोग करके उपयोगकर्ता द्वारा दर्ज की गई संख्या के अंकों का योग (sum of digits) प्रिंट करने के लिए एक प्रोग्राम लिखिए।
4. यूज़र-डिफाइंड फंक्शन का उपयोग करके उपयोगकर्ता से दिन संख्या (day number) लेकर उसका दिन का नाम (day name) प्रिंट करने के लिए एक प्रोग्राम लिखिए।

पायथन फ़ंक्शन तर्क

तर्क या आर्गुमेंट (argument) वह मान (value) होता है जिसे कोई फंक्शन स्वीकार करता है। ऊपर दिए गए उदाहरण में मान फंक्शन के भीतर ही उपयोगकर्ता से लिए जाते हैं, लेकिन यह भी संभव है कि यूज़र-डिफाइंड फंक्शन को कॉल करते समय उसे मान प्रदान किए जाएँ।

तर्क वह मान होता है जिसे फंक्शन कॉल के दौरान फंक्शन को दिया जाता है, और यह मान फंक्शन हेडर (function header) में परिभाषित संबंधित पैरामीटर (parameter) द्वारा प्राप्त किया जाता है।

उदाहरण: तर्कों के साथ फ़ंक्शन का उपयोग करके दो संख्याओं को जोड़ने के लिए एक पायथन कोड लिखिए।

```
# Sum of 2 numbers by using function
# with specified arguments
def add_numbers(a, b):
    sum = a + b
    print('Sum using function with arguments',a,b,'=',sum)


# Function call with arguments 4,5
add_numbers(4,5)
```

rial Not be Published

```
File Edit Shell Debug Options Window Help
>>> # Sum of 2 numbers by using function
>>> # with specified arguments
>>> def add_numbers(a, b):
...     sum = a + b
...     print('Sum using function with arguments',a,b,'=',sum)
...
...
>>> # Function call with arguments 4,5
>>> add_numbers(4,5)
    Sum using function with arguments 4 5 = 9
>>> # Function call with arguments -4,5
>>> add_numbers(-4,5)
    Sum using function with arguments -4 5 = 1
>>>
                                                Ln: 145 Col: 35
```

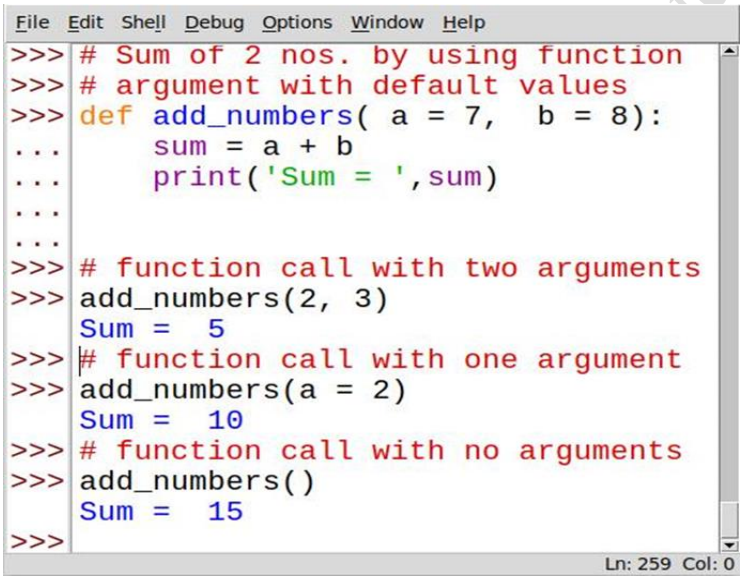
ऊपर दिए गए उदाहरण में, add_numbers() फ़ंक्शन दो पैरामीटर a और b लेता है। फ़ंक्शन add_numbers(4, 5) में यह निर्धारित किया गया है कि पैरामीटर a और b को क्रमशः 4 और 5 के मान प्राप्त होंगे। फ़ंक्शन add_numbers(-4, 5) में यह निर्धारित किया गया है कि पैरामीटर a और b को क्रमशः -4 और 5 के मान प्राप्त होंगे।

डिफ़ॉल्ट पैरामीटर

पायथन में पैरामीटर को एक डिफ़ॉल्ट मान (default value) प्रदान करने की अनुमति देता है। डिफ़ॉल्ट मान वह होता है जो पहले से निर्धारित किया जाता है और तब पैरामीटर को दिया जाता है जब फंक्शन कॉल के समय उसके लिए कोई आर्गुमेंट नहीं दिया जाता। डिफ़ॉल्ट मान देने के लिए = ऑपरेटर का उपयोग किया जाता है।

उदाहरण के लिए, उपरोक्त कोड को निम्न प्रकार से संशोधित किया जा सकता है।

```
# Sum of 2 nos. by using function
# argument with default values
def add_numbers( a = 7,  b = 8):
    sum = a + b
    print('Sum = ',sum)

# function call with two arguments
add_numbers(2, 3)

# function call with one argument
add_numbers(a = 2)

# function call with no arguments
add_numbers()
```



इस उदाहरण में पैरामीटर a और b के लिए क्रमशः डिफ़ॉल्ट मान 7 और 8 प्रदान किए गए हैं। आइए देखें कि यह प्रोग्राम तीन स्थितियों के लिए कैसे काम करता है:

1. add_number(2, 3): फ़ंक्शन कॉल के दौरान दोनों मान पास किए गए हैं। इसलिए, डिफ़ॉल्ट मानों के बजाय इन मानों का उपयोग किया जाता है।
2. add_number(2): फ़ंक्शन कॉल के दौरान केवल एक मान पास किया गया है। इसलिए, पोजिशनल तर्क के अनुसार 2 को तर्क a को सौंपा गया है, और पैरामीटर b के लिए डिफ़ॉल्ट मान का उपयोग किया गया है।

terial Not be Published

3. add_number(): फ़ंक्शन कॉल के दौरान कोई मान पास नहीं किया गया है। इसलिए, दोनों पैरामीटर a और b के लिए डिफ़ॉल्ट मान का उपयोग किया गया है।

भिन्न (division) के हर (denominator) को उचित भिन्न (proper fraction) में बदलने के लिए उपयोगकर्ता-परिभाषित फ़ंक्शन का उपयोग करते हुए एक और उदाहरण पर विचार कीजिए।

```
# Program to convert denominator of division into
# proper fraction using user defined funtion
def mixedFraction(num, deno = 1):
    remainder = num % deno
    if remainder != 0:
        quotient = int(num/deno)
        print("The mixed fraction = ",quotient,'(',remainder,'/',deno,')')
    else:
        print("It evaluates to whole number")
# Function ends
num = int(input("Enter the numerator : "))
deno = int(input("Enter the denominator : "))
print("The number is :",num, '/',deno)
if num > deno:            # Check for proper fraction
    mixedFraction(num,deno) # Function call
else:
    print("It is a proper fraction")
```

© PSSCIVE Draft Study Material Not be Published

```
File Edit Format Run Options Window Help
# proper fraction using user defined funtion
def mixedFraction(num, deno = 1):
    remainder = num % deno
    if remainder != 0:
        quotient = int(num/deno)
        print("The mixed fraction = ",quotient,'(',remainder,'/',deno,')')
    else:
        print("It evaluates to whole number")
# Function ends
num = int(input("Enter the nummerator : "))
deno = int(input("Enter the denominator : "))
print("The number is :",num, '/',deno)
if num > deno:              # Check for proper fraction
    mixedFraction(num,deno) # Function call
else:
    print("It is a proper fraction")
Ln: 13 Col: 38
```

जब आप प्रोग्राम को विभिन्न इनपुट के साथ रन रते हैं, तो आउटपुट नीचे दिए अनुसार होता है।

```
File Edit Shell Debug Options Window Help
    Enter the denominator : 3
    The number is : 20 / 3
    The mixed fraction =  6 ( 2 / 3 )
>>>
    ========================= RESTART
    Enter the nummerator : 20
    Enter the denominator : 2
    The number is : 20 / 2
    It evaluates to whole number
>>>
    ========================= RESTART
    Enter the nummerator : 6
    Enter the denominator : 20
    The number is : 6 / 20
    It is a proper fraction
>>>
```

उपरोक्त प्रोग्राम में, दर्ज किया गया हर 3 है, जो पैरामीटर "deno" को पास किया गया है, इसलिए तर्क deno का डिफ़ॉल्ट मान अधिलेखित (overwrite) हो जाता है। आइए निम्नलिखित फ़ंक्शन कॉल पर विचार कीजिए: mixedFraction(9)। यहाँ, num को 25 प्रदान किया जाएगा और deno डिफ़ॉल्ट मान 1 का उपयोग करेगा।

एक फ़ंक्शन तर्क एक एक्सप्रेशन भी हो सकता है, जैसे

mixedFraction(num+5, deno+5)

ऐसे मामले में, फ़ंक्शन को कॉल करने से पहले तर्क का मूल्यांकन किया जाता है ताकि पैरामीटर को एक वैध मान प्रदान किया जा सके। पैरामीटर उसी क्रम में होने चाहिए जैसे तर्क होते हैं। डिफ़ॉल्ट पैरामीटर फ़ंक्शन हेडर में अंतिम पैरामीटर होने चाहिए, जिसका अर्थ है कि यदि किसी पैरामीटर का डिफ़ॉल्ट मान है तो उसके दाईं ओर के अन्य सभी पैरामीटरों के भी डिफ़ॉल्ट मान होने चाहिए। उदाहरण के लिए,

```
def mixedFraction(num,deno = 1)
```

```
def mixedFraction(num = 2,deno = 1)
Let us consider few more function definition headers:
def calcInterest(principal = 1000, rate, time = 5):
```

उपरोक्त हेडर गलत है क्योंकि डिफ़ॉल्ट अंतिम पैरामीटर होना चाहिए। इसलिए, सही फ़ंक्शन हेडर निम्नानुसार होना चाहिए।

```
def calcInterest(rate, principal = 1000, time = 5):
```

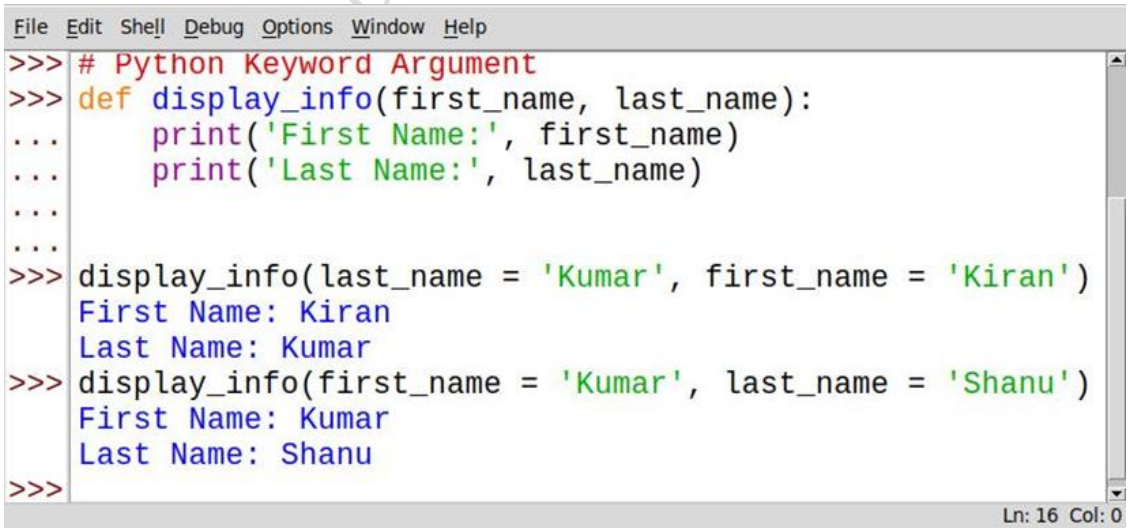
एक बात और ध्यान देने योग्य है कि फ़ंक्शन हेडर में एक्सप्रेशन नहीं हो सकते। इसलिए, निम्नलिखित फ़ंक्शन हेडर त्रुटि देंगे:

```
def mixedFraction(num+5,deno):
def mixedFraction(num+5,deno+5):
```

स्ट्रिंग एक पैरामीटर के रूप में

उपरोक्त प्रोग्राम में फ़ंक्शन को कॉल करते समय केवल संख्यात्मक प्रकार के तर्क पास किए गए हैं। हालाँकि, तर्क के रूप में स्ट्रिंग मान पास करने की भी आवश्यकता हो सकती है, जैसा कि नीचे दर्शाया गया है।

```
def display_info(first_name, last_name):
    print('First Name:', first_name)
    print('Last Name:', last_name)

display_info(last_name = 'Kumar', first_name = 'Kiran')
display_info(first_name = 'Kumar', last_name = 'Shanu')
```



यहाँ, फ़ंक्शन कॉल के दौरान तर्कों के नाम दिए गए हैं। इसलिए, फ़ंक्शन कॉल में first_name फ़ंक्शन परिभाषा में first_name को प्रदान किया जाता है। इसी तरह, फ़ंक्शन कॉल में last_name फ़ंक्शन परिभाषा में last_name को प्रदान किया जाता है। ऐसे मामलों में, तर्कों की स्थिति मायने नहीं रखती।

प्रथम नाम और अंतिम नाम इनपुट लेकर दो स्ट्रिंग्स के संयोजन का उपयोग करके पूरा नाम प्रदर्शित करने के लिए एक और उदाहरण पर विचार करें।

```
# Program to concatenate two strings parameters
# Function Start
def full_name(first_name, last_name):
    full_name = first_name + last_name
    print('Full Name : ',full_name)

# Function End
first_name = input('Enter First Name : ')
Enter First Name : Kumar
last_name = input('Enter Last Name : ')
Enter Last Name : Shanu
# Function Call
full_name(first_name, last_name)
```

[यहाँ आउटपुट दिया गया है]

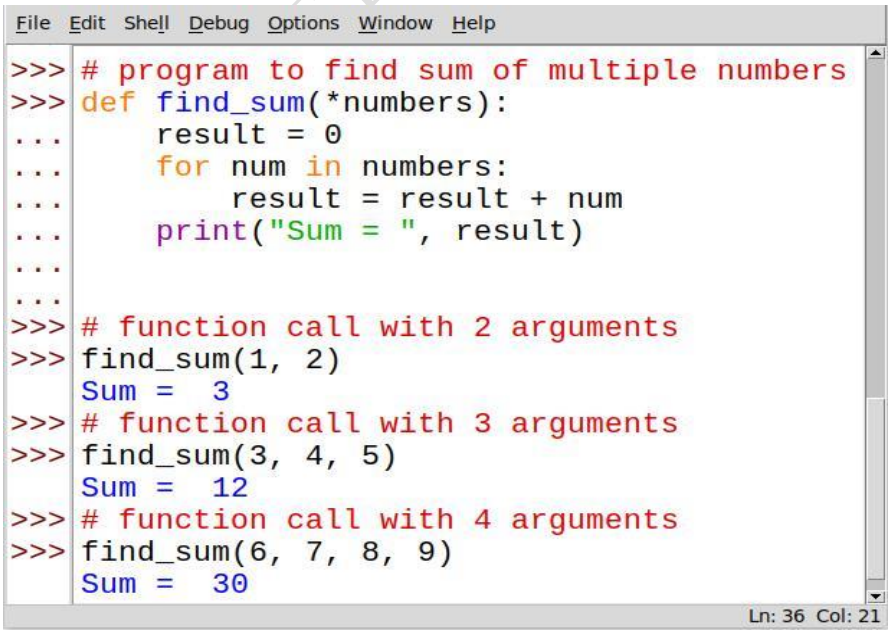
```
File Edit Shell Debug Options Window Help
>>> # Program to concatenate two strings parameters
>>> def full_name(first_name, last_name):
...     full_name = first_name + last_name
...     print('Full Name : ',full_name)
...
...
>>> # Function End
>>> first_name = input('Enter First Name : ')
    Enter First Name : Kumar
>>> last_name = input('Enter Last Name : ')
    Enter Last Name : Shanu
>>> # Function Call
>>> full_name(first_name, last_name)
    Full Name :  KumarShanu
>>> |
                                        Ln: 215 Col: 0
```

मनमाना तर्कों वाला पायथन फ़ंक्शन (Python Function with Arbitrary Arguments)

Study Material Not be Published

कभी-कभी, किसी फ़ंक्शन में पास किए जाने वाले तर्कों की संख्या पहले से ज्ञात नहीं हो सकती है। ऐसी स्थितियों को संभालने के लिए, पायथन में मनमाना तर्क (arbitrary arguments) का उपयोग किया जा सकता है। मनमाना तर्क हमें फ़ंक्शन कॉल के दौरान अलग-अलग संख्या में मान पास करने की अनुमति देते हैं। ऐसे तर्क को निरूपित करने के लिए पैरामीटर नाम से पहले तारांकन (*) का उपयोग किया जा सकता है। उदाहरण के लिए,

```
# program to find sum of multiple numbers
def find_sum(*numbers):
    result = 0
    for num in numbers:
        result = result + num
    print("Sum = ", result)

# function call with 2 arguments
find_sum(1, 2)
Sum =  3
# function call with 3 arguments
find_sum(3, 4, 5)
Sum =  12
# function call with 4 arguments
find_sum(6, 7, 8, 9)
Sum =  30
```



Draft Study Material Not be Published

इस उदाहरण में, फ़ंक्शन find_sum() मनमाना तर्क स्वीकार करता है। इसलिए एक ही फ़ंक्शन को अलग-अलग तर्कों के साथ कॉल करना संभव है। एकाधिक मान प्राप्त करने के बाद, numbers एक सरणी (array) की तरह व्यवहार करते हैं, इसलिए हम प्रत्येक मान तक पहुँचने के लिए for लूप का उपयोग करने में सक्षम हैं।

चर संख्या में तर्क

ऐसे मामलों में जहां आप उन तर्कों की सटीक संख्या नहीं जानते हैं जिन्हें आप किसी फ़ंक्शन में पास करना चाहते हैं, आप *args के साथ निम्नलिखित सिंटैक्स का उपयोग कर सकते हैं:

उदाहरण 2.2

```
File   Edit   Format   View   Help
# Example 2.2
# Define 'plus()' function to accept
# a variable number of arguments

def plus(*args):
    total = 0
    for i in args:
        total += i
    return total

# Calculate the sum
t = plus(20, 30, 40, 50)
print(t)
```

```
File  Edit  Shell  Debug  Options  Window  Help
>>>  140
                                    Ln: 7  Col: 0
```

पास-बाय-ऑब्जेक्ट-रेफरेंस

पायथन एक प्रणाली का उपयोग करता है, जिसे फ़ंक्शन में तर्क पास करने के लिए "कॉल बाय ऑब्जेक्ट रेफरेंस" या "कॉल बाय असाइनमेंट" के रूप में जाना जाता है। यदि आप किसी फ़ंक्शन में पूर्णांक, स्ट्रिंग्स या टुपल्स जैसे तर्क पास करते हैं, तो पासिंग कॉल-बाय-वैल्यू के समान है क्योंकि आप फ़ंक्शन को पास की जा रही अपरिवर्तनीय (immutable) वस्तुओं के मान को नहीं बदल सकते हैं जैसा कि उदाहरण 2.2 में दर्शाया गया है। जबकि परिवर्तनशील (mutable) वस्तुओं को पास करना कॉल बाय रेफरेंस माना जा सकता है क्योंकि जब उनके मान फ़ंक्शन के अंदर बदले जाते हैं, तो यह फ़ंक्शन के बाहर भी परिलक्षित होगा जैसा कि उदाहरण 2.3 में दर्शाया गया है।

उदाहरण 2.3 के लिए कोड और विवरण दिया गया है

```
File  Edit  Format  View  Help
# Example 2.3
# Python code to demonstrate call by value

string = "PSSCIVE"
def test(string):
    string = "NCERT"
    print("Inside Function:", string)

test(string)
print("Outside Function:", string)
```

```
File  Edit  Shell  Debug  Options  Window  Help
Inside Function: NCERT
Outside Function: PSSCIVE
>>>
Ln: 10  Col: 0
```

उदाहरण 2.4 के लिए कोड और विवरण दिया गया है

```
File  Edit  Format  View  Help
# Example 2.4
# Python code to demonstrate call by reference

def add_more(mylist_arg):
    mylist_arg.append(50)
    print("Inside Function: ", mylist_arg)

print()
mylist = [10, 20, 30, 40]
add_more(mylist)
print("Outside Function: ", mylist)
```

```
File  Edit  Shell  Debug  Options  Window  Help
Inside Function:  [10, 20, 30, 40, 50]
Outside Function:  [10, 20, 30, 40, 50]
>>>
Ln: 16  Col: 0
```

2.3.3 मान वापस करने वाले फ़ंक्शन

एक फ़ंक्शन कॉल किए जाने पर मान को वापस भी जा सकता है और नहीं भी। return स्टेटमेंट फ़ंक्शन से मान वापस होता है। अब तक दिए गए उदाहरणों में, फ़ंक्शन गणना करता है और परिणाम प्रदर्शित करता है। वे कोई मान नहीं लौटाते हैं। ऐसे फ़ंक्शन को शून्य (void) फ़ंक्शन कहा जाता है। लेकिन ऐसी स्थिति उत्पन्न हो सकती है, जहां हमें फ़ंक्शन से उसके कॉल करने वाले फ़ंक्शन को मान भेजने की आवश्यकता होती है। यह return स्टेटमेंट का उपयोग करके किया जाता है। return स्टेटमेंट निम्नलिखित कार्य करता है:

- नियंत्रण (control) को कॉल करने वाले फंक्शन को वापस लौटा देता है।

- मान (value) या None लौटाता है।

```
# Program to compute exponent using user defined function.

def calcpow(number, power):          # function definition
    result = 1
    for i in range(1, power + 1):
        result = result * number
    return result

print()
base = int(input("Enter the value for the Base: "))
expo = int(input("Enter the value for the Exponent: "))
answer = calcpow(base, expo)         # function call
print(base, "raised to the power", expo, "is ", answer)
```

Published

```
File  Edit  Shell  Debug  Options  Window  Help
Enter the value for the Base: 5
Enter the value for the Exponent: 4
5 raised to the power 4 is  625
>>>
                                        Ln: 22  Col: 0
```

return स्टेटमेंट शून्य या एक से अधिक मान (values) ले सकता है, जिन्हें अल्पविराम या कॉमा (,) द्वारा अलग किया जाता है। कॉमा का उपयोग करने पर वास्तव में एक ही ट्यूपल (tuple) लौटाया जाता है। एक से अधिक मान लौटाने के लिए ट्यूपल या लिस्ट का उपयोग किया जा सकता है। कॉमा से अलग किए गए कई मान लौटाना, ट्यूपल लौटाने के समान ही होता है।

अब हम ऐसे उदाहरण देखेंगे, जहाँ कोई फंक्शन एक से अधिक मान लौटाता है।

उदाहरण 2.5 के लिए कोड और विवरण दिया गया है

```
File  Edit  Format  View  Help
# Example 2.5

def test(x, y):
    return x * 2, y * 2

a, b = test(5, 10)
print(a)
print(b)
```

```
File  Edit  Shell  Debug  Options  Window  Help
10
30
>>>
                                        Ln: 27  Col: 0
```

उदाहरण 2.6 के लिए कोड और विवरण दिया गया है

```
File  Edit  Format  View  Help
# Example 2.6

def test(x, y):
    return x + y, x - y, x * y

print()
(sum_val, sub, mult) = test(10, 4)
print(sum_val)
print(sub)
print(mult)
```

```
File  Edit  Shell  Debug  Options  Window  Help
>>>  14
     6
     40
                                    Ln: 33  Col: 0
```

आइए, एक ऐसा प्रोग्राम लेते हैं जिसमें फंक्शन का उपयोग करके ट्यूपल (tuple) के माध्यम से आयत (rectangle) का क्षेत्रफल (area) और परिमाप (perimeter) — दो मान — लौटाए जाते हैं।

```
# Program to calculate area and perimeter
# of rectangle using user defined function

def calcAreaPeri(length, breadth):
    area = length * breadth
    perimeter = 2 * (length + breadth)
    return (area, perimeter)

l = float(input("Enter length of the rectangle: "))
b = float(input("Enter breadth of the rectangle: "))

# value of tuples assigned in order they are returned
area, perimeter = calcAreaPeri(l, b)

print("Area is:", area, "\nPerimeter is:", perimeter)
```

```
File  Edit  Shell  Debug  Options  Window  Help
>>>  Enter length of the rectangle: 12
     Enter breadth of the rectangle: 8
     Area is: 96.0
     Perimeter is: 40.0
                                    Ln: 40  Col: 0
```

अब तक हमने सीखा है कि किसी फंक्शन में पैरामीटर (parameter) हो भी सकते हैं और नहीं भी, तथा कोई फंक्शन मान (value) लौटाए भी सकता है और नहीं भी। पायथन में अपनी आवश्यकताओं के अनुसार हम निम्नलिखित तरीकों में से किसी भी रूप में फंक्शन बना सकते हैं:

- बिना तर्क (आर्गुमेंट) और बिना रिटर्न मान (वैल्यू) वाला फंक्शन

- बिना आर्गुमेंट और रिटर्न मान वाला फंक्शन
- आर्गुमेंट के साथ और बिना रिटर्न मान वाला फंक्शन
- आर्गुमेंट के साथ और रिटर्न मान वाला फंक्शन

2.3.4 निष्पादन का प्रवाह (Flow of Execution)

निष्पादन का प्रवाह उस क्रम को दर्शाता है जिसमें प्रोग्राम के स्टेटमेंट्स निष्पादित होते हैं। पायथन इंटरप्रेटर किसी प्रोग्राम के निर्देशों को पहले स्टेटमेंट से निष्पादित करना शुरू करता है। स्टेटमेंट्स को एक-एक करके, ऊपर से नीचे के क्रम में निष्पादित किया जाता है।

जब इंटरप्रेटर किसी फंक्शन की परिभाषा (function definition) को देखता है, तो उस फंक्शन के अंदर के स्टेटमेंट्स तब तक निष्पादित नहीं होते जब तक कि फंक्शन को कॉल नहीं किया जाता। बाद में जब इंटरप्रेटर किसी फंक्शन कॉल को देखता है, तो निष्पादन के प्रवाह में थोड़ा परिवर्तन होता है। उस स्थिति में, अगले स्टेटमेंट पर जाने के बजाय नियंत्रण (control) कॉल किए गए फंक्शन पर चला जाता है और उस फंक्शन के स्टेटमेंट्स को निष्पादित करता है।

इसके बाद नियंत्रण वापस उसी स्थान पर लौट आता है जहाँ से फंक्शन को कॉल किया गया था, ताकि प्रोग्राम के शेष स्टेटमेंट्स निष्पादित हो सकें। इसलिए, जब हम किसी प्रोग्राम को पढ़ते हैं, तो हमें केवल ऊपर से नीचे नहीं पढ़ना चाहिए, बल्कि नियंत्रण या निष्पादन के प्रवाह का भी पालन करना चाहिए। यह भी ध्यान रखना महत्वपूर्ण है कि किसी प्रोग्राम में फंक्शन को कॉल करने से पहले उसे परिभाषित किया जाना आवश्यक है।

आइए, एक उदाहरण के माध्यम से प्रोग्राम में फंक्शन के उपयोग के साथ निष्पादन के प्रवाह को समझते हैं।

उदाहरण 2.7 के लिए कोड दिया गया है।

```
File   Edit   Format   View   Help
# Example 2.7

print("Hello")

def test():
    print("Inside function")

print("Before function call")
test()
print("After function call")
```

```
File  Edit  Shell  Debug  Options  Window  Help
      Hello
      Before function call
      Inside function
      After function call
>>>
                                    Ln: 46  Col: 0
```

उपरोक्त कोड को निष्पादित करने पर निम्नलिखित त्रुटि उत्पन्न होती है:

```
# Program to understand the
# flow of execution using functions.

HelloPython()          # Function Call
def helloPython():      # Function definition
    print("I love Programming")
```

'function not defined' त्रुटि तब उत्पन्न होती है, जबकि फंक्शन परिभाषित किया गया होता है। जब किसी फंक्शन कॉल का सामना होता है, तो नियंत्रण को फंक्शन की परिभाषा पर जाकर उसे निष्पादित करना होता है। उपरोक्त प्रोग्राम में, क्योंकि फंक्शन कॉल फंक्शन की परिभाषा से पहले लिखा गया है, इसलिए इंटरप्रेटर को फंक्शन की परिभाषा नहीं मिलती और परिणामस्वरूप त्रुटि उत्पन्न हो जाती है।

```
File  Edit  Shell  Debug  Options  Window  Help
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py",
line 578, in runcode
    exec(code, self.locals)
  File "/home/dds/pythonprg/p12.12", line 4,
in <module>
    HelloPython()            #Function Call
NameError: name 'HelloPython' is not defined
>>>
                                    Ln: 24  Col: 0
```

इसीलिए, फ़ंक्शन परिभाषा फ़ंक्शन कॉल से पहले की जानी चाहिए जैसा कि नीचे दिखाया गया है:

```
def helloPython(): #Function definition
    print("I love Programming")
helloPython()      #Function Call
```

```
[2] def Greetings(Name):#Function Header
[3] print("Hello "+Name)

[1] Greetings("John")     #Function Call
[4] print("Thanks")
```

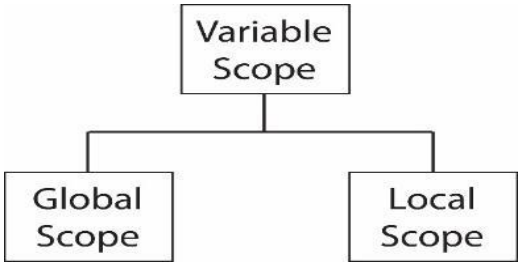
चित्र 2.5: स्टेटमेंट्स के निष्पादन का क्रम

चित्र 2.5 उपरोक्त प्रोग्राम के लिए निष्पादन के प्रवाह की व्याख्या करता है। वर्गाकार कोष्ठकों (square brackets) में संख्या स्टेटमेंट्स के निष्पादन का क्रम दिखाती है।

2.4 चर का दायरा

किसी फंक्शन के अंदर परिभाषित किया गया चर (वेरिएबल) उसके बाहर एक्सेस (access) नहीं किया जा सकता। प्रत्येक चर की पहुँच (accessibility) स्पष्ट रूप से निर्धारित होती है। प्रोग्राम का वह भाग जहाँ कोई वेरिएबल उपलब्ध (accessible) होता है, उस चर का दायरा या स्कोप (scope) कहलाता है।

किसी वेरिएबल का निम्न में से एक स्कोप हो सकता है: जो वेरिएबल पूरे प्रोग्राम में कहीं भी एक्सेस किया जा सकता है, उसे वैश्विक चर (ग्लोबल वेरिएबल) कहते हैं। और जो वेरिएबल किसी फंक्शन या ब्लॉक के भीतर ही सीमित रहता है, उसे स्थानीय चर (लोकल वेरिएबल) कहते हैं।



चित्र 2.6: चर का दायरा

वैश्विक चर (Global Variable) – पायथन में, एक चर जो किसी भी फ़ंक्शन या किसी भी ब्लॉक के बाहर परिभाषित किया गया है, वैश्विक चर के रूप में जाना जाता है। इसे आगे परिभाषित किसी भी फ़ंक्शन में एक्सेस किया जा सकता है। वैश्विक चर में किया गया कोई भी परिवर्तन प्रोग्राम के सभी फ़ंक्शनों को प्रभावित करेगा जहाँ उस चर को एक्सेस किया जा सकता है।

स्थानीय चर (Local Variable) – एक चर जो किसी फ़ंक्शन या किसी ब्लॉक के अंदर परिभाषित किया गया है, स्थानीय चर के रूप में जाना जाता है। इसे केवल उस फ़ंक्शन या ब्लॉक में एक्सेस किया जा सकता है जहाँ यह परिभाषित है। यह केवल तब तक मौजूद रहता है जब तक फ़ंक्शन निष्पादित होता है।

```
# Program to access any variable outside the function

num = 5
def myFunc1( ):
    y = num + 5
    print("Accessing num -> (global) in myFunc1, value = ", num)
    print("Accessing y -> (local variable of myFunc1) accessible, value = ", y)

myFunc1()
print("Accessing num outside myFunc1 ", num)
print("Accessing y outside myFunc1 ", y)
```

```
File  Edit  Shell  Debug  Options  Window  Help
Accessing num -> (global) in myFunc1, value =  5
Accessing y-> (local variable of myFunc1) accessible, value= 10
Accessing num outside myFunc1  5
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "/home/dds/pythonprg/p12.13", line 10, in <module>
    print("Accessing y outside myFunc1 ",y)
NameError: name 'y' is not defined
>>>
Ln: 21  Col: 0
```

वैश्विक चर में कोई भी संशोधन स्थायी होता है और उन सभी फ़ंक्शनों को प्रभावित करता है जहाँ इसका उपयोग किया जाता है। यदि वैश्विक चर के समान नाम वाला एक चर किसी फ़ंक्शन के अंदर परिभाषित किया गया है, तो इसे उस फ़ंक्शन के लिए स्थानीय माना जाता है और यह वैश्विक चर को छुपा देता है। यदि किसी वैश्विक चर के संशोधित मान का उपयोग फ़ंक्शन के बाहर करना है, तो फ़ंक्शन में चर के नाम से पहले कीवर्ड global लगाया जाना चाहिए।

```
# Program to access and modify a variable outside the function

num = 5
def myfunc1():
    global num
    print("Accessing num = ", num)
    num = 10
    print("num reassigned = ", num)

# function ends here
myfunc1()
print("Accessing num outside myfunc1", num)
```

Not be Published

```
File  Edit  Shell  Debug  Options  Window  Help
Accessing num =  5
num reassigned =  10
Accessing num outside myfunc1 10
>>>
Ln: 12  Col: 0
```

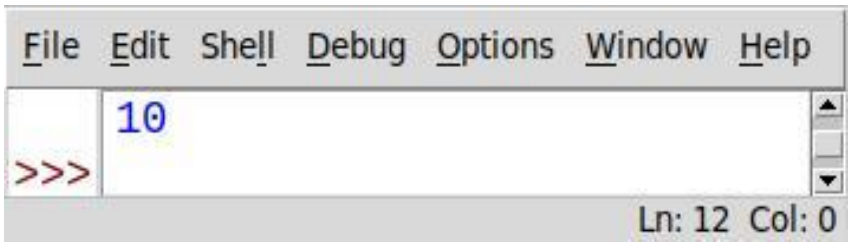
उपरोक्त प्रोग्राम के आउटपुट में, वैश्विक चर num को एक्सेस किया गया है क्योंकि कीवर्ड global को चर के नाम से पहले लगाकर अस्पष्टता को हल किया गया है।

पायथन में अनाम (Anonymous) फ़ंक्शन

अनाम फ़ंक्शन को पायथन में लैम्ब्डा (lambda) फ़ंक्शन भी कहा जाता है, क्योंकि इन्हें सामान्य def कीवर्ड के साथ घोषित करने के बजाय, हम lambda कीवर्ड का उपयोग करते हैं।

उदाहरण 2.8 के लिए कोड और विवरण दिया गया है।

```
File  Edit  Format  View  Help
# Example 2.8

double = lambda x: x * 2
y = double(5)
print(y)
```



फ़ंक्शन परिभाषा में पास स्टेटमेंट

फ़ंक्शन परिभाषाएँ खाली नहीं हो सकती हैं, लेकिन अगर किसी कारण से आपके पास बिना किसी सामग्री के फ़ंक्शन परिभाषा है, तो त्रुटि से बचने के लिए pass स्टेटमेंट डालें।

```
def myfunction():
    pass
```

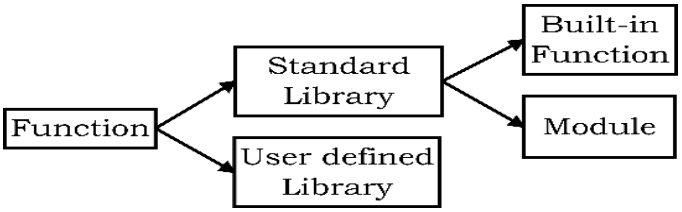
असाइनमेंट 2.2

1. एक ऐसा प्रोग्राम लिखिए जो फंक्शन का उपयोग करके एक धनात्मक पूर्णांक (positive integer) ले और उसका इकाई स्थान (one's place) का अंक लौटाए।
2. एक ऐसा प्रोग्राम लिखिए जो फंक्शन का उपयोग करके दो संख्याएँ n ले और उस संख्या को लौटाए जिसका इकाई स्थान का अंक न्यूनतम (minimum) हो।

2.5 मानक पुस्तकालय (Standard Library)

पायथन में एक बहुत विस्तृत मानक लाइब्रेरी होती है। यह कई बिल्ट-इन फंक्शन्स का एक संग्रह है, जिन्हें प्रोग्राम में आवश्यकता अनुसार कभी भी कॉल किया जा सकता है। इससे प्रोग्रामर का समय बचता है, क्योंकि बार-बार उपयोग होने वाले फंक्शन्स को हर बार बनाने की आवश्यकता नहीं होती।

कोई फंक्शन मानक लाइब्रेरी का हिस्सा हो सकता है यदि वह बिल्ट-इन फंक्शन हो या किसी मॉड्यूल में मौजूद हो; अन्यथा वह एक यूज़र-डिफाइंड फंक्शन (उपयोगकर्ता-परिभाषित फ़ंक्शन) हो सकता है, जैसा कि चित्र 2.5 में दिखाया गया है।



चित्र 2.5: फ़ंक्शन के प्रकार

2.5.1 बिल्ट-इन फ़ंक्शन (Built-in functions)

बिल्ट-इन फ़ंक्शन पायथन में तैयार किए गए फ़ंक्शन हैं जिनका उपयोग प्रोग्रामों में अक्सर किया जाता है। आइए निम्नलिखित पायथन प्रोग्राम का निरीक्षण करें:

```
#Program to calculate square of a number
```

```
a = int(input("Enter a number: ")
b = a * a
print(" The square of ", a, "is", b)
```

उपरोक्त प्रोग्राम में input(), int() और print() बिल्ट-इन फंक्शन्स हैं। इन बिल्ट-इन फंक्शन्स के लिए निष्पादित होने वाले निर्देश (instructions) पहले से ही पायथन इंटरप्रेटर में परिभाषित होते हैं।

आइए, अब एक ऐसे पायथन प्रोग्राम पर विचार करें जिसमें विभिन्न बिल्ट-इन फंक्शन्स के लिए फंक्शन कॉल शामिल हैं।

```
# Program to display
# the input using built in function
name = input("Enter your name: ")
print("Hello " + name)
```

उपरोक्त प्रोग्राम में input() और print() बिल्ट-इन फंक्शन्स हैं। यहाँ print() फंक्शन "Enter your name" स्ट्रिंग को आर्गुमेंट (तर्क) के रूप में स्वीकार करता है। यह फंक्शन एक मान (value) भी लौटाता है। चूँकि फंक्शन नाम से पहले असाइनमेंट (=) ऑपरेटर दिया गया है, इसका अर्थ है कि फंक्शन एक मान लौटाता है, जिसे name चर (वेरिएबल) में संग्रहित किया जाता है। इसलिए input() फंक्शन एक मान स्वीकार करता है और एक मान लौटाता भी है।

उपरोक्त प्रोग्राम में एक अन्य बिल्ट-इन फंक्शन print() का उपयोग "Hello" को उसी रूप में स्ट्रिंग मान के रूप में प्रदर्शित करने के लिए किया गया है।

इसी प्रकार, पायथन में अनेक बिल्ट-इन फंक्शन्स होते हैं। बिल्ट-इन फंक्शन्स के और उदाहरणों के लिए इस अध्याय के अंत में परिशिष्ट 1 (Appendix 1) (तालिका 2.2) देखें।

असाइनमेंट 2.3

1. एक ऐसा प्रोग्राम लिखिए जो किसी संख्या को पढ़े और फिर उसे पायथन के बिल्ट-इन फंक्शन्स का उपयोग करके उसके अष्टाधारी या ऑक्टल और हेक्साडेसिमल समतुल्य में परिवर्तित करे।
2. एक ऐसा प्रोग्राम लिखिए जो एक वास्तविक संख्या (real number) इनपुट करता है और उसे बिल्ट-इन फंक्शन्स का उपयोग करके निकटतम पूर्णांक (nearest integer) में परिवर्तित करता है। यह दी गई संख्या को दशमलव के बाद 3 स्थानों तक पूर्णांकित (राउंड ऑफ) करके भी प्रदर्शित करता है।

2.5.2 मॉड्यूल

बिल्ट-इन फंक्शन्स के अलावा, पायथन की मानक लाइब्रेरी में कई मॉड्यूल भी शामिल होते हैं। जहाँ फंक्शन निर्देशों (instructions) का एक समूह होता है, वहीं मॉड्यूल फंक्शन्स का एक समूह होता है। जैसा कि हम जानते हैं, जब कोई

प्रोग्राम बड़ा होता है, तो कोड को सरल बनाने और पुनरावृत्ति (repetition) से बचने के लिए फंक्शन्स का उपयोग किया जाता है। किसी जटिल समस्या के लिए पूरे कोड को एक ही फाइल में प्रबंधित करना संभव नहीं होता। ऐसे में प्रोग्राम को विभिन्न भागों में विभाजित किया जाता है, जिन्हें अलग-अलग स्तरों पर मॉड्यूल कहा जाता है।

इसके अलावा, यदि हमने किसी प्रोग्राम में कुछ फंक्शन्स बनाए हैं और हम उन्हें किसी अन्य प्रोग्राम में पुनः उपयोग (reuse) करना चाहते हैं, तो हम उन फंक्शन्स को एक मॉड्यूल में सहेज सकते हैं और बाद में उनका पुन: उपयोग कर सकते हैं। एक मॉड्यूल एक पायथन (.py) फाइल के रूप में बनाया जाता है, जिसमें कई फंक्शन्स की परिभाषाएँ होती हैं।

किसी मॉड्यूल का उपयोग करने के लिए हमें उसे इम्पोर्ट (import) या आयात करना होता है। एक बार मॉड्यूल इम्पोर्ट हो जाने के बाद, हम उस मॉड्यूल के सभी फंक्शन्स का सीधे उपयोग कर सकते हैं। इम्पोर्ट स्टेटमेंट का सिंटैक्स इस प्रकार है:

```
import modulename1 [, modulename2, …]
```

यह हमें मॉड्यूल के सभी फंक्शन्स तक पहुँच (access) प्रदान करता है। किसी मॉड्यूल के फंक्शन को कॉल करने के लिए, फंक्शन के नाम से पहले मॉड्यूल का नाम डॉट (.) के साथ लिखना होता है। इसका सिंटैक्स नीचे दर्शाया गया है:

```
modulename.functionname()
```

बिल्ट-इन मॉड्यूल

पायथन लाइब्रेरी में कई बिल्ट-इन मॉड्यूल हैं जो प्रोग्रामर के लिए वास्तव में सुविधाजनक हैं। आइए कुछ सामान्यतः उपयोग किए जाने वाले मॉड्यूल और उन मॉड्यूल में उपलब्ध अक्सर उपयोग किए जाने वाले फ़ंक्शनों का पता लगाएं – math, random, statistics।

math मॉड्यूल – इसमें विभिन्न प्रकार के गणितीय फ़ंक्शन होते हैं। इस मॉड्यूल के अधिकांश फ़ंक्शन एक फ़्लोट मान (वेल्यु) लौटाते हैं। किसी मॉड्यूल का उपयोग करने के लिए, इसे प्रोग्राम में कहीं भी केवल एक बार आयात (इम्पोर्ट) किया जाना चाहिए। इसलिए निम्नलिखित स्टेटमेंट का उपयोग करके math मॉड्यूल को आयात करें।

```
import math
```

आइए, gcd() फंक्शन का एक उदाहरण लेते हैं, जिसका उपयोग दो संख्याओं का महत्तम समापवर्तक (Greatest Common Divisor) ज्ञात करने के लिए किया जाता है।

```
>>> import math
>>> math.gcd(10,6)
2
```

फ़ंक्शन gcd(x,y) में, दोनों इनपुट पैरामीटर x और y धनात्मक पूर्णांक हैं। math मॉड्यूल में सामान्यतः उपयोग किए जाने वाले कुछ फ़ंक्शन इस अध्याय के अंत में परिशिष्ट 1 (तालिका 12.2) में दिए गए हैं।

random मॉड्यूल – इस मॉड्यूल में ऐसे फ़ंक्शन होते हैं जिनका उपयोग यादृच्छिक संख्याएँ उत्पन्न करने के लिए किया जाता है। इस मॉड्यूल का उपयोग करने के लिए, हम इसे निम्नलिखित स्टेटमेंट का उपयोग करके आयात कर सकते हैं:

import random

आइए random() फ़ंक्शन लेते हैं जिसका उपयोग 0.0 से 1.0 की सीमा में रेंडम रियल (यादृच्छिक वास्तविक) संख्या (फ्लोट) उत्पन्न करने के लिए किया जाता है।

```
>>> import random
>>> random.random()
0.011035221592589295
```

random मॉड्यूल में सामान्यतः उपयोग किए जाने वाले कुछ फ़ंक्शन इस अध्याय के अंत में परिशिष्ट 1 (तालिका 12.3) में दिए गए हैं।

statistics मॉड्यूल – यह मॉड्यूल संख्यात्मक (वास्तविक-मूल्य) डेटा के आँकड़ों की गणना के लिए फ़ंक्शन प्रदान करता है। इसे निम्नलिखित स्टेटमेंट का उपयोग करके प्रोग्राम में शामिल किया जा सकता है।

आइए mean () फ़ंक्शन लेते हैं जिसका उपयोग दी गई संख्याओं के समांतर माध्य (arithmetic mean) की गणना करने के लिए किया जाता है।

```
>>> import statistics
>>> statistics.mean([1,2,3,4,5])
3
```

statistics मॉड्यूल में सामान्यतः उपयोग किए जाने वाले कुछ फ़ंक्शन इस अध्याय के अंत में परिशिष्ट 1 (तालिका 12.4) में दिए गए हैं।

from स्टेटमेंट (from Statement)

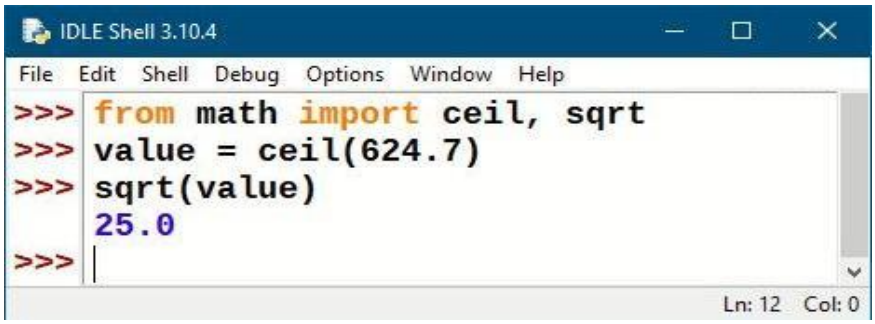
किसी मॉड्यूल को आयात करके सभी फ़ंक्शनों को मेमोरी में लोड करने के बजाय, किसी स्टेटमेंट का उपयोग केवल आवश्यक फ़ंक्शनों तक पहुँचने के लिए किया जा सकता है। यह किसी मॉड्यूल में सभी फ़ंक्शनों के बजाय केवल निर्दिष्ट फ़ंक्शन(फ़ंक्शनों) को लोड करता है। इसका सिंटैक्स है:

```
>>> from module_name import function_name [, function_name,...]
```

"from स्टेटमेंट" का उपयोग करके आयात (इम्पोर्ट) करने पर फ़ंक्शन का उपयोग करने के लिए हमें उससे पहले मॉड्यूल का नाम लगाने की आवश्यकता नहीं होती है। बल्कि हम सीधे फ़ंक्शन को कॉल कर सकते हैं जैसा कि निम्नलिखित उदाहरणों में दिखाया गया है:

उदाहरण 2.9 के लिए कोड और विवरण दिया गया है

उदाहरण 2.10 के लिए कोड और विवरण दिया गया है



उदाहरण 2.10 में, 624.7 का ceil मान वेरिएबल "value" में संग्रहीत किया गया है और फिर वेरिएबल "value" पर sqrt फ़ंक्शन लागू किया गया है। उपरोक्त उदाहरण को इस प्रकार फिर से लिखा जा सकता है:

>>> sqrt (ceil (624.7))

sqrt() फंक्शन का निष्पादन, ceil() फंक्शन के आउटपुट पर निर्भर करता है।

624.7 का पूर्णांक (integer) भाग प्राप्त करने के लिए, math मॉड्यूल के trunc() फंक्शन का उपयोग करें।

```
#ceil and sqrt already been imported above
>>> from math import trunc
>>> sqrt ( trunc (625.7))
The above code will give the result as 25.0.
```

एक प्रोग्रामिंग स्टेटमेंट जिसमें फ़ंक्शन या एक्सप्रेशन आउटपुट प्राप्त करने के लिए एक-दूसरे के निष्पादन पर निर्भर होते हैं, उसे कम्पोजीशन (composition) कहा जाता है, यहाँ कम्पोजीशन के कुछ अन्य उदाहरण दिए गए हैं:

a = int (input ("First number: "))

print ("Square root of ", a , " = ", math.sqrt(a))

print(floor(a+(b/c)))

math.sin (float(h)/float(c))

यूज़र-डिफाइंड मॉड्यूल को इम्पोर्ट करना (Importing User Defined Module)

पायथन की मानक लाइब्रेरी में उपलब्ध मॉड्यूल्स के अलावा, हम अपने स्वयं के फंक्शन्स से युक्त अपना मॉड्यूल भी बना सकते हैं। इसके लिए आपको केवल एक ऐसी फ़ाइल बनानी होती है जिसमें वैध (legitimate) पायथन कोड हो, और उस फ़ाइल को .py एक्सटेंशन के साथ नाम देना होता है। किसी मॉड्यूल का उपयोग करने के लिए हमें उसे import स्टेटमेंट का उपयोग करके इम्पोर्ट करना होता है। एक बार मॉड्यूल इम्पोर्ट हो जाने के बाद, हम उस मॉड्यूल के सभी फंक्शन्स का सीधे उपयोग कर सकते हैं। आइए, यह सीखते हैं कि मॉड्यूल कैसे बनाया जाता है और उसके फंक्शन्स का उपयोग करने के लिए उसे कैसे इम्पोर्ट किया जाता है।

```
# Create a user defined module "basic_math"
# This module contains basic arithmetic operations.

# Beginning of module

def addnum(x, y):
    return (x + y)
```

```
def subnum(x, y):
    return (x - y)

def multnum(x, y):
    return (x * y)

def divnum(x, y):
    if y == 0:
        print("Division by Zero Error")
    else:
        return (x / y)          # End of module
```

नोट: इस मॉड्यूल को फ़ाइल नाम p2_9.py से सेव करें, न कि p2.9 से, जैसा कि इस पुस्तक में फ़ाइल नामकरण में दिखाया गया है। क्योंकि डॉट (.) का पायथन में विशेष महत्व होता है, इसलिए इसे ऐसे मॉड्यूल के नाम में उपयोग नहीं किया जा सकता, जिसे किसी अन्य पायथन प्रोग्राम में संदर्भित (refer) करना हो।

उपरोक्त कोड को एक अलग फ़ाइल में p2_9.py नाम से सेव करें। अब नीचे दिया गया कोड एक नई फ़ाइल में लिखें। इसे p2_10.py के रूप में सेव करें और फिर आउटपुट प्राप्त करने के लिए इसे रन करें।

```
# Statements for using module p2_9

import p2_9
print()

# Call addnum() function of module p2_9
a = p2_9.addnum(2, 5)
print(a)

# Call subnum() function of module p2_9
b = p2_9.subnum(2, 5)
print(b)

# Call multnum() function of module p2_9
c = p2_9.multnum(2, 5)
print(c)

# Call divnum() function of module p2_9
d = p2_9.divnum(2, 5)
print(d)
```

© PSSCIVE Draft Study Material 1

असाइनमेंट 2.4

एक 'length_conversion.py' नाम का मॉड्यूल बनाइए, जिसमें विभिन्न लंबाई (length) रूपांतरण (conversion) के लिए जैसे mile_to_km(), km_to_mile(), feet_to_inches() और inches_to_feet() फंक्शन्स संग्रहीत करता हों। उपरोक्त मॉड्यूल का उपयोग करने के लिए एक प्रोग्राम भी लिखिए।

सारांश

यह सत्र फंक्शन्स (functions) का परिचय देता है, जो पुनः उपयोग किए जा सकने वाले (reusable) कोड के ब्लॉक होते हैं और विशिष्ट कार्यों को करने के लिए उपयोग किए जाते हैं। इस सत्र में विद्यार्थी सीखा:

• फंक्शन्स को कैसे परिभाषित (define) और कॉल (call) किया जाता है।

• पैरामीटर्स (parameters) और रिटर्न वैल्यू (return values) का उपयोग।

• कोड को मॉड्यूलर (modular), पठनीय (readable) और पुनः उपयोग योग्य (reusable) बनाने में फंक्शन्स का महत्व।

पुनरावृत्ति (repetition) को कम करने में फंक्शन्स मदद करते हैं और प्रोग्राम को अधिक व्यवस्थित (organized) तथा कुशल (efficient) बनाते हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. किसी विशेष कार्य को पूरा करने वाले निर्देशों के नामित समूह को क्या कहा जाता है? (क) स्ट्रिंग (string) (ब) कंट्रोल (control) (स) ट्यूपल (tuple) (द) फंक्शन (Function)
2. फ़ंक्शन बनाने के लिए कौन-सा कीवर्ड उपयोग किया जाता है? (अ) fun (ब) define (स) def (द) function
3. फंक्शन्स के दो मुख्य प्रकार कौन-से हैं? (अ) कस्टम फंक्शन (Custom function) (ब) बिल्ट-इन फंक्शन और यूज़र-डिफाइंड फंक्शन (Built-in function & User defined function) (स) यूज़र फंक्शन (User function) (द) सिस्टम फंक्शन (System function)
4. पायथन में id() फंक्शन का उपयोग क्या है? (अ) id ऑब्जेक्ट की पहचान (identity) लौटाता है (ब) हर ऑब्जेक्ट का एक विशिष्ट id नहीं होता (स) उपरोक्त सभी (द) उपरोक्त में से कोई नहीं
5. गणितीय फ़ंक्शन को दर्शाता है? (अ) sqrt (ब) rhombus (स) add (द) minus
6. पायथन रन-टाइम पर अनाम (anonymous) फ़ंक्शन बनाने की सुविधा देता है, जिसे कहते है। (अ) lambda (ब) pi (स) anonymous (द) इनमें से कोई नहीं
7. निम्नलिखित पायथन कोड का आउटपुट होगा? (अ) 12 (ब) 14 (स) 27 (द) 64

© PSSCIVE Draft 2023

```
y = 4
z = lambda x: x * y
print(z(3))
```

8. निम्नलिखित पायथन कोड का आउटपुट होगा? (अ) 27, 81, 343 (ब) 6, 9, 12 (स) 9, 27, 81 (द) 64, 27, 343

```
L = [lambda x: x ** 2, lambda x: x ** 3, lambda x: x ** 4]
for f in L:
    print(f(3))
```

9. निम्नलिखित में से कौन-सा statistics module का फ़ंक्शन नहीं है? (अ) fabs() (ब) mean() (स) median() (द) mode()
10. किसी प्रोग्राम में उपयोगकर्ता द्वारा बनाया गया फंक्शन दूसरे प्रोग्राम में पुनः उपयोग किए जा सकते है। ऐसे फंक्शन को किसके अंतर्गत संग्रहीत किया जाता है? (अ) स्ट्रिंग (string) (ब) कंट्रोल (control) (स) ट्यूपल (tuple) (द) मॉड्यूल (module)

Not to be Published

ब. रिक्त स्थान भरें

1. किसी कंप्यूटर प्रोग्राम को अलग-अलग स्वतंत्र कोड ब्लॉक्स या विभिन्न नामों और विशिष्ट कार्यों वाले उप-समस्याओं में विभाजित करने की प्रक्रिया को _____ प्रोग्रामिंग कहा जाता है।
2. प्रोग्रामर की आवश्यकताओं के अनुसार किसी कार्य को पूरा करने के लिए परिभाषित फंक्शन को _____ फंक्शन कहा जाता है।
3. _____ वह मान होता है, जिसे फंक्शन कॉल के दौरान फंक्शन को दिया जाता है और जिसे फंक्शन हेडर में परिभाषित संबंधित पैरामीटर प्राप्त करता है।
4. _____ स्टेटमेंट फ़ंक्शन से मान (values) वापस लौटाता है।
5. अनेक बिल्ट-इन फंक्शन्स का वह संग्रह, जिन्हें आवश्यकता अनुसार प्रोग्राम में कॉल किया जा सकता है, _____ कहलाता है।
6. मॉड्यूल एक पायथन फ़ाइल के रूप में बनाया जाता है, जिसकी एक्सटेंशन _____ होती है, और जिसमें फंक्शन परिभाषाओं का संग्रह होता है।
7. किसी मॉड्यूल का उपयोग करने के लिए हमें _____ स्टेटमेंट का उपयोग करके उसे इम्पोर्ट (import) करना होता है।
8. किसी मॉड्यूल के फंक्शन को कॉल करने के लिए, फंक्शन के नाम से पहले मॉड्यूल का नाम _____ (separator) के साथ लिखा जाता है।
9. रैंडम (random) संख्याएँ उत्पन्न करने के लिए उपयोग किए जाने वाले फंक्शन्स _____ मॉड्यूल में होते हैं।

10. किसी मॉड्यूल को इम्पोर्ट करके उसकी सभी फंक्शन्स को मेमोरी में लोड करने के बजाय, केवल आवश्यक फंक्शन्स को एक्सेस करने के लिए _____ स्टेटमेंट का उपयोग किया जा सकता है।

स. सत्य या असत्य बताएं

1. फंक्शन का उपयोग मॉड्यूलरिटी (modularity) और पुनः उपयोगिता (reusability) प्राप्त करने के एक साधन के रूप में किया जाता है।
2. फंक्शन हेडर हमेशा अर्धविराम या सेमीकोलन (;) पर समाप्त होता है।
3. लैम्ब्डा (lambda) फंक्शन में return स्टेटमेंट होता है।
4. पायथन में लैम्ब्डा (lambda) एक अनाम (anonymous) फंक्शन है।
5. मॉड्यूल फंक्शन्स का एक समूह होता है।
6. एक बार मॉड्यूल इम्पोर्ट करने के बाद, हम उसके सभी फंक्शन्स का सीधे उपयोग कर सकते हैं।
7. "from स्टेटमेंट" का उपयोग करके इम्पोर्ट किए गए फंक्शन को उपयोग करने के लिए, उसके पहले मॉड्यूल का नाम लिखने की आवश्यकता नहीं होती।
8. पायथन की मानक लाइब्रेरी में उपलब्ध मॉड्यूल्स के अलावा अपना स्वयं का फंक्शन बनाना संभव नहीं है।
9. randrange() रैंडम (random) मॉड्यूल का एक फंक्शन है।
10. fmod () statistics मॉड्यूल का एक फ़ंक्शन है।

द. लघु उत्तरीय प्रश्न

1. निम्नलिखित कोड में यदि कोई एरर हों तो उन्हें पहचानें।

(अ)
```
def create (text, freq):
    for i in range (1, freq):
    print text
    create (5) #function call
```

(ब)
```
from math import sqrt,ceil
    def calc ():
    print cos (0)
    calc () #function call
```

(स)
```
mynum = 9 def add9():
    mynum = mynum + 9
    print mynum
    add9() #function call
```

(द) def findValue(val) = 1.1, val2, val3):

```
final = (val2 + val3)/ vall
print(final)
findvalue () #function call
```

(इ) def greet ():

```
return ("Good morning")
greet () = message #function call
```

2. एक ऐसा प्रोग्राम लिखिए जो यूज़र-डिफाइंड फंक्शन का उपयोग करके जाँच करे कि कोई संख्या 7 से विभाज्य है या नहीं, जहाँ संख्या फंक्शन को पैरामीटर के रूप में दी जाए।

3. एक ऐसा प्रोग्राम लिखिए जो यूज़र-डिफाइंड फंक्शन का उपयोग करे, जो नाम और लिंग (M पुरुष के लिए, F महिला के लिए) स्वीकार करता है और लिंग के आधार पर Mr./Ms. जोड़कर नाम प्रदर्शित करता है।

4. एक ऐसा प्रोग्राम लिखिए जो दो यूज़र-डिफाइंड फंक्शन्स का उपयोग करके तापमान को सेल्सियस से फारेनहाइट और फारेनहाइट से सेल्सियस में परिवर्तित करे।

5. एक ऐसा प्रोग्राम लिखिए जिसमें यूज़र-डिफाइंड फंक्शन का उपयोग करके द्विघात समीकरण (quadratic equation) के गुणांक (coefficients) स्वीकार किए जाएँ और उसका सारणिक (determinant) ज्ञात किया जाए। उदाहरण के लिए, यदि गुणांक a, b, c में संग्रहित हैं, तो सारणिक $b^2 - 4ac$ होगा। सारणिक के धनात्मक, शून्य या ऋणात्मक होने की स्थिति के अनुसार उपयुक्त परिणाम प्रदर्शित करें।

6. ABC स्कूल ने अपने वार्षिक दिवस समारोह में लकी ड्रॉ के लिए सभी अभिभावकों को 1 से 600 तक के अद्वितीय टोकन आईडी दिए हैं। विजेता को विशेष पुरस्कार मिलेगा। इस कार्य को स्वचालित करने के लिए पायथन में एक प्रोग्राम लिखिए। (संकेत: random मॉड्यूल का उपयोग करें)

7. एक ऐसा प्रोग्राम लिखिए जो यूज़र-डिफाइंड फंक्शन का उपयोग करके मूलधन (Principal Amount), दर (Rate), समय (Time) और ब्याज के संयोजन की संख्या (Number of Times Interest is Compounded) को स्वीकार करे और चक्रवृद्धि ब्याज (Compound Interest) की गणना करके प्रदर्शित करे। (संकेत: CI = P(1 + R/N)^(NT))

8. एक ऐसा प्रोग्राम लिखिए जिसमें यूज़र-डिफाइंड फंक्शन दो संख्याओं को पैरामीटर के रूप में स्वीकार करे। यदि पहली संख्या दूसरी से छोटी हो, तो दोनों संख्याओं को अदला-बदली (swap) करके लौटाया जाए; अन्यथा उसी क्रम में लौटाया जाए।

9. एक ऐसा प्रोग्राम लिखिए जिसमें विभिन्न आकृतियों जैसे वर्ग (square), आयत (rectangle), त्रिभुज (triangle), वृत्त (circle) और बेलन (cylinder) के क्षेत्रफल, परिमाप या पृष्ठीय क्षेत्रफल (surface area) की गणना के लिए यूज़र-डिफाइंड फंक्शन्स हों। ये फंक्शन्स आवश्यक मानों को पैरामीटर के रूप में स्वीकार करें और गणना का परिणाम लौटाएँ। मॉड्यूल को इम्पोर्ट करके उपयुक्त फंक्शन्स का उपयोग करें।

10. एक ऐसा प्रोग्राम लिखिए जो किसी भी पाँच सामान्य ज्ञान (GK) प्रश्नों की प्रश्नोत्तरी (क्विज) बनाए। प्रश्नों को रेंडम (random) रूप से प्रदर्शित किया जाए। क्विज का स्कोर गणना करने हेतु score() नामक यूज़र-डिफाइंड फंक्शन बनाएँ तथा remark(score value) नामक एक अन्य फंक्शन बनाएँ जो अंतिम स्कोर के आधार पर उपयुक्त टिप्पणी (remarks) प्रदर्शित करे:

Marks (अंक)	Remarks (टिप्पणी)
5	उत्कृष्ट
4	बहुत अच्छा
3	अच्छा
2	अधिक स्कोर करने के लिए और पढ़ें
1	रुचि लेने की आवश्यकता है
0	सामान्य ज्ञान हमेशा आपकी मदद करेगा। इसे गंभीरता से लें।

परिशिष्ट 1 (Appendix 1)

तालिका 2.1 : math मॉड्यूल में सामान्यतः उपयोग किए जाने वाले फ़ंक्शन

Function Syntax	**Arguments**	**Returns**	**Example Output**

math.ceil(x)	x may be an integer or floating-point number	ceiling value of x	>>> math.ceil(-9.7)
			-9
			>>> math.ceil (9.7)
			10
			>>> math.ceil(9) 9

math.floor(x)	x may be an integer or floating-point number	floor value of x	>>> math.floor(-4.5)
			-5
			>>> math.floor(4.5)
			4
			>>> math.floor(4)
			4

math.fabs(x)	x may be an integer or floating-point number	absolute value of x	>>> math.fabs(6.7) 6.7
			>>> math.fabs(-6.7) 6.7
			>>> math.fabs(-4) 4.0

© PSSCIVE Draft Study M... Not be Published

math.factorial(x)	x is a positive integer	factorial of x	>>> math.factorial(5) 120

math.fmod(x,y)	x and y may be an integer or floating-point number	x % y with sign of x	>>> math.fmod(4,4.9)
			4.0
			>>> math.fmod(4.9,4.9)
			0.0
			>>> math.fmod(-4.9,2.5)
			-2.4
			>>> math.fmod(4.9,-4.9)
			0.0

math.gcd(x,y)	x, y are positive integers	gcd (greatest common divisor) of x and y	>>> math.gcd(10,2) 2

math.pow(x,y)	x, y may be an integer or floating-point number	x^y (x raised to the power y)	>>> math.pow(3,2)
			9.0
			>>> math.pow(4,2.5)
			32.0
			>>> math.pow(6.5,2)
			42.25
			>>> math.pow(5.5,3.2)
			233.97

math.sqrt(x)	x may be a positive integer or floating-point number	square root of x	>>> math.sqrt(144) 12.0
			>>> math.sqrt(.64)
			0.8

math.sin(x)	x may be an integer or floating-point number in radians	sine of x in radians	>>> math.sin(0)
			0
			>>> math.sin(6)
			-0.279

तालिका 2.2 : random मॉड्यूल में सामान्यतः उपयोग किए जाने वाले फ़ंक्शन

Function Syntax	**Argument**	**Return**	**Example Output**

© PSSCIVE Draft Study Material Not be Published

random. random()	No argument (void)	Random Real Number (float) in the range 0.0 to 1.0	>>> random. random() 0.65333522
random. randrange(x,y)	x and y are positive integers signifying the start and stop value	Random integer between x and y	>>> random.randrange(2,7) 2
random. randrage(y)	y is a positive integer signifying the stop value	Random integer between 0 and y	>>> random.randrange(5) 4

तालिका 2.3 : statistics मॉड्यूल में उपलब्ध कुछ फ़ंक्शन

Function Syntax	Argument	Return	Example Output
statistics.mean(x)	x is a numeric sequence	arithmetic mean	>>> statistics. mean([11,24,32,45,51]) 32.6
statistics.median(x)	x is a numeric sequence	median (middle value) of x	>>>statistics. median([11,24,32,45,51]) 32
statistics.mode(x)	x is a sequence	mode (the most repeated value)	>>> statistics. mode([11,24,11,45,11]) 11 >>> statistics. mode(("red","blue","red")) 'red'

मॉड्यूल 2. डेटा विज्ञान

(Data Science)

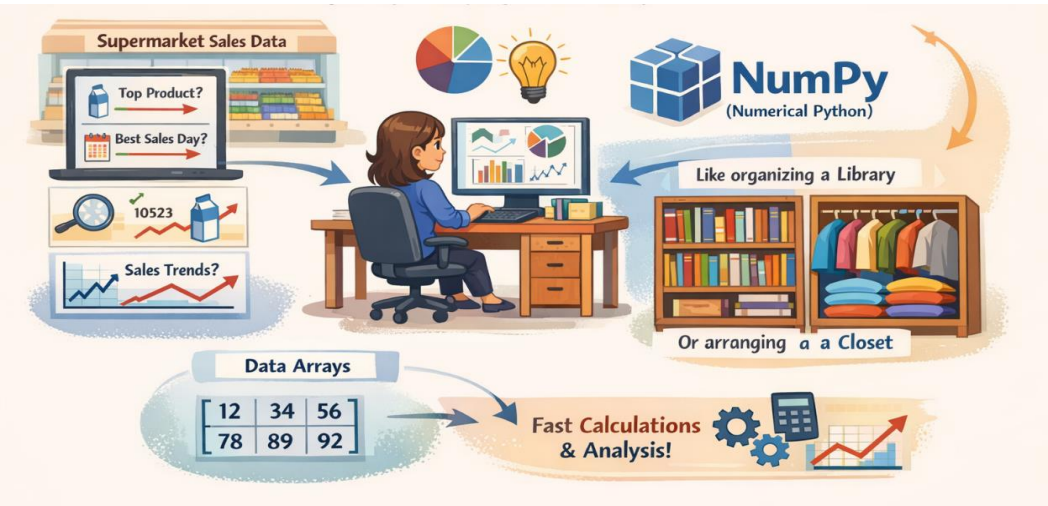
मॉड्यूल संक्षिप्त परिचय (Module Overview)

डेटा विज्ञान (Data Science) का अर्थ डेटा का उपयोग करके ज्ञान प्राप्त करना और बेहतर निर्णय लेना है। यह कार्य हम डेटा को एकत्रित (collect), साफ (clean), विश्लेषण (analyse) और व्याख्या (interpret) करके उसमें छिपे पैटर्न और महत्वपूर्ण जानकारियाँ (insights) खोजकर करते हैं। डेटा विज्ञान उद्योगों को अधिक स्मार्ट, तेज़ और सूचित निर्णय लेने में मदद करता है।

कल्पना कीजिए कि एक सुपरमार्केट हजारों उत्पादों की दैनिक बिक्री रिकॉर्ड करता है। यदि प्रबंधक जानना चाहता है कि कौन सा उत्पाद सबसे अधिक बिकता है, किस दिन सबसे अधिक बिक्री हुई या समय के साथ बिक्री कैसे बदल रही है, तो वे मैन्युअल रूप से प्रत्येक बिल की जांच नहीं कर सकते। इसके बजाय, वे डेटा विज्ञान उपकरणों का उपयोग करते

हैं जो सेकंडों में बड़ी मात्रा में संख्याओं को व्यवस्थित (organize), विश्लेषण (analyse) और संसाधित (process) कर सकते हैं।

इसी तरह, यह इकाई आपको नमपी या NumPy ((Numerical Python) से परिचित कराती है, जो बड़ी मात्रा में डेटा को कुशलतापूर्वक संभालने के लिए एक शक्तिशाली उपकरण है। जैसे किसी पुस्तकालय (लाइब्रेरी) में पुस्तकों को व्यवस्थित रूप से रखना या अलमारी में कपड़ों को व्यवस्थित करना हमारा समय बचाता है, उसी तरह NumPy संख्याओं को एरे (arrays) में व्यवस्थित करता है जिन्हें आसानी से एक्सेस (access) किया जा सकता है, फिर से आकार (reshape) दिया जा सकता है और उन पर गणना (compute) की जा सकती है।



चित्र 2.0 NumPy का परिचय

इस इकाई के अंत तक, आप NumPy का उपयोग करना सीखेंगे। यह न केवल डेटा विज्ञान की नींव (foundation) है, बल्कि मशीन लर्निंग (Machine Learning) और आर्टिफिशियल इंटेलिजेंस (Artificial Intelligence) जैसे अनुप्रयोगों की ओर एक महत्वपूर्ण कदम भी है, जहाँ बड़ी मात्रा में डेटा को तेज़ी और सटीकता के साथ संसाधित (process) करने की आवश्यकता होती है।

अधिगम के परिणाम (Learning Outcomes)

इस मॉड्यूल को पूरा करने के बाद आप निम्नलिखित कार्य करने में सक्षम होंगे:

- बड़े डेटासेट को संग्रहीत करने के लिए NumPy एरे बनाना और उनका उपयोग करना।
- एरे को जोड़कर, विभाजित करके, पुनःआकार देकर (reshape), आकार बदलकर (resize) तथा उनके आयामों में परिवर्तन करके उन्हें संचालित करना।
- एरे पर कुशलतापूर्वक अंकगणितीय और गणितीय संगणनाएँ करना।
- डेटा विश्लेषण के लिए समष्टि फलनों (aggregate functions) और मैट्रिक्स संक्रियाओं (matrix operations) का उपयोग करना।

मॉड्यूल संरचना (Module Structure)

सत्र 1. NumPy का परिचय (Introduction to NumPy)

सत्र 2. NumPy का उपयोग करके एरे हेरफेर (Array Manipulation using NumPy)

सत्र 3. NumPy का उपयोग करके एरे संगणना (Array Computation using NumPy)

सत्र 1. NumPy का परिचय (Introduction to NumPy)

एक पुस्तकालय के बारे में सोचें जहाँ हजारों पुस्तकें रखी हैं। यदि ये पुस्तकें व्यवस्थित ढंग से नहीं रखी हो, तो किसी एक को ढूँढना कठिन और समय लेने वाला हो जाता है। लेकिन जब उन्हें व्यवस्थित रूप से अलमारियों पर रखा जाता है, तो उन्हें खोजना और प्रबंधित करना तेज़ और आसान हो जाता है। उसी तरह, NumPy एरे डेटा को एक संरचित तरीके से व्यवस्थित करने में मदद करते हैं, जिससे सामान्य पायथन सूचियों (lists) की तुलना में इसे संसाधित करना आसान और तेज़ हो जाता है। चित्र 1.1 इसे दर्शाता है।

चित्र 1.1: NumPy का उपयोग करके डेटा का संगठन

इस सत्र में आप समझेंगे कि NumPy एरे कैसे काम करते हैं, वे सूचियों (lists) की तुलना में तेज़ क्यों हैं और उन्हें कैसे बनाया और उपयोग किया जाता है। आप एरे के प्रमुख गुणों जैसे आकार (shape), परिमाण (size), आयाम (dimensions), डेटा प्रकार और मेमोरी उपयोग के बारे में भी सीखेंगे। आप 1-डी, 2-डी और बहु-आयामी एरे बनाने का अभ्यास करेंगे और पुनःआकार देना (reshaping), समतल करना (flattening) और प्रसारण (broadcasting) जैसी तकनीकों का भी पता लगाएंगे।

1.1 परिचय

पायथन में, हम कई आइटम संग्रहीत करने के लिए सूचियों (lists) का उपयोग करते हैं। हालाँकि, जब हमारे पास बड़ी मात्रा में डेटा होता है तो सूचियाँ संसाधित होने में धीमी हो सकती हैं। NumPy एक एरे ऑब्जेक्ट प्रदान करता है जो पारंपरिक पायथन सूचियों की तुलना में बहुत तेज़ है। NumPy एरे को सूचियों के विपरीत, मेमोरी में एक निरंतर स्थान

पर संग्रहीत किया जाता है, इसलिए प्रक्रियाएँ उन्हें बहुत कुशलता से एक्सेस और हेरफेर कर सकती हैं। NumPy एरे के सभी तत्वों का प्रकार समान होते, जबकि पायथन सूची के तत्व पूरी तरह से भिन्न प्रकार के हो सकते हैं।

NumPy का मतलब Numerical Python है। यह पायथन में वैज्ञानिक कंप्यूटिंग के लिए मौलिक पैकेज है। यह एक लाइब्रेरी है जो एक बहुआयामी एरे ऑब्जेक्ट और एरे और मैट्रिसेस पर तेज़ गणितीय संगणना के लिए फ़ंक्शन प्रदान करती है। NumPy में यह शक्तिशाली n-आयामी एरे डेटा प्रोसेसिंग को गति देता है। NumPy ऑब्जेक्ट का उपयोग मुख्य रूप से एरे या मैट्रिसेस बनाने के लिए किया जाता है जिनका उपयोग डीप लर्निंग या मशीन लर्निंग मॉडल में किया जा सकता है।

1.2 एरे (Array)

एरे एक डेटा प्रकार है जिसका उपयोग एक ही चर (variable) नाम के माध्यम से अनेक मानों (values) को संग्रहित करने के लिए किया जाता है। एरे में डेटा तत्वों (elements) का एक क्रमबद्ध (ordered) संग्रह होता है, जहाँ प्रत्येक तत्व एक ही प्रकार का होता है और उसे उसके सूचकांक (index/स्थिति) के द्वारा एक्सेस (अभिगम) किया जा सकता है।

एरे की महत्वपूर्ण विशेषताएँ निम्नलिखित हैं:

- एक एरे कई मान संग्रहीत करता है, लेकिन सभी मान एक ही डेटा प्रकार के होने चाहिए।
- एक एरे को मेमोरी में सन्निहित (contiguously) (एक निरंतर ब्लॉक में) संग्रहीत किया जाता है, जो उसके संचालन को तेज़ बनाता है।
- एक एरे में प्रत्येक तत्व को उसके सूचकांक (position number) द्वारा पहचाना जाता है। सूचकांक 0 से शुरू होता है। इसलिए पहला तत्व सूचकांक (इंडेक्स) 0 पर, दूसरा सूचकांक 1 पर और इसी प्रकार आगे होता है।

उदाहरण के लिए, 5 तत्वों वाले एक एरे पर विचार करें:

a = [21, 9, 29, 17, 30]

यहाँ एरे का पहला तत्व 21 है जिसका सूचकांक मान [0] है। दूसरा मान 9 है जिसका सूचकांक [1] है और इसी तरह आगे। अंतिम तत्व 30 है जिसका सूचकांक [4] है। यह पायथन में सूचियों (lists) के अनुक्रमण (indexing) के समान ही है।

1.2.1 NumPy एरे (NumPy Array)

NumPy एरे का उपयोग संख्यात्मक डेटा की सूचियों (lists), वेक्टर (vectors) और मैट्रिक्स (matrices) को संग्रहीत करने के लिए किया जाता है। NumPy लाइब्रेरी में NumPy एरे बनाने, उन्हें संचालित करने (manipulate) और रूपांतरित करने (transform) के लिए अनेक अंतर्निर्मित (built-in) फ़ंक्शन उपलब्ध होते हैं। पायथन भाषा में भी एक एरे डेटा संरचना मौजूद है, लेकिन वह NumPy एरे जितनी बहुउपयोगी (versatile), कुशल (efficient) और उपयोगी नहीं है। NumPy एरे को आधिकारिक तौर पर *ndarray* कहा जाता है लेकिन आमतौर पर इसे एरे के रूप में ही जाना जाता है।

1.2.2 NumPy एरे का महत्व

डेटा विश्लेषण के लिए NumPy का उपयोग करने के कई कारण हैं, जो निम्नलिखित हैं—

1. NumPy में बहुत बड़े एरे बनाए जा सकते हैं।
2. NumPy का उपयोग करके विभिन्न प्रकार की क्रियाएँ की जा सकती हैं, जैसे— रैखिक बीजगणित (linear algebra), छँटाई (sorting), खोज (searching) तथा रैंडम संख्याओं के साथ कार्य करना आदि।
3. NumPy का मुख्य ऑब्जेक्ट ndarray होता है, जो हमें समान प्रकार के डेटा तत्वों के n-आयामी (n-dimensional) एरे बनाने की सुविधा देता है।
4. NumPy एरे का आकार (size) निश्चित (fixed) होता है। आकार बदलने के लिए मूल एरे को हटाकर नया एरे बनाना पड़ता है।
5. NumPy लूप के उपयोग में अधिक दक्षता प्रदान करता है। उदाहरण के लिए: दो एरे को गुणा करने का एक सरल उदाहरण देखिए। पायथन में हम इस प्रकार लूप लिखते हैं:

```
c = [ ]
for i in range(len(a)):
c.append(a[i] * b[i])
```

यह कोड कार्य करता है, लेकिन यदि एरे a और b में लाखों संख्याएँ हों, तो यह प्रक्रिया बहुत धीमा हो जाएगा। NumPy में यही गुणा केवल एक पंक्ति में सरल और तेज़ी से किया जा सकता है:

```
c = a * b
```

यह स्पष्ट करता है कि NumPy अधिक कुशल प्रोग्रामिंग की सुविधा प्रदान करता है।

6. NumPy बहुत तेज़ है। इसका कोड संक्षिप्त (concise) और पढ़ने में आसान होता है। कम पंक्तियों के कारण त्रुटियों (bugs) की संभावना भी कम होती है।
7. NumPy को अन्य पायथन लाइब्रेरी—जैसे पांडस (Pandas), मैटप्लोटलिब (Matplotlib) और साइपी (SciPy) के साथ आसानी से एकीकृत (integrate) किया जा सकता है, जो विज्ञान और डेटा विश्लेषण के लिए उपयोगी हैं।

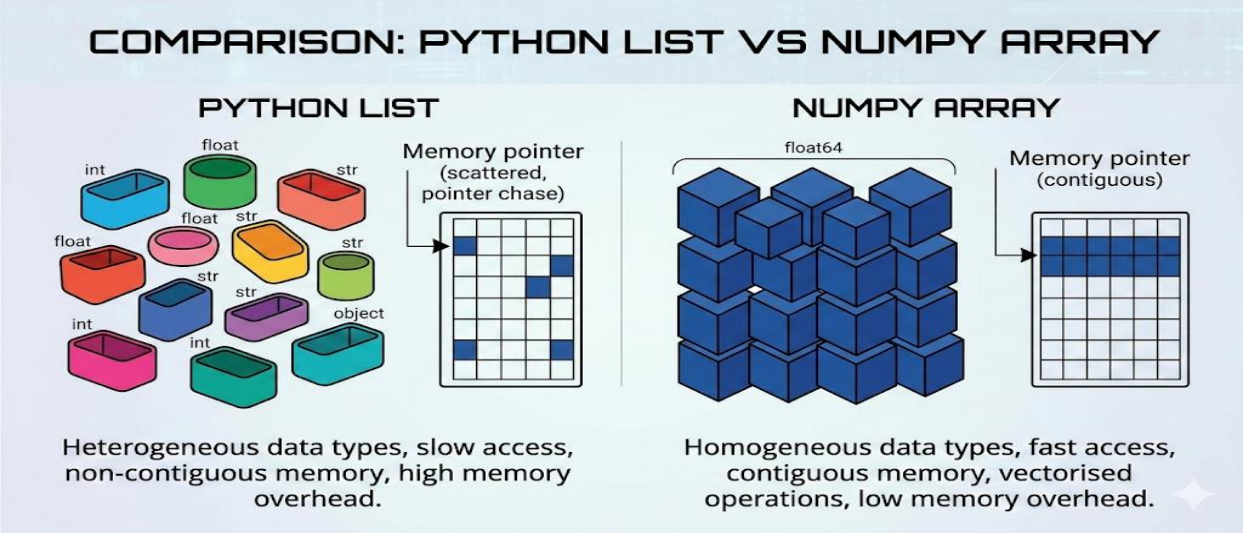
1.2.3 NumPy एरे बनाम पायथन सूचियाँ (Python Lists)

हालाँकि NumPy एरे और पायथन सूचियाँ (lists) दोनों ही तत्वों (elements) को संग्रहित करती हैं, फिर भी ये अलग-अलग डेटा संरचनाएँ (data structures) हैं। इनके बीच मुख्य अंतर निम्नलिखित हैं।

NumPy एरे	**पायथन सूची**
NumPy का उपयोग करने के लिए आपको NumPy लाइब्रेरी को इंस्टॉल और इम्पोर्ट करना आवश्यक है।	यह पायथन की एक अंतर्निर्मित (built-in) विशेषता है और हमेशा उपलब्ध रहती है।
NumPy एरे का आकार निश्चित (fixed) होता है। इसका आकार बदलने के लिए आपको नया एरे बनाना पड़ता है या मौजूदा एरे को पुनःनिर्धारित (overwrite) करना होता है।	आप सूची (list) में आसानी से तत्व जोड़ या हटा सकते हैं (जैसे append(), insert() आदि का उपयोग करके)।

NumPy एरे के सभी तत्व एक ही प्रकार (homogeneous) के होने चाहिए। उदाहरण के लिए, [1.2, 5.4, 2.7] सभी फ्लोट (float) प्रकार के हैं।	एक सूची में विभिन्न प्रकार (heterogeneous) के तत्व हो सकते हैं। उदाहरण के लिए, [1, 3.4, 'hello', 'a@']।
एरे में तत्व-स्तरीय (element-wise) क्रियाएँ समर्थित होती हैं। उदाहरण के लिए, यदि A1 एक एरे है, तो A1/3 लिखकर प्रत्येक तत्व को 3 से विभाजित किया जा सकता है।	सूचियाँ सीधे तौर पर तत्व-स्तरीय (element-wise) क्रियाओं का समर्थन नहीं करतीं। इसके लिए आपको लूप का उपयोग करना पड़ता है।
NumPy एरे कम मेमोरी का उपयोग करते हैं, क्योंकि इन्हें प्रत्येक तत्व के डेटा प्रकार की अतिरिक्त जानकारी संग्रहीत करने की आवश्यकता नहीं होती।	सूचियाँ अधिक मेमोरी लेती हैं, क्योंकि उन्हें प्रत्येक तत्व के डेटा प्रकार की जानकारी अलग-अलग संग्रहीत करनी पड़ती है।
NumPy एरे वेक्टराइज्ड (vectorised) ऑपरेशन्स का समर्थन करते हैं, अर्थात आप किसी क्रिया को पूरे एरे पर एक साथ लागू कर सकते हैं।	सूचियाँ वेक्टराइज्ड (vectorised) ऑपरेशन्स (संक्रियाओं) का समर्थन नहीं करतीं हैं।

चित्र 1.2, पायथन लिस्ट और NumPy एरे का विस्तृत तुलना ग्राफ के माध्यम से इस प्रकार है।



चित्र 1.2: पायथन लिस्ट और NumPy एरे की तुलना

1.2.4 NumPy का इंस्टॉलेशन (Installation of NumPy)

NumPy का उपयोग करने के लिए इसे इंस्टॉल करना आवश्यक है। यदि आप Anaconda का उपयोग कर रहे हैं, तो सामान्यतः NumPy पहले से ही उसमें शामिल होता है। यदि ऐसा नहीं है, तो आप अपने कंप्यूटर के कमांड प्रॉम्प्ट या टर्मिनल में pip कमांड का उपयोग करके इसे इंस्टॉल कर सकते हैं।

pip install numpy

1.3 NumPy एरे बनाना (Creating Numpy Array)

NumPy में मुख्य ऑब्जेक्ट ndarray होता है, जिसका उपयोग सजातीय (homogeneous) के बहुआयामी (multidimensional) एरे बनाने के लिए किया जाता है। "Homogeneous" का अर्थ है कि सभी तत्व एक ही प्रकार के होते हैं। NumPy एरे के तत्वों को गैर-ऋणात्मक पूर्णांकों (non-negative integers) के टपल (tuple) द्वारा इंडेक्स किया जाता है, जिसकी शुरुआत 0 से होती है। किसी एरे के आयामों (dimensions) की संख्या को NumPy में उसकी **axes (अक्ष)** कहा जाता है।

1.3.1 NumPy का उपयोग करके बेसिक एरे बनाना

एरे बनाने के कई तरीके हैं। सबसे बुनियादी तरीका एक पायथन सूची को NumPy एरे में बदलना है।

चरण 1. NumPy इम्पोर्ट करें (Import NumPy): NumPy का उपयोग करने के लिए सबसे पहले आपको इस लाइब्रेरी को इम्पोर्ट करना होता है। सामान्यतः इसे एक छोटा नाम (निकनेम) np दिया जाता है, ताकि पायथन में इसका उपयोग करना आसान हो जाए।

python

```
>>> import numpy as np
```

चरण 2. एक पायथन सूची या टपल बनाएँ: सबसे पहले, एक सामान्य पायथन सूची (list) या टपल (tuple) निर्धारित करें, जिसमें वे तत्व शामिल हों जिन्हें आप अपने एरे में रखना चाहते हैं।

python

```
>>> my_list = [1, 2, 3, 4, 5]
```

चरण 3. NumPy एरे में बदलें: पायथन सूची को NumPy एरे में बदलने के लिए np.array() फ़ंक्शन (function) का उपयोग करें।

python

```
>>> my_array = np.array(my_list)
```

चरण 4. एरे को प्रिंट या उपयोग करें: परिणामी एरे एक NumPy ndarray ऑब्जेक्ट है, जो संख्यात्मक संक्रियाओं (numerical operations) के लिए तैयार है।

python

```
>> print(my_array)
[1 2 3 4 5]
>>> print (type(my_array))
<class 'numpy.ndarray'>
```

© PSSCIVE

1.3.2 1-डी एरे बनाना (Creating a 1-D Array)

एक ही पंक्ति (row) में तत्वों वाला एरे एक-आयामी (1- डी) एरे कहलाता है। आइए, संख्याओं और एक स्ट्रिंग वाली सूची से एक 1-D एरे बनाते हैं।

उदाहरण: एक सूची से एक-आयामी एरे बनाएँ।

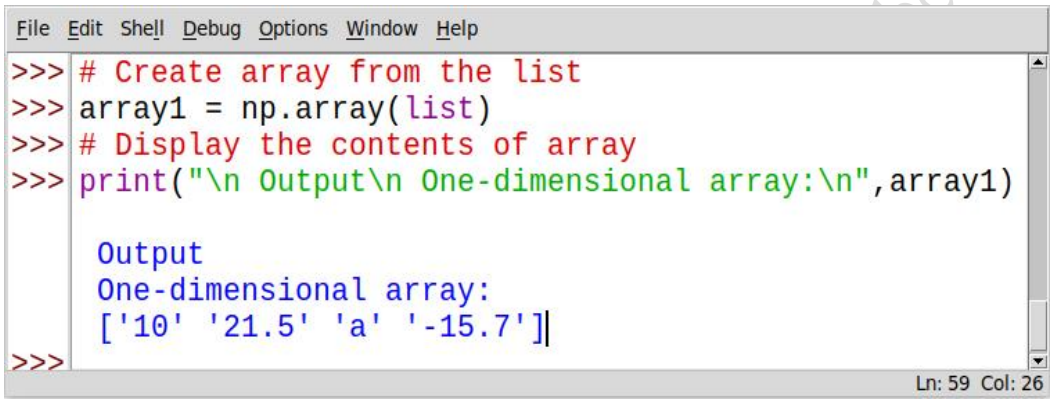
python

```
# Create a one-dimensional array from a list
list = [10,21.5,'a',-15.7]
# Create array from the list
array1 = np.array(list)
# Display the contents of array
print("\n Output\n One-dimensional array:\n",array1)
```

Output

One-dimensional array:

['10' '21.5' 'a' '-15.7']

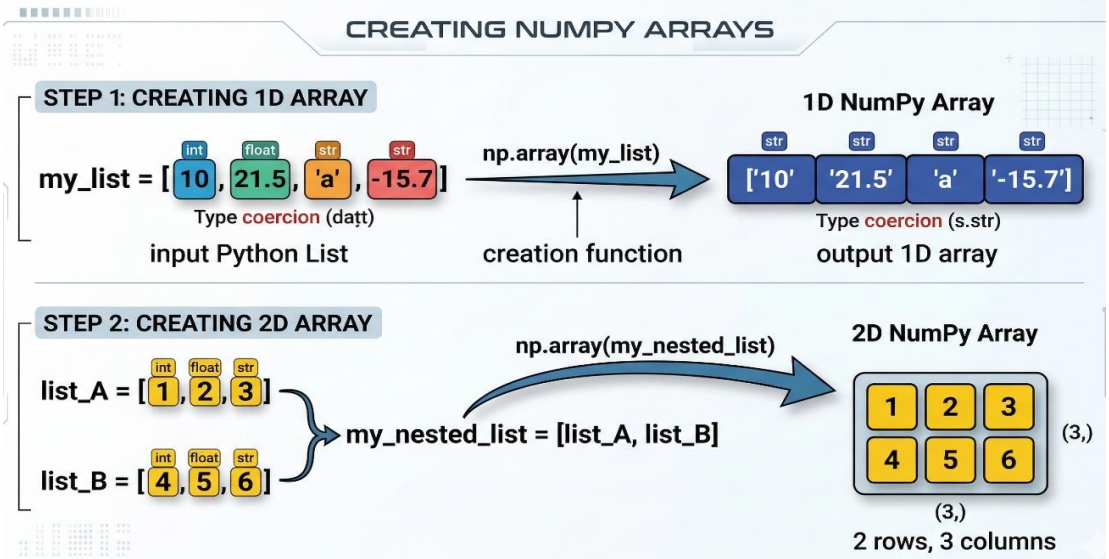


ध्यान दें कि सूची में एक स्ट्रिंग मान 'a' होने के कारण सभी संख्याएँ भी स्ट्रिंग में परिवर्तित हो गई हैं। ऐसा इसलिए होता है क्योंकि NumPy एरे के सभी तत्व एक ही प्रकार के होते है।

Note: A common mistake is forgetting to put square brackets around the list when passing it to array()

```
>>> a = np.array(1,2,3,4) # Incorrect way
>>> a = np.array([1,2,3,4]) # Correct way
```

© PSSCIVE Draft not to be Published



चित्र 1.3: NumPy एरे बनाना

1.3.3 2-डी एरे बनाना (Creating a 2-D Array)

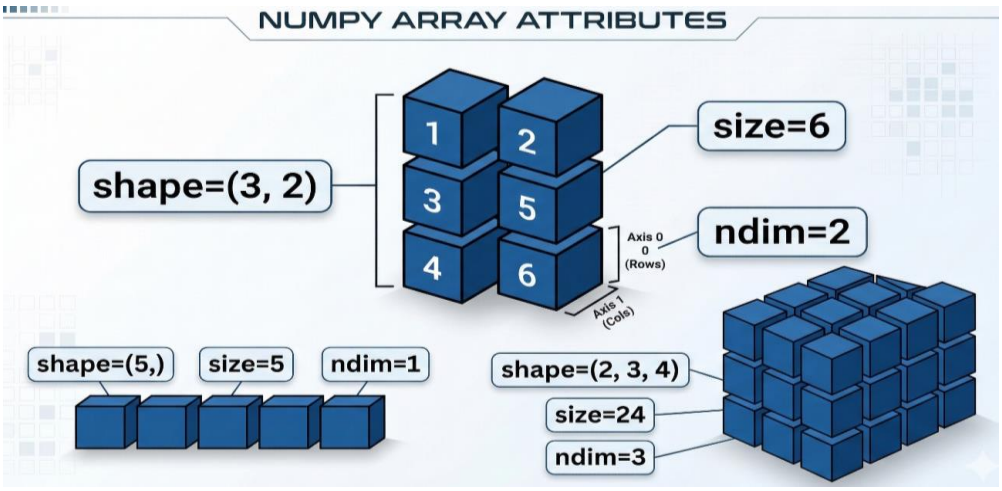
हम array() फ़ंक्शन में सूचियों की सूची (nested lists) को पास करके द्वि-आयामी (2-D) एरे बना सकते हैं।

उदाहरण: 2-डी एरे बनाना।

```
File  Edit  Shell  Debug  Options  Window  Help
>>> # Create 2-dimensional Array
>>> import numpy as np
>>> array2 = np.array([[3.4, 3], [- 3.4, 7], [0, -1]])
>>> # Display the contents of 2-D array
>>> print("\n Output: 2-dimensional array\n",array2)

 Output: 2-dimensional array
 [[ 3.4  3. ]
 [-3.4  7. ]
 [ 0.  -1. ]]
>>>
                                                Ln: 86  Col: 13
```

1.4 NumPy एरे के गुण (Attributes)

NumPy एरे में कई महत्वपूर्ण गुण होते हैं जो हमें एरे के बारे में जानकारी देते हैं।



चित्र 1.4: NumPy एरे के गुण

1.4.1 आकार (Shape) और माप (Size) के गुण

गुण (प्रॉपर्टी)	उदाहरण Python कोड के साथ

ndarray.shape: यह एक टपल (tuple) लौटाता है, जो एरे के आयामों (dimensions) को बताता है, जैसे पंक्तियों (rows) और स्तंभों (columns) की संख्या।	>>> a = np.array([[1, 2], [3, 4], [5, 6]])
	>>> print(a.shape)
	(3, 2) # This means the array has 3 rows and 2 columns.

ndarray.ndim: यह एरे के आयामों (axes) की संख्या को एक पूर्णांक (integer) के रूप में बताता है। 1-D एरे के लिए ndim = 1, 2-D एरे के लिए ndim = 2 और इसी प्रकार आगे।	>>> a1 = np.array([1, 2, 3])
	>>> print(a1.ndim)
	1
	>>> a2 = np.array([[1, 2], [3, 4]])
	>>> print(a2.ndim)
	2

ndarray.size: यह एरे में मौजूद कुल तत्वों (elements) की संख्या बताता है।	>>> a = np.array([[1, 2], [3, 4]])
	>>> print(a.size)
	4

ndarray.itemsize: यह प्रत्येक तत्व का आकार (बाइट्स में) बताता है। उदाहरण के लिए, यदि किसी एरे का डेटा प्रकार int32 है, तो प्रत्येक तत्व 32 बिट (अर्थात 4 बाइट, 32/8 = 4) का होता है।	>>> a = np.array([1, 2, 3], dtype=np.int32)
	>>> print(a.itemsize)
	4

ndarray.nbytes: यह एरे द्वारा उपयोग की गई कुल मेमोरी (बाइट्स में) बताता है। यह size × itemsize के बराबर होता है।	>>> a = np.array([1, 2, 3])
	>>> print(a.nbytes)
	24 # For a 64-bit system, each int is 8 bytes, so 3*8 = 24.

1.4.2. डेटा प्रकार के गुण (प्रॉपर्टी)

गुण (प्रॉपर्टी)	उदाहरण Python कोड के साथ

ndarray.dtype: यह एरे के तत्वों (elements) का डेटा प्रकार (data type) बताता है।	>>> a = np.array([1, 2, 3])
	>>> print(a.dtype)
	int64

ndarray.astype(dtype): इस फ़ंक्शन का उपयोग एरे के तत्वों के डेटा प्रकार को बदलने के लिए किया जाता है।	>>> a = np.array([4.3, 5.4])
	>>> int_a = a.astype(int)
	>>> print(int_a)
	[4 5] # The floats are converted to integers by truncating the decimal part.

1.4.3. गणितीय और तार्किक गुण (Mathematical and Logical Properties)

© PSSCIVE Draft Study Material Not to be Published

गुण (प्रॉपर्टी)	उदाहरण Python कोड के साथ

तत्व-स्तरीय क्रियाएँ (Element-wise Operations): NumPy एरे आपको प्रत्येक तत्व पर एक साथ क्रियाएँ करने की सुविधा देते हैं।	>>> a = np.array([1, 2, 3])
	>>> print(a * 2)
	[2 4 6] # Each element is multiplied by 2.

ndarray.all() : यदि एरे के सभी तत्व गैर-शून्य (non-zero) या सत्य (True) हैं, तो यह सत्य (True) लौटाता है।	>>> a = np.array([1, 2, 3])
	>>> print(a.all())
	True

ndarray.any() : यदि कम से कम एक तत्व गैर-शून्य (non-zero) या सत्य (True) है, तो यह सत्य (True) लौटाता है।	>>> a = np.array([0, 0, 1])
	>>> print(a.any())
	True

1.4.4. पुनः आकार देना (Reshaping) और दृश्य (Views)

गुण (प्रॉपर्टी)	उदाहरण Python कोड के साथ

ndarray.reshape(shape) : यह फ़ंक्शन उसी डेटा के साथ एक नया एरे लौटाता है, लेकिन उसका आकार (shape) अलग होता है। कुल तत्वों (elements) की संख्या समान रहनी चाहिए।	>>> a = np.array([1, 2, 3, 4])
	>>> reshaped = a.reshape(2, 2)
	>>> print(reshaped)
	[[1 2]
	 [3 4]]

ndarray.ravel() : यह फ़ंक्शन बहु-आयामी (multi-dimensional) एरे को समतल (flatten) करके एक 1-डी (1D) एरे में बदल देता है।	>>> a = np.array([[1, 2], [3, 4]])
	>>> print(a.ravel())
	[1 2 3 4]

1.4.5. कॉपी बनाम व्यू (Copy vs View)

गुण (प्रॉपर्टी)	उदाहरण Python कोड के साथ

ndarray.view() : यह एक नया एरे ऑब्जेक्ट बनाता है, जो उसी डेटा को दर्शाता (view) है। यदि आप इस व्यू में डेटा बदलते हैं, तो मूल (original) एरे में भी प्रभावित होता है।	# Original array
	a = np.array([10, 20, 30, 40])
	
	# Create a view
	b = a.view()
	
	# Modify the view
	b[0] = 100
	
	print("Original array:", a)
	print("View array:", b)

© PSSCIVE Draft Study Material Not Published

	Output : Original array: [100 20 30 40] View array: [100 20 30 40]

ndarray.copy() : यह डेटा की एक प्रति (copy) के साथ नया एरे बनाता है। कॉपी में किए गए परिवर्तन मूल एरे को प्रभावित नहीं करते हैं।	# Original array
	a = np.array([10, 20, 30, 40])
	
	# Create a copy
	b = a.copy()
	
	# Modify the copy
	b[0] = 100
	
	print("Original array:", a)
	print("Copied array:", b)
	
	Output : Original array: [10 20 30 40]
	Copied array: [100 20 30 40]

1.5 एरे के तत्वों तक पहुँच (Accessing Array Elements)

NumPy एरे के भीतर तत्वों तक पहुँच इंडेक्सिंग (Indexing) और स्लाइसिंग (Slicing) का उपयोग करके की जाती है।

1.5.1 अनुक्रमण या इंडेक्सिंग (Indexing)

अनुक्रमण या इंडेक्सिंग का अर्थ है किसी एक तत्व (एलिमेंट) को उसके स्थिति संख्या (पोजीशन नंबर) से एक्सेस करना।

1D एरे के लिए, हम वर्गाकार कोष्ठक [] के अंदर एक सूचकांक (इंडेक्स) का उपयोग करते हैं। याद रखें, सूचकांक हमेशा 0 से शुरू होता है।

उदाहरण: 1D एरे के तत्वों तक पहुँच प्राप्त करना।

```
# Accessing elements of a 1D array
import numpy as np
arr = np.array([10, 20, 30, 40, 50])
print("Output:")
print("First element of array:", arr[0])
print("Third element of array:", arr[2])
```

Output:

First element of array: 10

Third element of array: 30

© PSSCIVE Not be Published

2D एरे के लिए, हम दो सूचकांकों (इंडेक्स) का उपयोग करते हैं: [पंक्ति (row), स्तंभ (column)]। पंक्ति और स्तंभ दोनों सूचकांक 0 से शुरू होते हैं।

उदाहरण: 2D एरे के तत्वों तक पहुँच प्राप्त करना।

```
# Accessing elements of a 2D array
import numpy as np
matrix = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print("Output:")
print("\nElement in the first row, second column:", matrix[0, 1])
print("\nElement in the second row, third column:", matrix[1, 2])
```

Output:

Element in the first row, second column: 2

Element in the second row, third column: 6

ऋणात्मक इंडेक्सिंग (Negative indexing) आपको एरे के अंत (end) से तत्वों तक पहुँचने की सुविधा देती है। इंडेक्स -1 अंतिम तत्व को संदर्भित करता है, -2 दूसरे अंतिम तत्व को और इसी प्रकार आगे।

उदाहरण: ऋणात्मक इंडेक्सिंग का उपयोग करके एरे के तत्वों तक पहुँच प्राप्त करना।

```
# Accessing elements of an array using negative indexing.
import numpy as np
        arr = np.array([1, 2, 3, 4, 5])
        print("Output:")
        print("Last element:", arr[-1])
        print("Third element from the end:", arr[-3])
```

Output:

Last element: 5

Third element from the end: 3

1.5.2. स्लाइसिंग (Slicing)

स्लाइसिंग का उपयोग एरे के किसी भाग को निकालने (extract) के लिए किया जाता है। इसका सामान्य प्रारूप start:stop:step होता है। यदि आप इनमें से कोई मान नहीं देते हैं, तो यह अपने डिफ़ॉल्ट मान (start=0, stop=end of array, step=1) ले लेता है।

1D एरे के लिए:

उदाहरण: स्लाइसिंग का उपयोग करके तत्वों तक पहुँच प्राप्त करना।

© PSSCIVE Draft Study Material Not be Published

```
# Accessing elements using slicing
import numpy as np
        arr = np.array([10, 20, 30, 40, 50])
        print("Output:")
        # Elements from index 1 up to, but not including, index 4
        print("Elements from index 1 to 3:", arr[1:4])
```

Output:

Elements from index 1 to 3: [20 30 40]

2D एरे के लिए: हम पंक्तियों (rows) के लिए एक स्लाइस और स्तंभों (columns) के लिए एक स्लाइस प्रदान करते हैं, जिन्हें अल्पविराम या कॉमा (,) द्वारा अलग किया जाता है।

उदाहरण: स्लाइसिंग का उपयोग करके 2D एरे के तत्वों तक पहुँच प्राप्त करना।

```
# Accessing elements of a 2D array using slicing
import numpy as np
        matrix = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
        print("Output:")
        print("Original matrix:\n", matrix)
        # Select first two rows (0:2) and second and third columns (1:3)
        print("\nFirst two rows, second and third columns:\n", matrix[0:2, 1:3])
```

Output:

Original matrix:

[[1 2 3]

[4 5 6]

[7 8 9]]

First two rows, second and third columns:

[[2 3]

[5 6]]

संपूर्ण पंक्तियों या स्तंभों तक पहुँच (Accessing entire rows or columns): किसी एक आयाम (dimension) के सभी तत्वों का चयन करने के लिए कोलन (:) का उपयोग किया जाता है।

उदाहरण: किसी संपूर्ण पंक्ति या स्तंभ तक पहुँच प्राप्त करना।

© PSSCIVE Draft Study Material Not published

```
# Accessing entire rows or columns of a 2D array using slicing
import numpy as np
matrix = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print("Output:")
print("Original matrix:\n", matrix)
# All columns (:) of the second row (index 1)
print("\nEntire second row:\n", matrix[1, :])
All rows (:) of the third column (index 2)
print("\nEntire third column:\n", matrix[:, 2])
```

Output:

```
Original matrix:
[[1 2 3]
[4 5 6]
[7 8 9]]
Entire second row:
[4 5 6]
Entire third column:
[3 6 9]
```

1.6 NumPy एरे बनाने के अन्य तरीके (OTHER WAYS OF CREATING NUMPY ARRAYS)

सूचियों (lists) से एरे बनाने के अलावा, NumPy में कुछ विशेष अंतर्निर्मित (built-in) फ़ंक्शन भी होते हैं, जिनकी सहायता से विशिष्ट मानों वाले एरे बनाए जा सकते हैं।

1.6.1 अंतर्निर्मित (Intrinsic) NumPy एरे निर्माण फ़ंक्शन

np.zeros(shape, dtype): यह 0 से भरा हुआ एरे बनाता है। डिफ़ॉल्ट रूप से इसका डेटा प्रकार फ्लोट (float) होता है।

```
import numpy as np
# Create a 2x3 array of zeros (float by default)
zeros_array = np.zeros((2, 3))
print("Array of zeros (float):\n", zeros_array)
# Create a 3x2 array of zeros, specifying integer type
```

Draft Study Material Not be Published

```
zeros_int = np.zeros((3, 2), dtype=int)
print("\nArray of zeros (integer):\n", zeros_int)
```

Output:

```
Array of zeros (float):
[[0. 0. 0.]
[0. 0. 0.]]
Array of zeros (integer):
[[0 0]
[0 0]
[0 0]]
```

np.ones(shape, dtype) : यह 1 से भरा हुआ एक एरे बनाता है।

```
import numpy as np
ones_array = np.ones((3, 2))
print("Array of ones:\n", ones_array)
```

Output:

```
Array of ones:
[[1. 1.]
[1. 1.]
[1. 1.]]
```

np.full(shape, fill_value, dtype) : यह ऐसा एरे बनाता है जिसमें प्रत्येक तत्व दिए गए fill_value के बराबर होता है।

```
import numpy as np
full_array = np.full((3, 2), 5)
print("Array filled with 5:\n", full_array)
```

Output:

```
Array filled with 5:
[[5 5]
[5 5]
[5 5]]
```

© PSSCE Draft Study Material Not to be Published

np.arange(start, stop, step, dtype) : यह संख्याओं के एक क्रम (sequence) वाला एरे बनाता है, जो पायथन के range() फ़ंक्शन के समान होता है।

```
import numpy as np
# Create an array from 0 to 10 (exclusive), stepping by 2
range_array = np.arange(0, 10, 2)
print("Array from arange:\n", range_array)
```

Output:

```
Array from arange:
[0 2 4 6 8]
```

np.linspace(start, stop, num, dtype) : यह ऐसा एरे बनाता है जिसमें start और stop के बीच (stop मान सहित) समान अंतराल (evenly spaced) वाले निर्धारित संख्या (num) के तत्व होते हैं।

```
import numpy as np
# Create 5 evenly spaced numbers between 0 and 2
linear_array = np.linspace(0, 2, 5)
print("Array from linspace:\n", linear_array)
```

Output:

```
Array from linspace:
[0. 0.5 1. 1.5 2. ]
```

Draft Study Material Not Published

1.6.2 रेंडम संख्याओं का उपयोग करके एरे बनाना (Generating arrays using random numbers)

NumPy **में ऐसे फ़ंक्शन भी होते हैं, जिनकी सहायता से रेंडम संख्याओं से भरे एरे बनाए जा सकते हैं।**

np.random.rand(d0, d1, ...) : यह दिए गए आकार (shape) का एक एरे बनाता है, जिसमें 0 और 1 के बीच की रेंडम संख्याएँ होती हैं।

np.random.randint(low, high, size) : यह दिए गए आकार (size) का एक एरे बनाता है, जिसमें low (समावेशी) और high (बहिष्कृत) के बीच के रेंडम पूर्णांक (integers) होते हैं।

सारांश (Summary)

- NumPy (न्यूमेरिकल पायथन) एक पायथन लाइब्रेरी है, जिसका उपयोग वैज्ञानिक गणना (scientific computing) और संख्यात्मक डेटा के साथ कार्य करने के लिए किया जाता है। यह "एरे" नामक एक विशेष डेटा संरचना प्रदान करती है, जो एक ही चर में अनेक मानों को संग्रहित करती है।

- NumPy एरे, पायथन सूचियों (lists) की तुलना में तेज़ और अधिक कुशल होते हैं, क्योंकि इनके तत्व निरंतर (continuous) मेमोरी लोकेशनों में संग्रहीत होते हैं।
- NumPy एरे के सभी तत्व एक ही डेटा प्रकार के होने चाहिए, जबकि पायथन सूचियों में विभिन्न प्रकार के तत्व हो सकते हैं।
- NumPy एरे एक-आयामी (1-D), द्वि-आयामी (2-D) या बहुआयामी (multi-dimensional) हो सकते हैं, जो डेटा संरचना पर निर्भर करता है।
- NumPy वेक्टराइज्ड (vectorised) ऑपरेशन्स का समर्थन करता है, जिससे गणितीय गणनाएँ एरे के सभी तत्वों पर एक साथ की जा सकती हैं।
- एरे पायथन सूचियों को परिवर्तित करके या NumPy के अंतर्निर्मित फ़ंक्शन्स जैसे array(), zeros(), ones(), arange() और linspace() का उपयोग करके बनाए जा सकते हैं।
- NumPy एरे के महत्वपूर्ण गुणों में shape, size, आयामों की संख्या (ndim) और डेटा प्रकार (dtype) शामिल हैं।
- एरे के तत्वों तक पहुँच इंडेक्सिंग (indexing) और स्लाइसिंग (slicing) के माध्यम से की जा सकती है, जिससे विशेष मान या एरे के भाग प्राप्त किए जा सकते हैं।
- अपनी गति, दक्षता और शक्तिशाली क्रियाओं के कारण NumPy का व्यापक उपयोग डेटा विश्लेषण, मशीन लर्निंग और वैज्ञानिक अनुप्रयोगों में किया जाता है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. कौन-सी पायथन लाइब्रेरी मुख्य रूप से संख्यात्मक गणना (numerical computing) और एरे के साथ कार्य करने के लिए उपयोग की जाती है? (अ) पांडस (ब) नमपाई (स) टिंकर (द) टर्टल
2. NumPy का पूरा नाम क्या है? (अ) न्यूमेरिकल पायथन (ब) न्यू पायथन (स) न्यूमेरिक प्रोग्राम (द) नेटवर्क पायथन
3. NumPy में डेटा संग्रहित करने के लिए उपयोग किया जाने वाला मुख्य ऑब्जेक्ट क्या कहलाता है? (अ) list (ब) ndarray (स) tuple (द) set
4. NumPy एरे में सभी तत्व किस प्रकार के होने चाहिए? (अ) विभिन्न प्रकार (Different types) (ब) समान डेटा प्रकार (Same data type) (स) रेंडम प्रकार (Random types) (द) केवल स्ट्रिंग प्रकार (String type only)
5. पायथन सूची से NumPy एरे बनाने के लिए किस फ़ंक्शन का उपयोग किया जाता है? (अ) np.list() (ब) np.array() (स) np.create() (द) np.make()
6. एरे के पहले तत्व का इंडेक्स (सूचकांक) क्या होता है? (अ) 0 (ब) 1 (स) -1 (द) 2
7. कौन-सा फ़ंक्शन शून्यों (zeros) से भरा हुआ एरे बनाता है? (अ) np.ones() (ब) np.full() (स) np.zeros() (द) np.arange()

8. NumPy एरे में तत्वों की कुल संख्या कौन-सा एट्रिब्यूट (गुण) बताता है? (अ) shape (ब) size (स) ndim (द) dtype

ब. रिक्त स्थान भरें

1. NumPy एरे तत्वों को _____ मेमोरी ब्लॉक्स में संग्रहीत करते हैं।
2. _____ एट्रिब्यूट एरे के आयामों (dimensions) की संख्या बताता है।
3. _____ फ़ंक्शन का उपयोग एरे को नए आकार (structure) में बदलने के लिए किया जाता है।
4. _____ फ़ंक्शन 1 से भरा हुआ एरे बनाता है।
5. _____ एट्रिब्यूट एरे के तत्वों का डेटा प्रकार बताता है।

स. सत्य या असत्य

1. NumPy एरे एक ही एरे में विभिन्न डेटा प्रकारों के तत्वों को संग्रहीत कर सकते हैं।
2. NumPy एरे संख्यात्मक गणनाओं के लिए पायथन सूचियों (lists) की तुलना में तेज़ होते हैं।
3. reshape() फ़ंक्शन किसी एरे का आकार (shape) बदलता है।
4. ऋणात्मक इंडेक्सिंग (Negative Indexing) का उपयोग किसी एरे के अंत से तत्वों तक पहुँचने के लिए किया जाता है।
5. np.zeros() फ़ंक्शन 1 से भरा एरे बनाता है।

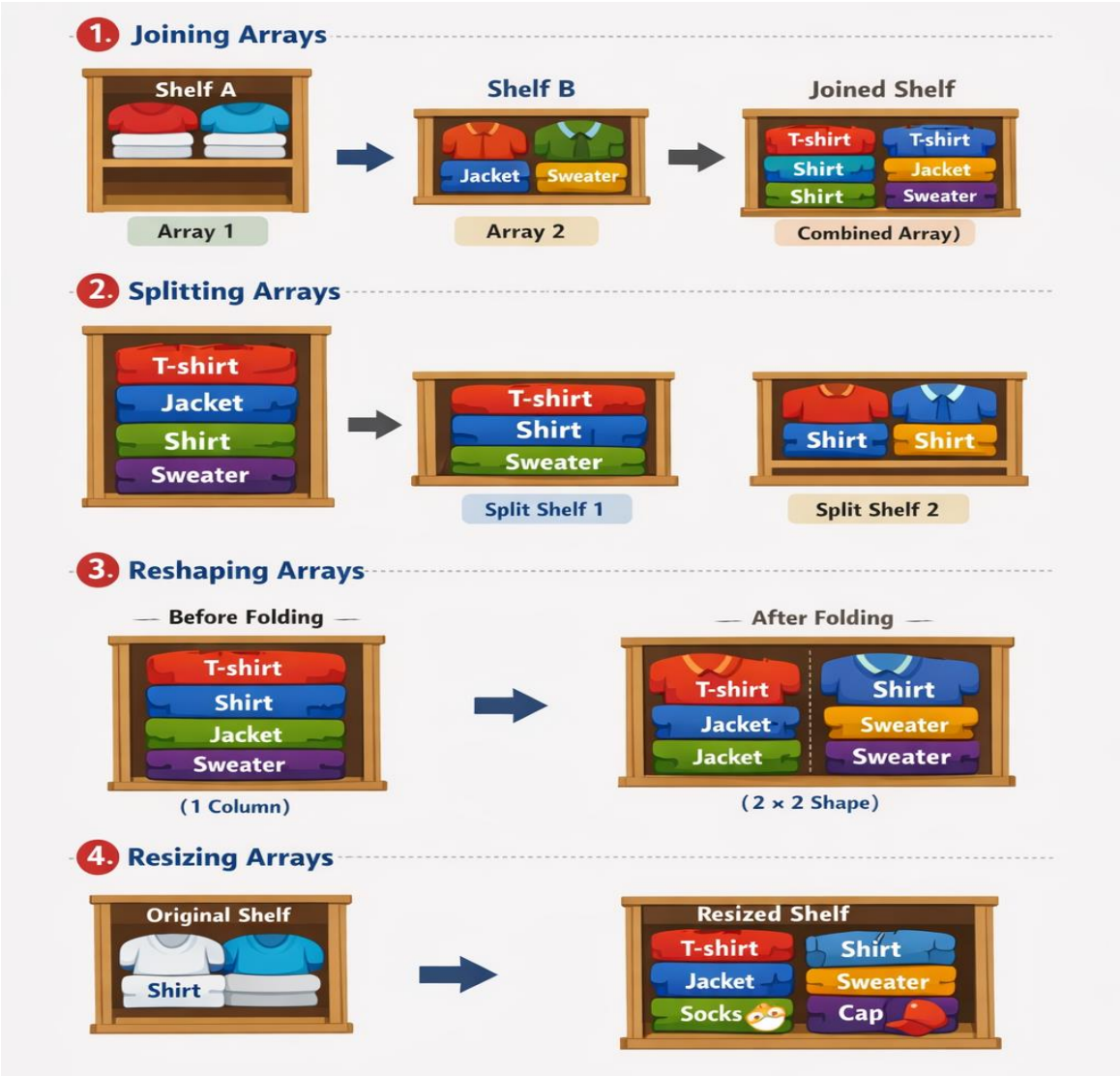
द. लघु उत्तरीय प्रश्न

1. NumPy क्या है?
2. NumPy एरे को परिभाषित कीजिए।
3. पायथन सूची और NumPy एरे के बीच एक अंतर लिखिए।
4. NumPy एरे में इंडेक्सिंग (indexing) क्या है?
5. NumPy एरे में स्लाइसिंग (slicing) क्या है?
6. गुण shape क्या दर्शाता है?
7. NumPy एरे का उपयोग करने का एक लाभ लिखिए।
8. NumPy एरे बनाने के लिए उपयोग किए जाने वाले किन्हीं दो फ़ंक्शन के नाम लिखिए।

सत्र 2. NumPy का उपयोग करके एरे का संचालन

(Array Manipulation using NumPy)

कल्पना कीजिए कि आप अपनी अलमारी में कपड़ों को व्यवस्थित कर रहे हैं। कभी आप उन्हें एक साथ एक ढेर में रखते हैं (जोड़ना), कभी उन्हें अलग-अलग शेल्फ़ में बाँट देते हैं (विभाजित करना), कभी आप उन्हें अलग तरीके से मोड़ते हैं (पुनःआकार देना) और कभी जगह बनाने के लिए उनका आकार बदलते हैं (रीसाइज़ करना)। इसी प्रकार, NumPy हमें डेटा को बेहतर ढंग से प्रबंधित करने के लिए एरे को जोड़ने, विभाजित करने, पुनःआकार देने या आकार बदलने की सुविधा देता है।



चित्र 2.1: NumPy में एरे प्रबंधन

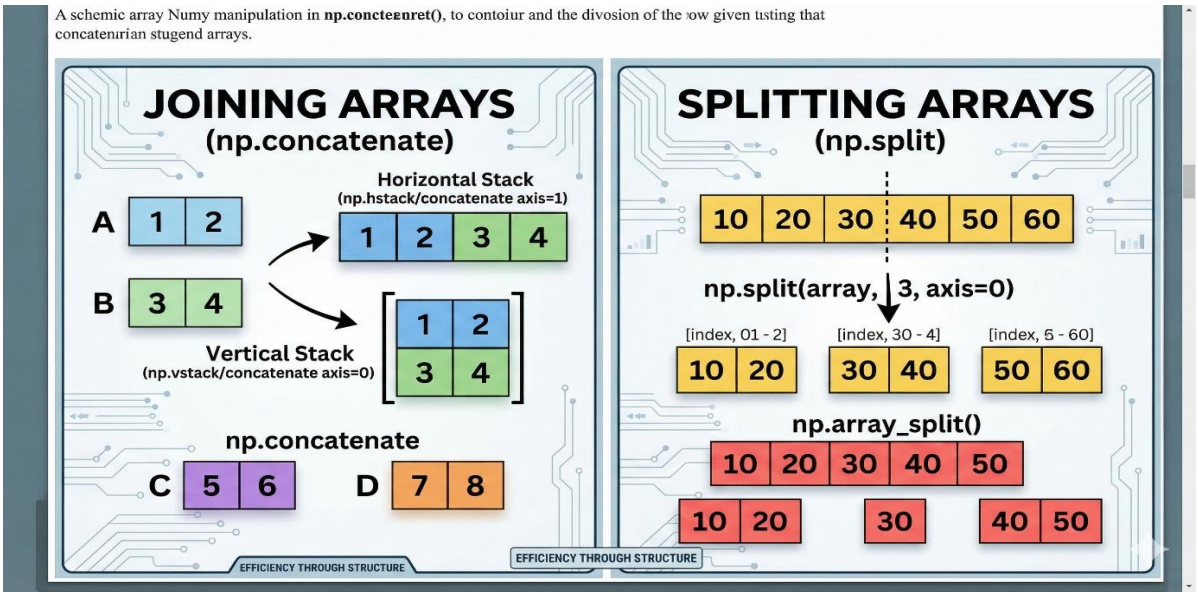
इस सत्र में आप एरे को संचालित (manipulate) करने के विभिन्न तरीकों को समझेंगे—जैसे जोड़ना (concatenate, vstack, hstack, dstack), विभाजित करना (split, array_split, vsplit, hsplit, dsplit), एरे को नए रूप में पुनःआकार देना (reshaping), बहुआयामी एरे को 1-डी में समतल करना (flattening), एरे का आकार बदलना (resizing—विस्तार या संक्षेप) तथा आयामों में परिवर्तन करना (axes जोड़ना या हटाना) शामिल है।

2.1 परिचय

NumPy, पायथन में वैज्ञानिक गणनाओं के लिए एक आधारभूत (foundation) लाइब्रेरी है। इसमें एरे के साथ कार्य करने के लिए कई उपयोगी फ़ंक्शन उपलब्ध होते हैं। डेटा विश्लेषण के दौरान हमें अक्सर अपने डेटा की संरचना (structure) को बदलने की आवश्यकता होती है। इस सत्र में हम एरे संचालन या हेरफेर (array manipulation) की विभिन्न तकनीकों पर चर्चा करेंगे।

2.2 एरे को जोड़ना और विभाजित करना (Joining and Splitting Arrays)

जोड़ना (Joining) और विभाजित करना (Splitting), NumPy की दो मूलभूत क्रियाएँ हैं। जोड़ने की प्रक्रिया में कई एरे को मिलाकर एक एरे बनाया जाता है, जबकि विभाजित करने में एक एरे को छोटे-छोटे कई एरे में बाँट दिया जाता है।



चित्र 2.2: NumPy में एरे प्रबंधन

2.2.1 एरे का जोड़ना (Joining of Arrays)

जोड़ने (या संयोजन) का मतलब है दो या दो से अधिक एरे को एक ही एरे में मिलाना।

np.concatenate(): इस फ़ंक्शन का उपयोग दो या दो से अधिक एरे को किसी मौजूदा अक्ष के साथ जोड़ने के लिए किया जाता है। डिफ़ॉल्ट रूप से, यह उन्हें axis=0 (पंक्ति-वार) के साथ जोड़ता है।

उदाहरण 2.1: दो एरे का पंक्तियों के साथ संयोजन (axis=0)

```
# Concatenation of Arrays along rows
import numpy as np

a1 = np.array([[1, 2], [3, 4]])
a2 = np.array([[5, 6], [7, 8]])

# Join arrays along rows (axis=0)
```

```
a3 = np.concatenate((a1, a2), axis=0)


print("Output")
print("Array 1:\n", a1)
print("Array 2:\n", a2)
print("Arrays after joining along rows:\n", a3)
```

Output

```
Array 1:
[[1 2]
[3 4]]
Array 2:
[[5 6]
[7 8]]
Arrays after joining along rows:
[[1 2]
[3 4]
[5 6]
[7 8]]
```

आउटपुट में, a2 की पंक्तियाँ a1 की पंक्तियों के नीचे जोड़ दी गई हैं।

उदाहरण 2.2: दो एरे का स्तंभों के साथ संयोजन (axis=1)

```
# Join arrays along columns (axis=1)
a4 = np.concatenate((a1, a2), axis=1)


print("Arrays after joining along columns:\n", a4)
```

Output

```
Arrays after joining along columns:
[[1 2 5 6]
[3 4 7 8]]
```

इस आउटपुट में, a2 के स्तंभ a1 के स्तंभों के दाईं ओर जोड़ दिए गए हैं।

© PSSCIVE Draft Study Material Not be Published

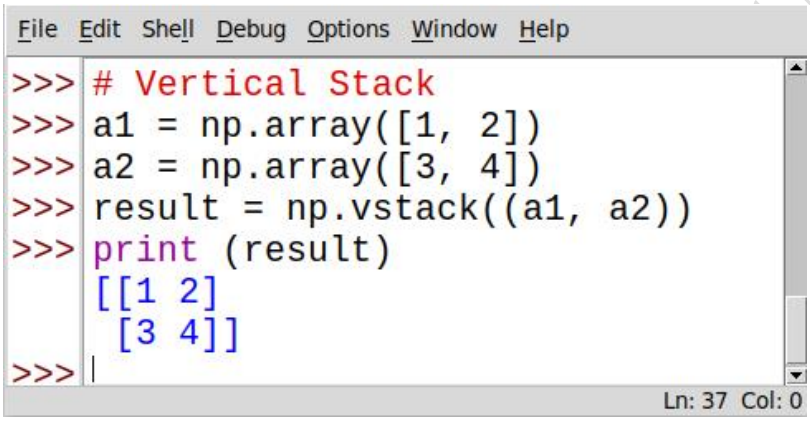
np.vstack() (वर्टिकल स्टैक) : यह फ़ंक्शन एरे को लंबवत (पंक्ति-वार) रखता है। यह axis=0 के साथ concatenate() के समान है।

उदाहरण 2.3:

```
# Vertical Stack
import numpy as np
a1 = np.array([1, 2])
a2 = np.array([3, 4])
result = np.vstack((a1, a2))
print(result)
```

Output

```
[[1 2]
[3 4]]
```



np.hstack() (हॉरिजॉन्टल स्टैक) : यह फ़ंक्शन एरे को क्षैतिज रूप से (स्तंभ-वार) रखता है। 1D एरे के लिए, यह उन्हें अंत से अंत तक जोड़ता है।

उदाहरण 2.4:

```
# Horizontal Stack
import numpy as np
a1 = np.array([1, 2])
a2 = np.array([3, 4])
result = np.hstack((a1, a2))
print(result)
```

Output

```
[1 2 3 4]
```

© PSSCIVE Not be Published

```
File Edit Shell Debug Options Window Help
>>> # Horizontal Stack
>>> a1 = np.array([1, 2])
>>> a2 = np.array([3, 4])
>>> result = np.hstack((a1, a2))
>>> print (result)
    [1 2 3 4]
>>> |
                                    Ln: 58 Col: 0
```

np.column_stack() : यह फ़ंक्शन 1D एरे को 2D एरे में स्तंभों के रूप में रखता है।

उदाहरण 2.5:

```
# Column Stack
import numpy as np
a1 = np.array([1, 2])
a2 = np.array([3, 4])
result = np.column_stack((a1, a2))
print(result)
```

Output

```
[[1 3]
[2 4]]
```

2.2.2 एरे को विभाजित करना (Splitting Arrays)

विभाजित करना जोड़ने के विपरीत है। यह एक एरे को कई छोटे एरे में विभाजित करता है।

np.split(): यह फ़ंक्शन किसी एरे को निर्दिष्ट अक्ष के साथ कई समान भागों में विभाजित करता है। यदि एरे को समान रूप से विभाजित नहीं किया जा सकता है, तो यह एक त्रुटि उत्पन्न करता है।

उदाहरण 2.6: एक एरे को 3 समान भागों में विभाजित करें।

```
# Split an Array
import numpy as np
a = np.array([1, 2, 3, 4, 5, 6])
# Split into 3 equal parts
result = np.split(a, 3)
print(result)
```

Output

© Study Material Not be Publ

```
[array([1, 2]), array([3, 4]), array([5, 6])]
```

np.array_split(): यह फ़ंक्शन भी एक एरे को विभाजित करता है लेकिन भागों को असमान होने की अनुमति देता है।

उदाहरण 2.7: 5 तत्वों वाले एक एरे को 3 भागों में विभाजित कीजिए।

```
# Unequal splitting of an array
import numpy as np
a = np.array([1, 2, 3, 4, 5])
# Split into 3 parts (unequal)
result = np.array_split(a, 3)
print(result)
```

Output

```
[array([1, 2]), array([3, 4]), array([5])]
```

np.vsplit(): किसी एरे को लंबवत (पंक्ति-वार) विभाजित करता है। यह 2D या उच्च-आयामी एरे पर काम करता है।

उदाहरण 2.8: 2D एरे का लंबवत विभाजन।

```
# Vertical splitting of an array.
import numpy as np
a = np.array([[1, 2], [3, 4], [5, 6]])
result = np.vsplit(a, 3)
print(result)
```

Output

```
[array([[1, 2]]), array([[3, 4]]), array([[5, 6]])]
```

np.hsplit() : किसी एरे को क्षैतिज (स्तंभ-वार) विभाजित करता है।

उदाहरण 2.9: 2D एरे का क्षैतिज विभाजन।

```
# Horizontal splitting of an array
import numpy as np
a = np.array([[1, 2, 3], [4, 5, 6]])
result = np.hsplit(a, 3)
print(result)
```

SCERT Draft Study Material Not be Published

Output

```
[array([[1],
[4]]), array([[2],
[5]]), array([[3],
[6]])]
```

2.3 एरे का पुनः आकार देना (Reshaping Arrays)

पुनः आकार देने का मतलब है किसी एरे के आकार (पंक्तियों और स्तंभों की संख्या) को बदलना। तत्वों की कुल संख्या समान रहनी चाहिए।

np.reshape() : यह फ़ंक्शन नए आकार के साथ एक नया एरे बनाता है।

उदाहरण 2.10: 1D एरे को 2D एरे में पुनः आकार देना।

```
# Reshaping an array
import numpy as np
a = np.array([1, 2, 3, 4, 5, 6])
# Reshape into a 2x3 array
reshaped = np.reshape(a, (2, 3))
print(reshaped)
```

Output

```
[[1 2 3]
[4 5 6]]
```

आप आयामों में से एक के लिए -1 का भी उपयोग कर सकते हैं, और NumPy स्वचालित रूप से सही आकार की गणना करेगा।

```
# Infer one dimension using -1
reshaped = np.reshape(a, (-1, 2))
print(reshaped)
```

Output

```
[[1 2]
[3 4]
[5 6]]
```

एरे का समतलन (Flattening Arrays) : एक बहु-आयामी एरे को 1D एरे में बदलना।

© PSSCIVE Draft Study Material Not be

ndarray.ravel() : यदि संभव हो तो एरे का एक समतल (flattened) व्यू (view) लौटाता है।

ndarray.flatten() : एरे की एक समतल (flattened) कॉपी (copy) लौटाता है।

उदाहरण 2.11: एरे का समतलन।

```
# Flattening an array
import numpy as np
a = np.array([[1, 2], [3, 4]])
print(a.ravel())
print(a.flatten())
```

Output

```
[1 2 3 4]
[1 2 3 4]
```

2.4 एरे का आकार बदलना (Resizing Arrays)

आकार बदलना पुनः आकार देने से अलग है। आकार बदलना एरे में तत्वों की कुल संख्या को बदल सकता है।

np.resize() : यह फ़ंक्शन निर्दिष्ट आकार के साथ एक *नया* एरे बनाता है। यदि नया आकार बड़ा है, तो यह इसे भरने के लिए मूल डेटा को दोहराता है। यदि यह छोटा है, तो यह डेटा को छोटा करता है।

उदाहरण 2.12: एरे का आकार बदलना।

```
# Resizing an Array
import numpy as np
a = np.array([1, 2, 3, 4])
# Resize to a 2x3 array (data will repeat)
resized = np.resize(a, (2, 3))
print(resized)

# Resize to a smaller shape (truncated)
resized = np.resize(a, (2, 2))
print(resized)
```

Output

```
[[1 2 3]
[4 1 2]]
```

© PSSCIVE Draft Study, not be Published

```
[[1 2]
[3 4]]
```

ndarray.resize(): यह विधि मूल एरे को *स्थान में* (in-place) संशोधित करती है।

पुनः आकार देना (Reshape) और आकार बदलना (Resize) के बीच मुख्य अंतर

विशेषता (Feature)	**पुनः आकार देना (Reshape)**	**आकार बदलना (Resize)**
लौटाता/संशोधित करता है	एक नया एरे लौटाता है (व्यू या प्रतिलिपि)।	एरे को स्थान में संशोधित करता है (ndarray.resize()) या एक नया एरे लौटाता है (np.resize())।
आकार मेल	पुनः आकार देने से पहले तत्वों की कुल संख्या मेल खानी चाहिए।	तत्वों की कुल संख्या बदल सकती है।
व्यवहार	डेटा को यथावत रखता है, केवल उसे पुनर्व्यवस्थित करता है।	नए आकार में फिट होने के लिए तत्वों को दोहरा या छोटा कर सकता है।

2.5 आयाम बदलना (Changing Dimensions)

हम किसी एरे में आयाम जोड़ या हटा सकते हैं। आकार 1 के आयाम को अक्सर "एकल" (singleton) आयाम कहा जाता है।

आयाम जोड़ना (Adding Dimensions)

np.newaxis: किसी एरे के आयाम को एक से बढ़ाने के लिए उपयोग किया जाता है।

np.expand_dims(): किसी विशिष्ट स्थान पर एक नया अक्ष स्पष्ट रूप से जोड़ता है।

उदाहरण 2.13: एक नया आयाम जोड़ना।

```
# Adding a new dimension to an array.
import numpy as np
a = np.array([1, 2, 3])   # Shape: (3,)
# Add a new dimension as a row
print(a[np.newaxis, :])   # Shape: (1, 3)

# Add a new dimension as a column using expand_dims
print(np.expand_dims(a, axis=1))  # Shape: (3, 1)
```

Output

```
[[1 2 3]]
```

© PSSCIVE Not to be Republished

```
[[1]
[2]
[3]]
```

आयाम हटाना (Removing Dimensions)

np.squeeze(): किसी एरे से आकार 1 के आयाम हटाता है।

उदाहरण 2.14: आकार 1 के आयाम को हटाना।

```
# Removing a dimension of size 1 from an array.
import numpy as np
a = np.array([[[11, 22, 33]]])   # Shape: (1, 1, 3)
# Remove size-1 dimensions
squeezed = np.squeeze(a)
print(squeezed)   # Shape: (3,)
print(squeezed.shape)
```

Output

```
[11 22 33]
(3,)
```

Study Material Not be Published

सारांश (Summary)

- ऐरे संचालन या हेरफेर (Array Manipulation) का अर्थ है डेटा को प्रभावी ढंग से प्रबंधित करने के लिए एरे की संरचना या व्यवस्था में परिवर्तन करना है।
- एरे को जोड़ना (Joining) दो या अधिक एरे को मिलाकर एक ही एरे बनाता है।
- np.concatenate() फ़ंक्शन पंक्तियों (rows) या स्तंभों (columns) के अनुसार एरे को जोड़ता है।
- np.vstack() एरे को ऊर्ध्वाधर (vertically/row-wise) रूप से जोड़ता है, जबकि np.hstack() ऐरे को क्षैतिज (horizontally/column-wise) रूप से जोड़ता है।
- एरे को विभाजित करना (Splitting) एक एरे को कई छोटे एरे में बाँटता है।
- np.split() फ़ंक्शन एरे को समान भागों में विभाजित करता है, जबकि np.array_split() असमान (unequal) विभाजन की अनुमति देता है।
- पुनःआकार देना (Reshaping) एरे के कुल तत्वों की संख्या बदले बिना उसके आकार (shape) को बदलता है।
- np.reshape() फ़ंक्शन का उपयोग एरे को नए आकार में बदलने के लिए किया जाता है।

- समतल करना (Flattening) बहुआयामी एरे को ravel() या flatten() का उपयोग करके एक-आयामी (1-D) एरे में परिवर्तित करता है।
- आकार बदलना (Resizing) एरे के आकार को बदलता है और np.resize() का उपयोग करके तत्वों को दोहरा (repeat) या हटा (remove) सकता है।
- NumPy में np.newaxis या np.expand_dims() का उपयोग करके नए आयाम (dimensions) जोड़े जा सकते हैं।
- np.squeeze() एरे से आकार 1 वाले आयामों (dimensions) को हटा देता है।
- ये सभी एरे संचालन प्रक्रियाएँ बड़े डेटा सेट को व्यवस्थित और विश्लेषित करने में कुशलता प्रदान करती हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. कौन-सा NumPy फ़ंक्शन किसी मौजूदा अक्ष (axis) के साथ दो एरे को जोड़ने के लिए उपयोग किया जाता है? (अ) np.split() (ब) np.concatenate() (स) np.reshape() (द) np.resize()
2. कौन-सा फ़ंक्शन ऐरे को लंबवत/ ऊर्ध्वाधर (पंक्ति-वार / row-wise) रूप से जोड़ता है? (अ) np.vstack() (ब) np.hstack() (स) np.dstack() (द) np.column()
3. कौन सा फ़ंक्शन एरे को क्षैतिज (स्तंभ-वार/column-wise) रूप से जोड़ता है? (अ) np.vstack() (ब) np.hstack() (स) np.vstackaxis() (द) np.row_stack()
4. कौन सा फ़ंक्शन किसी एरे को समान भागों में विभाजित करता है? (अ) np.concatenate() (ब) np.split() (स) np.reshape() (द) np.ones()
5. कौन सा फ़ंक्शन किसी एरे को असमान भागों में विभाजित करने की अनुमति देता है? (अ) np.array_split() (ब) np.split() (स) np.concatenate() (द) np.arange()
6. कौन-सा फ़ंक्शन एरे के डेटा को बदले बिना उसका आकार (shape) बदलने के लिए उपयोग किया जाता है? (अ) np.resize() (ब) np.reshape() (स) np.split() (द) np.stack()
7. कौन-सा फ़ंक्शन एक बहु-आयामी एरे को 1D एरे में परिवर्तित करता है? (अ) reshape() (ब) flatten() (स) concatenate() (द) split()
8. कौन-सा फ़ंक्शन एरे से आकार 1 वाले आयामों (dimensions) को हटाने के लिए उपयोग किया जाता है? (अ) squeeze() (ब) expand() (स) reshape() (द) join()

ब. रिक्त स्थान भरें

1. एरे को जोड़ना (Joining arrays) का अर्थ है कई एरे को मिलाकर एक एरे _____ करना है।
2. _____ फ़ंक्शन एरे को ऊर्ध्वाधर (vertically) रूप से जोड़ता है।
3. _____ फ़ंक्शन एरे को क्षैतिज (horizontally) रूप से जोड़ता है।
4. _____ फ़ंक्शन एरे को नए आकार (structure) में बदलता है।

5. _____ फ़ंक्शन एरे से आकार 1 वाले आयामों (dimensions) को हटा देता है।

स. सत्य या असत्य बताएँ

1. विभाजन (Splitting) एक बड़े एरे को छोटे एरे में विभाजित करता है।
2. reshape() फ़ंक्शन किसी एरे में तत्वों की कुल संख्या बदल सकता है।
3. resize() फ़ंक्शन तत्वों को दोहरा या छोटा कर सकता है।
4. np.vstack() फ़ंक्शन एरे को स्तंभ-वार जोड़ता है।
5. समतलन (Flattening) एक बहु-आयामी एरे को एक-आयामी एरे में परिवर्तित करता है।

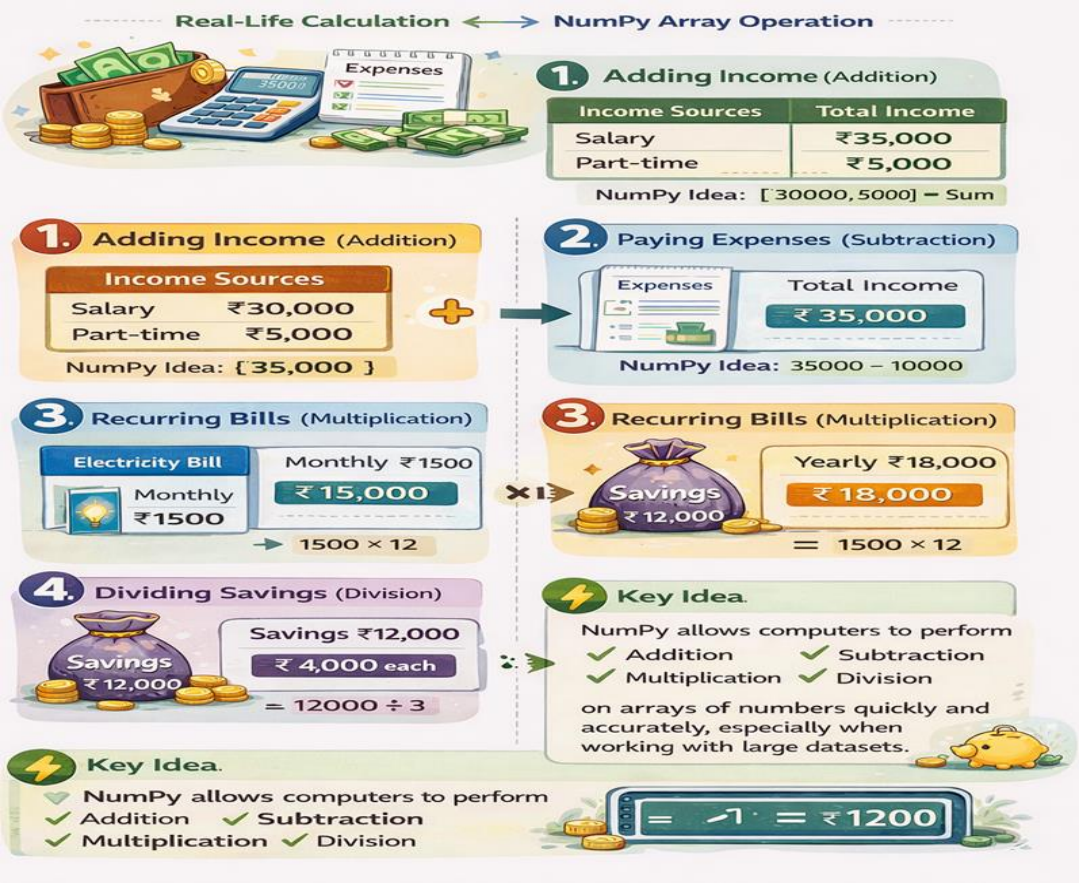
द. लघु उत्तरीय प्रश्न

1. NumPy में एरे संचालन या हेरफेर (Array Manipulation) क्या है?
2. एरे जोड़ने का क्या अर्थ है?
3. np.split() और np.array_split() के बीच क्या अंतर है?
4. NumPy एरे में पुनः आकार देना (reshaping) क्या है?
5. एरे का समतलन (flattening) क्या है?
6. reshape() और resize() के बीच एक अंतर लिखिए?
7. np.expand_dims() का उद्देश्य क्या है?
8. np.squeeze() क्या करता है?

सत्र 3. NumPy का उपयोग करके एरे संगणना

(Array Computation using NumPy)

अपने मासिक खर्चों को संभालने के बारे में सोचें। आप आय जोड़ते हैं, खर्च घटाते हैं, आवर्ती बिलों को गुणा करते हैं, और बचत को जरूरतों के बीच विभाजित करते हैं। ये सभी गणनाएँ हैं। इसी तरह, NumPy एरे पर ऐसी अंकगणितीय और गणितीय संक्रियाओं की अनुमति देता है, लेकिन बहुत तेज़ गति और उच्च सटीकता के साथ। NumPy एरे पर विभिन्न संक्रियाएँ करने का एक शक्तिशाली और कुशल तरीका प्रदान करता है। ये संक्रियाएँ पारंपरिक पायथन सूचियों का उपयोग करने की तुलना में, विशेष रूप से बड़े डेटासेट के साथ बहुत तेज़ हैं।



चित्र 3.1: NumPy के साथ मासिक खर्चों का प्रबंधन

इस सत्र में, आप एरे पर अंकगणितीय संक्रियाएँ (जोड़, घटाव, गुणा, भाग) और गणितीय संक्रियाएँ (मापांक, घातांक, पूर्णांक भाग) करने में सक्षम होंगे। आप योग, माध्य, गुणनफल और मानक विचलन जैसे समग्र फलनों को लागू करना भी सीखेंगे और डॉट उत्पाद और ट्रांसपोज़ जैसी मैट्रिक्स संक्रियाओं का पता लगाएंगे।

3.1 अंकगणितीय संक्रियाएँ (Arithmetic Operations)

NumPy एरे पर अंकगणितीय संक्रियाएँ तत्व-वार (element-wise) की जाती हैं। इसका मतलब है कि संक्रिया एरे से संबंधित तत्वों के प्रत्येक युग्म पर लागू होती है।

यह ध्यान रखना महत्वपूर्ण है कि तत्व-वार संक्रियाओं के लिए, दोनों एरे का आकार समान होना चाहिए (या प्रसारण के माध्यम से संगत होना चाहिए)।

3.2.1 जोड़ (Addition)

यह दो एरे के संगत तत्वों को जोड़ता है या एक अदिश (एकल संख्या) को एरे के प्रत्येक तत्व में जोड़ता है।

ऑपरेटर: +

फ़ंक्शन: np.add()

उदाहरण 3.1: दो एरे का जोड़।

```
# Addition of two arrays
import numpy as np
```

```
a1 = np.array([11, 22, 33])
a2 = np.array([10, 20, 30])
# Element-wise addition
result = a1 + a2
print(result)
```

Output

```
[21 42 63]
```

उदाहरण 3.2: एक एरे में अदिश जोड़ना।

```
# Adding a scalar to an array
result = a1 + 10
print(result)
```

Output

```
[21 32 43]
```

3.2.2. घटाव (Subtraction)

यह दो एरे के संगत तत्वों को घटाता है या प्रत्येक तत्व में से एक अदिश घटाता है।

ऑपरेटर: -

फ़ंक्शन: np.subtract()

उदाहरण 3.3: दो एरे का घटाव।

```
# Subtraction on arrays.
import numpy as np
a1 = np.array([10, 20, 30])
a2 = np.array([40, 50, 60])
result = a2 - a1
print(result)
```

Output

```
[30 30 30]
```

3.2.3. गुणा (Multiplication)

यह दो एरे के संगत तत्वों को गुणा करता है या प्रत्येक तत्व को एक अदिश से गुणा करता है।

ऑपरेटर: *

© PSSCIVE Draft Material Not be Published

फ़ंक्शन: np.multiply()

उदाहरण 3.4: एरे का गुणा।

```
# Multiplication of arrays
import numpy as np
a1 = np.array([11, 22, 33])
a2 = np.array([3, 2, 1])
result = a1 * a2
print(result)
```

Output

[33 44 33]

3.2.4. भाग (Division)

यह दो एरे के संगत तत्वों को विभाजित करता है या प्रत्येक तत्व को एक अदिश से विभाजित करता है।

ऑपरेटर: /

फ़ंक्शन: np.divide()

उदाहरण 3.5: एरे का भाग।

```
# Division of arrays
import numpy as np
a1 = np.array([11, 22, 33])
a2 = np.array([44, 55, 66])
result = a2 / a1
print(result)
```

Output

[4. 2.5 2.]

3.2.5 पूर्णांक भाग (Floor Division)

यह पूर्णांक विभाजन करता है, यह परिणाम के दशमलव भाग को हटा देता है।

ऑपरेटर: //

फ़ंक्शन: np.floor_divide()

उदाहरण 3.6: एरे का पूर्णांक भाग।

```
# Floor division of arrays.
```

© PSSCIVE Draft Study ... Not to be Published

```
import numpy as np
a1 = np.array([10, 20, 30])
a2 = np.array([44, 59, 62])
result = a2 // a1
print(result)
```

Output

[4 2 2]

3.2.6 मापांक (Modulus) (शेषफल)

यह भाग के बाद शेषफल की गणना करता है।

ऑपरेटर: %

फ़ंक्शन: np.mod()

उदाहरण 3.7: मापांक संक्रिया।

```
# Modulus operation
import numpy as np
a1 = np.array([10, 20, 30])
a2 = np.array([42, 55, 66])
result = a2 % a1
print(result)
```

Output

[2 15 6]

3.2.7 घातांक (Exponentiation)

यह किसी एरे के प्रत्येक तत्व को एक घात तक बढ़ाता है।

ऑपरेटर: **

फ़ंक्शन: np.power()

उदाहरण 3.8: घातांक।

```
# Exponentiation
import numpy as np
a1 = np.array([0, 1, 2])
a2 = np.array([4, 5, 6])
```

```
result = np.power(a1, a2)   # a1 raised to the power of a2
print(result)
```

Output

```
[ 0 1 64]
```

3.2.8 अंकगणित में प्रसारण (Broadcasting in Arithmetic)

NumPy एक सुविधा का उपयोग करके विभिन्न आकृतियों के एरे पर संक्रियाएँ कर सकता है जिसे **प्रसारण (broadcasting)** कहा जाता है। छोटे एरे को बड़े एरे में "प्रसारित" किया जाता है ताकि उनके आकार संगत हो जाएँ।

उदाहरण 3.9: प्रसारण।

```
# Broadcasting in arithmetic
import numpy as np
a = np.array([[1, 2], [3, 4]])   # 2x2 array

# Broadcasting a scalar
result = a + 12
print("Array + scalar:\n", result)

# Broadcasting a 1D array
result = a + np.array([1, 2])     # 1x2 array is broadcast to each row
print("\nArray + 1D array:\n", result)
```

Output

```
Array + scalar:
[[13 14]
[15 16]]
Array + 1D array:
[[2 4]
[4 6]]
```

3.3 संपूर्ण फ़ंक्शन (Aggregate Functions)

यह फ़ंक्शन संपूर्ण एरे से या किसी विशिष्ट अक्ष के साथ एकल परिणाम की गणना करते हैं।

np.sum() : सभी तत्वों के योग की गणना करता है।

© PSSCIVE Draft Study Material Not be Publi

उदाहरण 3.10: एरे तत्वों का योग।

```
import numpy as np
a = np.array([3, 5, 6, 7])
print(np.sum(a))
```

Output

21

np.prod() : Calculates the product of all elements.

np.prod() : सभी तत्वों के गुणनफल की गणना करता है।

उदाहरण 3.11: एरे तत्वों का गुणनफल।

```
import numpy as np
a = np.array([1, 2, 3, 4])
print(np.prod(a))
```

Output

24

np.mean() : सभी तत्वों के औसत (माध्य) की गणना करता है।

उदाहरण 3.12: एरे तत्वों का माध्य।

```
import numpy as np
a = np.array([5, 6, 3, 4])
print(np.mean(a))
```

Output

4.5

np.std() : मानक विचलन की गणना करता है, जो मापता है कि संख्याएँ कितनी फैली हुई हैं।

उदाहरण 3.13: मानक विचलन।

```
import numpy as np
a = np.array([1, 2, 3, 4])
print(np.std(a))
```

Output

1.118033988749895

SCIVE Draft Study Material Not be Published

np.cumsum() : संचयी योग की गणना करता है। (उदा., [1,2,3] के लिए -> [1, 1+2, 1+2+3])

उदाहरण 3.14: संचयी योग।

```
import numpy as np
a = np.array([1, 2, 3, 4])
print(np.cumsum(a))
```

Output

```
[ 1 3 6 10]
```

np.cumprod() : संचयी गुणनफल की गणना करता है।

उदाहरण 3.15: संचयी गुणनफल।

```
import numpy as np
a = np.array([1, 2, 3, 4])
print(np.cumprod(a))
```

Output

```
[ 1 2 6 24]
```

मैट्रिक्स संक्रियाएँ: np.dot() : दो एरे पर मैट्रिक्स गुणन (डॉट उत्पाद) करता है।

उदाहरण 3.16: डॉट उत्पाद।

```
# Matrix multiplication
import numpy as np
a = np.array([[1, 2], [3, 4]])
b = np.array([[5, 6], [7, 8]])
result = np.dot(a, b)
print(result)
```

Output

```
[[19 22]
[43 50]]
```

3.4 त्रिकोणमितीय फ़ंक्शन (Trigonometric Functions)

NumPy त्रिकोणमितीय गणनाओं के लिए भी फ़ंक्शन प्रदान करता है। कोणों को रेडियन में होना चाहिए जब तक कि परिवर्तित न किया जाए।

np.sin(), np.cos(), np.tan(): साइन, कोसाइन और टेंजेंट की गणना करें।

© PSSCIVE Draft Study Material Not be Published

np.arcsin(), np.arccos(), np.arctan(): व्युत्क्रम त्रिकोणमितीय फलनों की गणना करें। वे रेडियन में कोण लौटाते हैं।

np.radians(): डिग्री को रेडियन में परिवर्तित करता है।

np.degrees(): रेडियन को डिग्री में परिवर्तित करता है।

np.hypot(): एक समकोण त्रिभुज के कर्ण की गणना करता है।

उदाहरण 3.17: त्रिकोणमितीय फ़ंक्शन।

```
# Trigonometric functions
import numpy as np

angles_deg = np.array([0, 90, 180])
angles_rad = np.radians(angles_deg)  # Convert to radians first

print("Sine of angles:", np.sin(angles_rad))
print("Cosine of angles:", np.cos(angles_rad))
print("Tangent of 45 deg:", np.tan(np.radians(45)))

# Inverse sine
print("Arcsin of 1 (in degrees):", np.degrees(np.arcsin(1)))

# Hypotenuse of a 3-4-5 triangle
print("Hypotenuse:", np.hypot(3, 4))
```

Output

```
Sine of angles: [0.0000000e+00 1.0000000e+00 1.2246468e-16]
Cosine of angles: [ 1.000000e+00 6.123234e-17 -1.000000e+00]
Tangent of 45 deg: 0.9999999999999999
Arcsin of 1 (in degrees): 90.0
Hypotenuse: 5.0
```

© PSSCIVE Draft Study Material Not be Published

सारांश (Summary)

- NumPy एरे पर तीव्र अंकगणितीय और गणितीय क्रियाएँ करने की सुविधा देता है।
- एरे पर क्रियाएँ तत्व-वार (element-wise) की जाती हैं।
- क्रियाएँ करने के लिए एरे का आकार समान या संगत (compatible) होना चाहिए।
- मूलभूत क्रियाओं में जोड़, घटाव, गुणा और भाग शामिल हैं।
- NumPy फ्लोर डिवीजन, मॉड्यूलस और घातांक (exponentiation) जैसी क्रियाओं को भी समर्थन देता है।
- प्रसारण (ब्रॉडकास्टिंग) के माध्यम से अलग-अलग आकार के एरे पर भी क्रियाएँ की जा सकती हैं।
- sum(), mean(), prod() और std() जैसे समग्र (एग्रीगेट) फ़ंक्शन एरे के समग्र मानों की गणना करते हैं।
- NumPy मैट्रिक्स ऑपरेशन्स जैसे डॉट प्रोडक्ट का भी समर्थन करता है।
- यह sin(), cos() और tan() जैसे त्रिकोणमितीय फ़ंक्शन भी प्रदान करता है।
- ये सभी विशेषताएँ कुशल संख्यात्मक गणना और डेटा विश्लेषण में सहायता करती हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. NumPy में, एरे पर अंकगणितीय क्रियाएँ की जाती हैं: (अ) रेंडम रूप से(Randomly) (ब) तत्व-वार (Element-wise) (स) केवल पंक्ति-वार (Row-wise only) (द) केवल स्तंभ-वार (Column-wise only)
2. NumPy एरे में घातांक (Exponentiation) के लिए कौन-सा ऑपरेटर का उपयोग होता है? (अ) ^ (ब) ** (स) % (द) //
3. कौन-सा फ़ंक्शन किसी एरे के सभी तत्वों का योग (sum) निकालने के लिए उपयोग होता है? (अ) np.add() (ब) np.sum() (स) np.mean() (द) np.prod()
4. कौन-सा फ़ंक्शन किसी एरे के तत्वों का औसत (average) मान निकालने के लिए उपयोग होता है? (अ) np.sum() (ब) np.prod() (स) np.mean() (द) np.std()
5. विभाजन के बाद शेषफल (remainder) देने वाली क्रिया कौन-सी है? (अ) मापांक (Modulus) (ब) भाग (Division) (स) गुणा (Multiplication) (द) जोड़ (Addition)
6. NumPy में मैट्रिक्स गुणन (matrix multiplication) के लिए कौन-सा फ़ंक्शन उपयोग होता है? (अ) np.multiply() (ब) np.dot() (स) np.sum() (द) np.prod()
7. कौन-सा NumPy फ़ंक्शन डिग्री (degrees) को रेडियन (radians) में बदलता है? (अ) np.degrees() (ब) np.radians() (स) np.sin() (द) np.cos()
8. एरे के तत्वों का मानक विचलन (standard deviation) निकालने के लिए कौन-सा फ़ंक्शन उपयोग होता है? (अ) np.mean() (ब) np.std() (स) np.sum() (द) np.add()

ब. रिक्त स्थान भरें

1. NumPy अंकगणितीय क्रियाएँ _____ तरीके से करता है।
2. फ़ंक्शन _____ एरे तत्वों के माध्य या औसत (mean) की गणना करता है।
3. ऑपरेटर _____ का उपयोग मॉड्यूलस (modulus) क्रिया के लिए किया जाता है।
4. फ़ंक्शन _____ एरे तत्वों के संचयी योग (cumulative sum) की गणना करता है।
5. मैट्रिक्स गुणन (matrix multiplication) करने के लिए फ़ंक्शन _____ का उपयोग किया जाता है।

स. सत्य या असत्य बताएं

1. NumPy की क्रियाएँ (operations) पायथन लिस्ट (सूची) की क्रियाओं की तुलना में धीमी होती हैं।
2. प्रसारण (ब्रॉडकास्टिंग) विभिन्न आकार (shapes) वाले एरे के बीच भी क्रियाएँ करने की सुविधा देता है।
3. फ्लोर डिवीजन (Floor division) परिणाम के दशमलव भाग को हटा देता है।
4. फ़ंक्शन np.prod() एरे के सभी तत्वों का गुणनफल (product) की गणना करता है।
5. NumPy में त्रिकोणमितीय (trigonometric) फ़ंक्शनों के लिए कोण (angles) रेडियन (radians) में होना आवश्यक है।

द. लघु उत्तरीय प्रश्न

1. NumPy में एरे संगणना (array computation) क्या है?
2. NumPy में प्रसारण (broadcasting) क्या होता है?
3. फ़ंक्शन np.sum() क्या करता है?
4. np.dot() का उद्देश्य क्या है?
5. np.std() क्या गणना करता है?
6. भाग (division) (/) और पूर्णांक भाग (floor division) (//) में क्या अंतर है?
7. NumPy में उपलब्ध कोई दो त्रिकोणमितीय (trigonometric) फ़ंक्शन लिखें।
8. np.cumsum() का क्या उद्देश्य है?

परिशिष्ट I: NumPy में डेटा प्रकार

(Data Types in NumPy)

NumPy बड़ी संख्या में डेटा प्रकारों का समर्थन करता है। यहाँ कुछ सामान्य प्रकार दिए गए हैं।

1. पूर्णांक प्रकार (Integer Type)

पूर्ण संख्याओं को संग्रहीत करने के लिए उपयोग किया जाता है।

डेटा प्रकार	विवरण	उदाहरण

int8	8-बिट हस्ताक्षरित पूर्णांक	>>> np.array([127, -128], dtype=np.int8)
uint8	8-बिट अहस्ताक्षरित पूर्णांक	>>> np.array([0, 255], dtype=np.uint8)
int16	16-बिट हस्ताक्षरित पूर्णांक	>>> np.array([-32768, 32767], dtype=np.int16)
int32	32-बिट हस्ताक्षरित पूर्णांक	>>> np.array([1, -2147483648], dtype=np.int32)
int64	64-बिट हस्ताक्षरित पूर्णांक	>>>np.array([1,-9223372036854775808], dtype=np.int64)

2. फ़्लोटिंग-पॉइंट प्रकार (Floating-Point Types)

दशमलव बिंदु वाली संख्याओं के लिए उपयोग किया जाता है।

डेटा प्रकार	विवरण	उदाहरण
float16	अर्ध-परिशुद्धता	>>> np.array([1.5, -2.5], dtype=np.float16)
float32	एकल-परिशुद्धता	>>> np.array([1.234, -5.678], dtype=np.float32)
float64	दोहरी-परिशुद्धता	>>> np.array([1.123456789, -9.87654321], dtype=np.float64)

3. बूलियन प्रकार (Boolean Type)

सत्य या असत्य मानों के लिए उपयोग किया जाता है।

डेटा प्रकार	विवरण	उदाहरण
bool_	बूलियन प्रकार	>>> np.array([True, False], dtype=np.bool_)

4. स्ट्रिंग प्रकार (String Types)

पाठ संग्रहीत करने के लिए उपयोग किया जाता है।

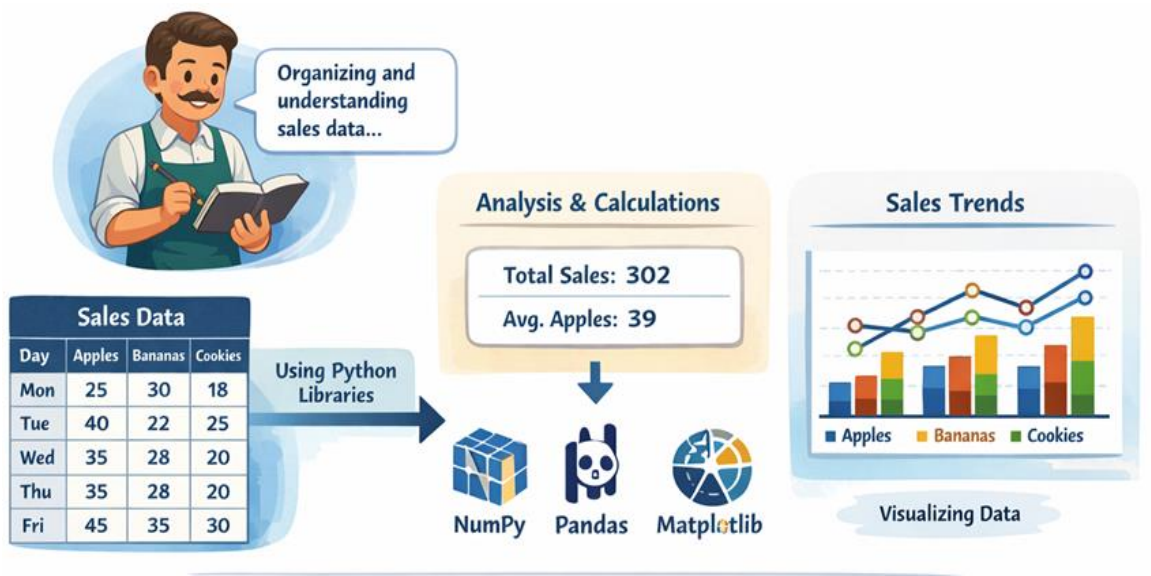
डेटा प्रकार	विवरण	उदाहरण
string_	निश्चित-आकार ASCII स्ट्रिंग	>>> np.array(['a', 'bc'], dtype=np.string_)
unicode_	निश्चित-आकार यूनिकोड स्ट्रिंग	>>> np.array(['hello', 'world'], dtype=np.unicode_)

मॉड्यूल 3. डेटा विश्लेषण

(Data Analysis)

मॉड्यूल संक्षिप्त परिचय (Module Overview)

कल्पना कीजिए एक दुकानदार की, जो अपनी दैनिक बिक्री को एक कॉपी में लिखता है। यदि वह कॉपी केवल संख्याएँ ही दिखाती है, तो यह समझना कठिन हो जाता है कि कौन-सा उत्पाद सबसे अधिक बिक रहा है या समय के साथ बिक्री में क्या बदलाव आ रहा है। लेकिन यदि वह इन आँकड़ों को एक तालिका में व्यवस्थित करता है, सरल गणनाओं के माध्यम से उनका विश्लेषण करता है और उन्हें ग्राफ के रूप में प्रस्तुत करता है, तो वह आसानी से पैटर्न पहचान सकता है और बेहतर निर्णय ले सकता है। यही डेटा विश्लेषण (Data Analysis) का मूल उद्देश्य है—यह हमें कच्चे डेटा को उपयोगी जानकारी में बदलने में मदद करता है।



चित्र 3.0: पायथन लाइब्रेरी के साथ डेटा विश्लेषण सीखना

इस इकाई (unit) में आप सीखेंगे कि पायथन (Python) की लाइब्रेरीज़ जैसे NumPy, Pandas और Matplotlib का उपयोग करके डेटा को कैसे संभाला जाए, उसका विश्लेषण कैसे किया जाए और उसे कैसे प्रस्तुत किया जाए। इसमें सरल तालिकाएँ बनाने से लेकर गणनाएँ करने और अंत में चार्ट व ग्राफ बनाने तक सीखेंगे। यह आपको वास्तविक जीवन में डेटा विश्लेषण के लिए आवश्यक बुनियादी कौशल प्रदान रने में मदद करेगी।

अधिगम के परिणाम (Learning Outcomes)

इस मॉड्यूल को पूरा करने के बाद आप सक्षम होंगे:

- वास्तविक जीवन स्थितियों में डेटा विश्लेषण के महत्व को समझेंगे।
- विभिन्न प्रकार के डेटा और उनके प्रतिनिधित्व की पहचान करेंगे।

- NumPy, Pandas, और Matplotlib जैसी पायथन लाइब्रेरी का उपयोग करेंगे।
- कच्चे डेटा (raw data) को संरचित प्रारूपों (तालिकाओं) में व्यवस्थित करेंगे।
- बुनियादी डेटा विश्लेषण और गणनाएँ करेंगे।
- डेटा को विज़ुअलाइज़ करने के लिए चार्ट और ग्राफ बनाएंगे।
- बेहतर निर्णय लेने के लिए पैटर्न और रुझानों की व्याख्या करेंगे।

मॉड्यूल संरचना (Module Structure)

सत्र 1. पांडस का परिचय (Introduction to Pandas)

सत्र 2. पांडस सीरीज (Pandas Series)

सत्र 3. पांडस डेटाफ्रेम (Pandas Dataframe)

सत्र 4. मैटप्लोटलिब का उपयोग करके डेटा विज़ुअलाइज़ेशन (Data Visualisation Using Matplotlib)

सत्र 1. पांडस का परिचय

(Introduction to Pandas)

कल्पना कीजिए कि आप अपनी कक्षा के सभी विद्यार्थियों के अंकों का प्रबंधन कर रहे हैं। यदि यह डेटा केवल एक कॉपी में लिखा हो, तो जल्दी से यह पता लगाना कठिन हो जाता है कि टॉपर कौन है या कक्षा का औसत कितना है। लेकिन यदि यही डेटा कंप्यूटर पर एक सुव्यवस्थित टेबल के रूप में रखा जाए, तो आप इन उत्तरों को तुरंत प्राप्त कर सकते हैं। पांडस हमें पायथन में ठीक इसी प्रकार डेटा के साथ काम करने में मदद करता है, जैसा कि चित्र 1.1 में दिखाया गया है।

चित्र 1.1: पांडस के साथ विद्यार्थी अंकों का संगठन

इस सत्र में आप सीखेंगे कि Pandas क्या है, यह साधारण लिस्ट की तुलना में क्यों बेहतर है तथा इसके मूल डेटा संरचनाओं (data structures) जैसे Series और DataFrames को कैसे बनाया और उपयोग किया जाता है।

1.1 परिचय

पांडस एक लोकप्रिय पायथन लाइब्रेरी है जिसका उपयोग डेटा के साथ काम करने के लिए किया जाता है। इसे एक शक्तिशाली टूल की तरह समझा जा सकता है, जो Excel में दिखाई देने वाली टेबल्स को बनाने और प्रबंधित करने में मदद करता है। यह NumPy नामक दूसरी लाइब्रेरी के ऊपर आधारित है और डेटा का विश्लेषण (analyze), साफ-सफाई (clean) तथा अन्वेषण (explore) करना बहुत आसान बना देता है।

Pandas में दो मुख्य डेटा संरचनाएँ (data structures) होती हैं, जो डेटा को व्यवस्थित करने में मदद करती हैं: **सीरीज (Series)** और **डेटाफ्रेम (DataFrame)**।

नाम "पांडस" शब्द "*पैनल डेटा (Panel Data)*" शब्द से आया है, जिसका अर्थ बहु-आयामी (multi-dimensional) डेटा के बारे में बात करने का एक तरीका से है। इसे 2008 में वेस मैककिनी (Wes McKinney) ने बनाया था।

1.1.1 पांडस की आवश्यकता?

आप सोच सकते हैं, एक साधारण सूची (list) या शब्दकोश (dictionary) का उपयोग क्यों न करें? यहाँ बताया गया है कि पांडस क्यों बेहतर है:

***मिश्रित डेटा प्रकार* (Mixed Data Types):** पायथन की एक साधारण सूची (list) आमतौर पर एक ही प्रकार का डेटा रखती है। जबकि Pandas की तालिका या डेटाफ्रेम (DataFrame) में एक ही स्थान पर टेक्स्ट या पाठ (नाम), संख्याएँ (अंक) और तारीखें (जन्मदिन) अलग-अलग कॉलम में रखी जा सकती हैं।

***उपयोग में आसानी* (Easy to Use):** Pandas में जटिल कार्यों जैसे फ़ाइल से डेटा लोड करना, उसे फ़िल्टर करना या समूह (group) बनाना बहुत आसान होता है। ये काम सामान्य Python में करने के लिए कई पंक्तियों का कोड लिखना पड़ता है।

***संगठित डेटा* (Organized Data):** डेटाफ्रेम में स्तंभ (कॉलम) नाम और पंक्ति के लेबल होते हैं, जिससे यह समझना बहुत आसान हो जाता है कि डेटा क्या दर्शाता है। यह एक साधारण सूची (list) की तुलना में अधिक व्यवस्थित है।

1.1.2 आप पांडस के साथ क्या कर सकते हैं?

पांडस डेटा विश्लेषण के लिए एक आदर्श उपकरण है क्योंकि यह निम्नलिखित कार्यों को सरल बनाता है:

डेटा सफाई (Data Cleaning): अव्यवस्थित या भिखरे डेटा (जैसे लापता मान) को ठीक करना।

डेटा परिवर्तन (Data Transformation): आपके डेटा के प्रारूप को बदलना।

डेटा विलय (Data Merging): दो अलग-अलग डेटासेट को एक साथ जोड़ना।

डेटा विश्लेषण (Data Analysis): औसत, योग और अन्य आँकड़ों की आसानी से गणना करना।

1.1.3 NumPy और Pandas के बीच अंतर

NumPy और Pandas दोनों ही बहुत उपयोगी पायथन लाइब्रेरी हैं, लेकिन इनका उपयोग विभिन्न कार्यों के लिए किया जाता है। यहाँ एक सरल तुलना दी गई है:

विशेषता	NumPy	Pandas

मुख्य उद्देश्य	संख्याओं पर गणितीय क्रियाएँ करना।	सभी प्रकार के डेटा का प्रबंधन (manipulate) और विश्लेषण (analyse) करना।
डेटा संरचना	एरे (Arrays) — यानी संख्याओं की एक ग्रिड (जाल) के रूप में डेटा।	सीरीज (1D) और डेटाफ्रेम (2D - एक तालिका की तरह)।
डेटा प्रकार	यह एक ही प्रकार के डेटा (जैसे सभी संख्याएँ) के साथ सबसे अच्छा काम करता है।	एक ही तालिका में विभिन्न प्रकार के डेटा (टेक्स्ट, संख्याएँ, तारीखें) को संभाल सकता है।
उपयोग में आसानी	इसमें आपको अधिकांश काम स्वयं करना पड़ता है।	सामान्य कार्यों के लिए सरल और उच्च-स्तरीय (high-level) कमांड प्रदान करता है।

1.2 पांडस इंस्टॉलेशन (Installation of Pandas)

Pandas का उपयोग करने के लिए सबसे पहले यह सुनिश्चित करना आवश्यक है कि आपके कंप्यूटर में Python इंस्टॉल हो।

जांचें कि पायथन इंस्टॉल है या नहीं:

अपने कमांड प्रॉम्प्ट (या टर्मिनल) को खोलें और यह टाइप करें:

```Bash
python --version
```

यदि आपको अपने कंप्यूटर में कोई पायथन वर्ज़न नंबर दिखाई देता है, तो इसका मतलब है कि Python इंस्टॉल है। यदि नहीं, तो आपको पहले Python इंस्टॉल करना होगा।

Pandas इंस्टॉल करें:

जब पायथन तैयार हो जाए, तब आप pip नामक टूल का उपयोग करके Pandas इंस्टॉल कर सकते हैं। अपने कमांड प्रॉम्प्ट में निम्नलिखित कमांड टाइप करें:

```Bash
pip install pandas
```

© PSSCIVE

```
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\GFG0371>pip install pandas
Defaulting to user installation because normal site-packages is not writeable
Collecting pandas
  Downloading pandas-2.1.3-cp312-cp312-win_amd64.whl.metadata (18 kB)
Requirement already satisfied: numpy<2,>=1.26.0 in c:\users\gfg0371\appdata\roam
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\gfg0371\appdat
Collecting pytz>=2020.1 (from pandas)
  Downloading pytz-2023.3.post1-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.1 (from pandas)
  Downloading tzdata-2023.3-py2.py3-none-any.whl (341 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 341.8/341.8 kB 1.5 MB/s eta 0:00:0
Requirement already satisfied: six>=1.5 in c:\users\gfg0371\appdata\roaming\pyth
Downloading pandas-2.1.3-cp312-cp312-win_amd64.whl (10.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.5/10.5 MB 2.1 MB/s eta 0:00:00
Downloading pytz-2023.3.post1-py2.py3-none-any.whl (502 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 502.5/502.5 kB 3.2 MB/s eta 0:00:00
Installing collected packages: pytz, tzdata, pandas
```

यह कमांड Pandas लाइब्रेरी को डाउनलोड और इंस्टॉल करेगा।

अपने कोड में Pandas इम्पोर्ट करें (Import Pandas in your code):

इंस्टॉलेशन के बाद, इसका उपयोग करने के लिए आपको इसे अपने पायथन प्रोग्राम में इम्पोर्ट करना होगा। आमतौर पर इसे एक छोटे नाम (alias) pd के साथ इम्पोर्ट किया जाता है।

python

```
import pandas as pd
```

अब हर बार pandas लिखने की बजाय आप सिर्फ pd लिखकर काम कर सकते हैं।

1.3 पांडस में डेटा संरचनाएँ (Data Structures in Pandas)

पांडस की दो मुख्य डेटा संरचनाएँ हैं जिनका आप सबसे अधिक उपयोग करेंगे:

सीरीज (Series): यह एक एक-आयामी (one-dimensional) लेबलयुक्त एरे होता है। इसे आप तालिका के **एक कॉलम (एकल स्तंभ)** के रूप में समझ सकते हैं।

डेटाफ्रेम (DataFrame): यह एक दो-आयामी (two-dimensional) लेबलयुक्त डेटा संरचना होती है। इसे आप कई पंक्तियों (rows) और कॉलम (columns) वाली **संपूर्ण तालिका** के रूप में समझ सकते हैं।

डेटाफ्रेम को कई सीरीज ऑब्जेक्ट रखने वाले कंटेनर के रूप में समझ सकते है। डेटाफ्रेम में प्रत्येक स्तंभ एक सीरीज है।

यहाँ आप अपने दिमाग में एक डेटाफ्रेम के एक सरल उदाहरण सोच सकते है:

Name	**Age**	**City**
Asha	20	Nagpur
Boby	22	Bhopal

इस तालिका में 'Name' स्तंभ (कॉलम) एक सीरीज (Series) है, 'Age' कॉलम दूसरी Series है और इसी प्रकार अन्य कॉलम भी अलग-अलग Series होते हैं। पूरी तालिका मिलकर एक DataFrame बनाती है।

सीरीज और डेटाफ्रेम की तुलना

विशेषता	**सीरीज (Series)**	**डेटाफ्रेम (DataFrame)**
आयाम (Dimensions)	एक-आयामी (1D)	दो-आयामी (2D)
संरचना (Structure)	एकल स्तंभ या सूची की तरह।	एक संपूर्ण तालिका या स्प्रेडशीट की तरह।

सारांश (Summary)

- Pandas डेटा विश्लेषण (data analysis) के लिए उपयोग की जाने वाली एक Python लाइब्रेरी है।
- यह NumPy के ऊपर आधारित है।
- इसमें दो मुख्य डेटा संरचनाएँ होती हैं: सीरीज (1D) और डेटाफ्रेम (2D)।
- यह डेटा को साफ करने (cleaning), खोजने (exploring) और प्रबंधन (manipulating) के लिए बहुत उपयोगी है।
- यह विभिन्न प्रकार के डेटा (संख्याएँ, टेक्स्ट, तारीखें) को बहुत अच्छी तरह संभालता है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. पायथन में पांडस (Pandas) का मुख्य रूप से उपयोग किसके लिए किया जाता है? (अ) वेबसाइट बनाना (ब) एरे के साथ संख्यात्मक गणना हेतु (स) सारणीबद्ध डेटा के साथ काम करना और और उसका विश्लेषण करना (द) वीडियो गेम बनाना
2. पांडस कौन-सी दो मुख्य डेटा संरचनाएँ प्रस्तुत करता है? (अ) सूचियाँ और शब्दकोश (ब) एरे और मैट्रिसेस (स) सीरीज और डेटाफ्रेम (द) सेट और टपल
3. पांडस में एक-आयामी लेबल वाला एरे निम्न में से कौन सा है? (अ) डेटाफ्रेम (ब) सीरीज (स) पैनल (द) तालिका
4. एक पांडस डेटाफ्रेम किसके समान होते है? (अ) डेटा का एकल कॉलम (ब) पंक्तियों और कॉलम वाली दो-आयामी तालिका (स) एक सरल सूची (द) एक गणितीय समीकरण
5. आपने पायथन कोड में पांडस को सामान्यतः कैसे इम्पोर्ट करते हैं? (अ) import panda (ब) import pandas as pd (स) include pandas (द) use pandas
6. पांडस किस अन्य पायथन लाइब्रेरी के ऊपर आधारित है? (अ) Matplotlib (ब) Scikit-learn (स) NumPy (द) SciPy

7. my_data नाम के DataFrame की शुरुआती कुछ पंक्तियाँ देखने के लिए कौन-सा फ़ंक्शन उपयोग करेंगे? (अ) my_data.first() (ब) my_data.top() (स) my_data.head() (द) my_data.start()
8. पांडस के डेटाफ्रेम या सीरीज में NaN क्या दर्शाता है? (अ) शून्य के बराबर संख्या (ब) एक स्ट्रिंग मान (स) लापता या अपरिभाषित डेटा (द) एक गणना त्रुटि
9. यदि आप [10, 20, 30] जैसी सूची से बिना इंडेक्स दिए Series बनाते हैं, तो डिफ़ॉल्ट इंडेक्स क्या होगा? (अ) ['A', 'B', 'C'] (ब) [1, 2, 3] (स) [0, 1, 2] (द) ['item1', 'item2', 'item3']
10. पांडस विशेष रूप से किस प्रकार के डेटा को संभालने में उपयोगी है? (अ) छवियाँ (Images) (ब) सारणीबद्ध डेटा, जैसे स्प्रेडशीट (स) ऑडियो फ़ाइलें (द) बाइनरी डेटा

ब. रिक्त स्थान भरें

1. पांडस एक लोकप्रिय पायथन लाइब्रेरी है जिसका उपयोग मुख्य रूप से डेटा _____ और विश्लेषण के लिए किया जाता है। पांडस में दो मुख्य डेटा संरचनाएँ सीरीज और _____ हैं। एक पांडस सीरीज एक _____-आयामी लेबल वाला एरे है। एक पांडस डेटाफ्रेम एक _____- आयामी (dimensional) लेबलयुक्त एरे होता है।
2. Pandas DataFrame एक _____-आयामी तालिका या स्प्रेडशीट के समान होता है।
3. पांडस का उपयोग शुरू करने के लिए, आप आमतौर पर इसे import pandas as _____ का उपयोग करके इम्पोर्ट करते हैं।
4. पांडस _____ लाइब्रेरी के ऊपर आधारित है, जो कुशल एरे ऑपरेशन्स प्रदान करती है।
5. df._____() मेथड का उपयोग DataFrame df की शुरुआती कुछ पंक्तियाँ देखने के लिए किया जाता है।
6. पांडस में लापता या गायब (missing) डेटा बिंदुओं को आमतौर पर NaN के रूप में दर्शाया जाता है, जिसका अर्थ है _____ a Number।
7. जब किसी सूची (list) से बिना इंडेक्स दिए Series बनाई जाती है, तो डिफ़ॉल्ट इंडेक्स पूर्णांक (integer) आधारित होता है, जो _____ से शुरू होता है।
8. पांडस विशेष रूप से संरचित (structured) डेटा के साथ काम करने में अच्छा है, जिसे अक्सर _____ डेटा कहा जाता है।

स. सत्य या असत्य बताएँ

1. पांडस एक पायथन लाइब्रेरी है जिसका उपयोग वेबसाइट बनाने के लिए किया जाता है।
2. पांडस की दो मुख्य डेटा संरचनाएँ सीरीज और डेटाफ्रेम हैं।
3. एक पांडस सीरीज एक दो-आयामी (two-dimensional) लेबलयुक्त एरे है।
4. एक पांडस डेटाफ्रेम पंक्तियों और स्तंभों वाली एक तालिका की तरह है।
5. आप आमतौर पर पांडस को import pandas as pd का उपयोग करके इम्पोर्ट करते हैं।
6. पांडस Matplotlib लाइब्रेरी के ऊपर आधारित है।

7. df.head() विधि डेटाफ्रेम की अंतिम 5 पंक्तियाँ दिखाती है।
8. पांडस में NaN का अर्थ "Not a Number" होता है और यह लापता या गायब डेटा को दर्शाता है।
9. जब आप किसी सूची से एक सीरीज बनाते हैं, तो डिफ़ॉल्ट इंडेक्स 1 से शुरू होता है।
10. पांडस स्प्रेडशीट जैसे दिखने वाले डेटा को संभालने में बहुत अच्छा है।

द. लघु उत्तरीय प्रश्न

1. Python में पांडस (Pandas) क्या है?
2. पांडस कैसे इंस्टॉल करें?
3. सीरीज और डेटाफ्रेम के बीच क्या अंतर है?
4. पांडस में डेटा संरचनाओं (Data Structures) के विभिन्न प्रकार बताइए।
5. पांडस में सीरीज को परिभाषित कीजिए।
6. पांडस में डेटाफ्रेम को परिभाषित कीजिए।
7. पांडस में सूचकांक (index) क्या होता है?

सत्र 2. पांडस सीरीज

(Pandas Series)

अपने दोस्तों के नामों की एक सूची के बारे में सोचिए। अब उनके फोन नंबरों की एक अलग सूची के बारे में सोचिए। यदि ये दोनों सूचियाँ अलग-अलग हों, तो किसी दोस्त का नंबर ढूँढना मुश्किल हो जाता है। लेकिन अगर आप हर नाम को उसके नंबर के साथ जोड़ दें, तो यह काम आसान हो जाता है। एक पांडस सीरीज भी बिल्कुल यही काम करती है। यह हर डेटा (जैसे फोन नंबर) को एक लेबल (जैसे नाम) के साथ जोड़ती है। यह लेबलयुक्त डेटा एक साधारण सूची की तुलना में अधिक शक्तिशाली होता है।

इस सत्र में, आप सीखेंगे कि सीरीज (Series) कैसे बनाई जाती है, उसके डेटा को कैसे एक्सेस किया जाता है और उसकी विशेष सुविधाओं (features) का उपयोग कैसे किया जाता है।

2.1 परिचय

Pandas की Series एक स्मार्ट सूची (list) की तरह होती है। यह डेटा को एक निश्चित क्रम में रखती है और प्रत्येक डेटा के साथ एक अद्वितीय लेबल होता है, जिसे **सूचकांक (index)** कहा जाता है। यह index डेटा को ढूँढने, जोड़ने या बदलने को बहुत आसान बना देता है। आप Series में संख्याएँ, टेक्स्ट या अन्य Python ऑब्जेक्ट्स भी स्टोर कर सकते हैं।

2.2 सीरीज का निर्माण (Creation of Series)

आप विभिन्न स्रोतों से एक सीरीज बना सकते हैं। पहले, pandas इम्पोर्ट करना याद रखिएं।

1. एक खाली सीरीज बनाना (Creating an empty series)

यह बिना किसी डेटा के एक सीरीज बनाता है। यह कुछ भी लिखने से पहले एक नई नोटबुक बनाने जैसा ही है।

```
import pandas as pd
empty_series = pd.Series()
print(empty_series)
# Output: Series([], dtype: object)
```

2. पायथन सूची से सीरीज का निर्माण (Creation of Series from a Python list)

यह सबसे सामान्य तरीका है। आप pd.Series() में एक सूची (list) देते हैं और Pandas स्वतः ही उसे 0 से शुरू होने वाला संख्यात्मक (numeric) इंडेक्स प्रदान कर देता है।

```
import pandas as pd
data = [10, 20, 30, 40, 50]
series = pd.Series(data)
print(series)
# Output:
# 0    10
# 1    20
# 2    30
# 3    40
# 4    50
# dtype: int64
```

3. शब्दकोश (Dictionary) से सीरीज का निर्माण

जब आप dictionary का उपयोग करते हैं, तो उसके keys (कुंजियाँ सूचकांक) इंडेक्स लेबल बन जाते हैं और values डेटा बन जाते हैं। यह लेबलयुक्त डेटा के लिए बिल्कुल उपयुक्त होता है।

```
import pandas as pd
data = {'a': 10, 'b': 20, 'c': 30}
series = pd.Series(data)
print(series)
# Output:
# a    10
# b    20
# c    30
```

© PSSCIF Draft Study Material Not be Published

```
# dtype: int64
```

4. NumPy एरे से सीरीज का निर्माण

यदि आप NumPy के साथ काम कर रहे हैं, तो आप आसानी से एक एरे को सीरीज में बदल सकते हैं।

```
import pandas as pd
import numpy as np
data = np.array([10, 20, 30, 40, 50])
series = pd.Series(data)
print(series)
# Output:
# 0    10
# 1    20
# 2    30
# 3    40
# 4    50
# dtype: int64
```

5. अदिश (Scalar) मान से सीरीज का निर्माण

आप एक सीरीज बना सकते हैं जहाँ प्रत्येक मान समान हो। आपको इसे बनाने के लिए एक सूचकांक प्रदान करना होगा।

```
import pandas as pd
scalar = 10
index = ['a', 'b', 'c', 'd', 'e']
series = pd.Series(scalar, index=index)
print(series)
# Output:
# a    10
# b    10
# c    10
# d    10
# e    10
# dtype: int64
```

© PSSCIVE Draft Material Not be Published

6. range() फ़ंक्शन का उपयोग करके सीरीज का निर्माण

range() फ़ंक्शन संख्याओं की एक क्रम (sequence) बनाने का एक उपयोगी तरीका है।

```
import pandas as pd
series = pd.Series(range(1, 6))  # Creates numbers from 1 to 5
print(series)
# Output:
# 0    1
# 1    2
# 2    3
# 3    4
# 4    5
# dtype: int64
```

7. सूची समझ (List Comprehension) का उपयोग करके सीरीज का निर्माण

सूची समझ पायथन में सूचियाँ बनाने का एक संक्षिप्त तरीका है। आप अपनी सीरीज के लिए डेटा बनाने के लिए उनका उपयोग कर सकते हैं।

```
import pandas as pd
# Create a list of squares of numbers from 0 to 4
data = [x**2 for x in range(5)]
series = pd.Series(data, index=['a', 'b', 'c', 'd', 'e'])
print(series)
# Output:
# a    0
# b    1
# c    4
# d    9
# e    16
# dtype: int64
```

2.3 पांडस सीरीज के तत्वों तक पहुँचना (Accessing Elements of a Pandas Series)

© PSSCIVE Draft Study Not be Published

आप Series से डेटा दो मुख्य तरीकों से प्राप्त कर सकते हैं: उसके स्थान (position) के आधार पर या उसके लेबल (label) के आधार पर।

2.3.1 अनुक्रमण या इंडेक्सिंग (Indexing)

1. स्थिति द्वारा पहुँच या Accessing by position (using [])

आप वर्गाकार कोष्ठक (स्क्वायर ब्रैकेट्स) [] और स्थिति संख्या (पोज़िशन नंबर) का उपयोग कर सकते हैं, बिल्कुल एक सूची (list) की तरह। ध्यान रखें, गिनती 0 से शुरू होती है।

```
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50])

print(series[0])  # Output: 10 (first element)
print(series[2])  # Output: 30 (third element)
```

2. लेबल द्वारा पहुँच या Accessing by label (using [])

यदि आपकी सीरीज में कस्टम लेबल हैं, तो आप [] के अंदर लेबल का उपयोग कर सकते हैं।

```
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50], index=['a', 'b', 'c', 'd', 'e'])

print(series['c'])    # Output: 30
print(series[['a', 'd']]) # Output: a 10, d 40
```

2.3.2 स्लाइसिंग (Slicing)

स्लाइसिंग का अर्थ है सीरीज का एक भाग निकालना। आप [start:end] सिंटैक्स का उपयोग करते हैं।

स्थिति के साथ (With position): end मान *शामिल नहीं* किया जाता है।

```
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50])
print(series[1:4])  # Output: elements at positions 1, 2, and 3 (20, 30, 40)
```

With labels using .loc[]: .loc[] लेबल के साथ स्लाइस करने का अनुशंसित तरीका है। यहाँ, अंतिम लेबल *शामिल* किया जाता है।

```
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50], index=['a', 'b', 'c', 'd', 'e'])
print(series.loc['b':'d'])  # Output: elements with labels b, c, and d (20, 30, 40)
```

© PSSCE Draft Study Material not be Published

With positions using .iloc[]: .iloc[] स्थितियों के साथ स्लाइस करने का अनुशंसित तरीका है।

```
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50], index=['a', 'b', 'c', 'd', 'e'])
print(series.iloc[1:4])  # Output: elements at positions 1, 2, and 3 (20, 30, 40)
```

बूलियन इंडेक्सिंग (Boolean indexing)

आप एक शर्त का उपयोग करके एक सीरीज को फ़िल्टर कर सकते हैं। यह बहुत शक्तिशाली है।

```
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50])
# Get all elements greater than 30
print(series[series > 30])  # Output: 3 40, 4 50
```

2.4 पांडस सीरीज के गुण (Attributes of Pandas Series)

गुण (Attributes) आपकी सीरीज (Series) के बारे में जानकारी (facts) की तरह होते हैं। इन्हें उपयोग करने के लिए parentheses () लगाने की आवश्यकता नहीं होती।

गुण (Attribute)	उद्देश्य (Purpose)	उदाहरण (Example)
.index	सीरीज के लेबल को लौटाता है।	print(series.index)
.values	डेटा को NumPy एरे के रूप में लौटाता है।	print(series.values)
.dtype	तत्वों (elements) का डेटा प्रकार (data type) बताता है।	print(series.dtype)
.size	तत्वों की कुल संख्या लौटाता है।	print(series.size)
.shape	आकार को एक टपल के रूप में लौटाता है (जैसे, (5,))।	print(series.shape)
.name	सीरीज का नाम लौटाता है या सेट करता है।	series.name = "Marks"
.ndim	आयामों की संख्या लौटाता है (हमेशा 1)।	print(series.ndim)
.empty	यदि सीरीज खाली है तो सत्य (True) लौटाता है।	print(series.empty)
.hasnans	यदि कोई लापता मान (missing values) (NaN) है तो सत्य (True) लौटाता है।	print(series.hasnans)

2.5 पांडस सीरीज की विधियाँ (Pandas Series Methods)

विधियाँ (Methods) वे फ़ंक्शन हैं जो सीरीज ऑब्जेक्ट से संबंधित होते हैं और डेटा के साथ कुछ करते हैं। उनके पास कोष्ठक () होते हैं।

© PSSCIVE Draft Study Material Not to be Published

विधि (Method)	उद्देश्य (Purpose)	उदाहरण (Example)
.head(n)	पहली n पंक्तियाँ लौटाता है (डिफ़ॉल्ट 5)।	series.head(3)
.tail(n)	अंतिम n पंक्तियाँ लौटाता है (डिफ़ॉल्ट 5)।	series.tail(3)
.unique()	अद्वितीय मानों का एक एरे लौटाता है।	series.unique()
.nunique()	अद्वितीय मानों की संख्या लौटाता है।	series.nunique()
.count()	गैर-NaN मानों की संख्या लौटाता है।	series.count()
.value_counts()	अद्वितीय मानों की गणना के साथ एक सीरीज लौटाता है।	series.value_counts()

2.6 सीरीज पर गणितीय संक्रियाएँ (Mathematical Operations on Series)

आप सीरीज पर आसानी से गणित कर सकते हैं।

1. अदिश (Scalars) के साथ संक्रियाएँ (Operations with scalars)

आप पूरी सीरीज को एकल संख्या से जोड़, घटा, गुणा या भाग कर सकते हैं। संक्रिया प्रत्येक तत्व पर लागू होती है।

```
Example with Python code
import pandas as pd
series = pd.Series([10, 20, 30, 40, 50])
print(series + 5)  # Output: 15, 25, 35, 45, 55
print(series * 2)  # Output: 20, 40, 60, 80, 100
```

2. दो सीरीज के बीच संक्रियाएँ (Operations between two Series)

जब आप दो सीरीज जोड़ते हैं, तो पांडस स्मार्ट है। यह मानों को उनके **सूचकांक लेबल** के आधार पर जोड़ता है, न कि केवल उनकी स्थिति के आधार पर। इसे **संरेखण (alignment)** कहा जाता है।

```
Example with Python code
import pandas as pd
s1 = pd.Series([10, 20, 30], index=['a', 'b', 'c'])
s2 = pd.Series([1, 2, 3], index=['a', 'c', 'd'])
result = s1 + s2
print(result)
# Output:
# a    11.0   (from a in s1 and a in s2)
# b     NaN   (b is only in s1)
# c    32.0   (c is in s1 and c in s2)
```

```
# d     NaN   (d is only in s2)
# dtype: float64
```

जैसा कि आप देख सकते हैं, यदि कोई सूचकांक दोनों सीरीज में मौजूद नहीं है, तो उस सूचकांक के लिए परिणाम NaN (Not a Number यानी लापता) होता है। आप इन लापता मानों को संभालने के लिए fill_value के साथ .add(), .sub(), .mul(), .div() जैसी विधियों का उपयोग कर सकते हैं।

2.7 पांडस सीरीज मानों पर सांख्यिकीय गणना (Statistical Calculation on Pandas Series Values)

सीरीज में सामान्य आँकड़ों के लिए अंतर्निहित विधियाँ हैं। ये विधियाँ स्वचालित रूप से NaN मानों को अनदेखा करती हैं।

```
Example with Python code
import pandas as pd
import numpy as np
data = pd.Series([10, 20, 30, 40, 50, np.nan])
print("Sum:", data.sum())        # Output: Sum: 150.0
print("Mean:", data.mean())      # Output: Mean: 30.0
print("Median:", data.median()) # Output: Median: 30.0
print("Min:", data.min())        # Output: Min: 10.0
print("Max:", data.max())        # Output: Max: 50.0
print("Std Dev:", data.std())    # Output: Std Dev: 15.81
```

सारांश (Summary)

- Series एक एक-आयामी (one-dimensional) लेबलयुक्त एरे है (जैसे Excel में एक कॉलम)।
- प्रत्येक मान (value) एक इंडेक्स (label) से जुड़ा होता है, जिससे डेटा को प्राप्त करना आसान हो जाता है।

© PSSCIVE Draft Study Material not published

- इसे lists, arrays या dictionaries से बनाया जा सकता है।
- गुण: .dtype, .index, .empty, .hasnans.
- विधियाँ: .head(), .tail(), .count(), .unique(), .value_counts().
- अंकगणितीय संक्रियाओं का समर्थन करता है और स्वचालित रूप से सूचकांक द्वारा संरेखित होता है, लापता मानों को NaN से भरता है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. पांडस सीरीज बनाने के लिए किस फ़ंक्शन का उपयोग किया जाता है? (अ) pd.series() (ब) pd.Series() (स) pandas.create_series() (द) Series()
2. एक खाली पांडस सीरीज बनाने के लिए सही सिंटैक्स है? (अ) s = pd.Series(empty) (ब) s = pd.Series() (स) s = pd.Series(None) (द) Both b and c
3. पायथन शब्दकोश से पांडस सीरीज बनाते समय, शब्दकोश की कुंजियाँ (keys) डिफ़ॉल्ट रूप से क्या बन जाती हैं? (अ) सीरीज के मान (ब) सीरीज का डेटा प्रकार (स) सीरीज के सूचकांक लेबल (द) सीरीज का नाम
4. निम्नलिखित में से कौन सा एक पांडस सीरीज बनाएगा जहाँ सभी तत्वों का मान 5 है, जिसमें सूचकांक लेबल 'a', 'b', और 'c' हैं? (अ) pd.Series(5, index=['a', 'b', 'c']) (ब) pd.Series([5, 5, 5], index=['a', 'b', 'c']) (स) pd.Series({'a': 5, 'b': 5, 'c': 5}) (द) उपरोक्त सभी
5. यदि आप एक NumPy एरे से एक पांडस सीरीज बनाते हैं और सूचकांक निर्दिष्ट नहीं करते हैं, तो डिफ़ॉल्ट सूचकांक क्या होगा? (अ) वर्णमाला लेबल ('A', 'B', 'C'...) (ब) 1 से शुरू होने वाले पूर्णांक लेबल (स) 0 से शुरू होने वाले पूर्णांक लेबल (द) रेंडम रूप से उत्पन्न अद्वितीय लेबल
6. पांडस सीरीज बनाने के लिए डेटा के रूप में निम्नलिखित में से किस डेटा प्रकार या संरचना का उपयोग किया जा सकता है? (अ) एक पायथन सूची (ब) एक पायथन शब्दकोश (स) एक NumPy ndarray (द) उपरोक्त सभी
7. एक सीरीज बनाने के लिए जहाँ प्रत्येक तत्व में एक अदिश मान होता है जो प्रदान किए गए सूचकांक की लंबाई के आधार पर दोहराता है, आपको यह करना होगा: (अ) एक अदिश मान और एक सूचकांक दोनों प्रदान करें। (ब) केवल एक अदिश मान प्रदान करें (स) केवल एक सूचकांक प्रदान करें (द) इस तरह से सीरीज बनाना संभव नहीं है
8. पांडस सीरीज का कौन सा गुण सीरीज में डेटा मानों का एरे लौटाता है? (अ) .index (ब) .values (स) .dtype (द) .name
9. पांडस सीरीज का .index गुण क्या लौटाता है? (अ) डेटा मानों की एक सूची (ब) सीरीज का नाम (स) प्रत्येक तत्व के लिए लेबल युक्त एक पांडस इंडेक्स ऑब्जेक्ट (द) सीरीज में तत्वों की संख्या।

10. किस गुण का उपयोग सीरीज ऑब्जेक्ट के आयामों की संख्या प्राप्त करने के लिए किया जाता है? (अ) .dimensions (ब) .ndim (स) .shape (द) .axis
11. किसी सीरीज का .hasnans गुण सत्य (True) लौटाता है यदि: (अ) सीरीज में केवल पूर्णांक मान हैं (ब) सीरीज में कम से कम एक NaN (लापता) मान है (स) सीरीज का कोई नाम नहीं है (द) सीरीज में एक कस्टम सूचकांक है।
12. कौन सा गुण सीरीज की मेमोरी उपयोग को बाइट्स में प्रदान करता है? (अ) .size (ब) .nbytes (स) .memory (द) .itemsize

ब. रिक्त स्थान भरें

1. एक पांडस सीरीज एक _____-आयामी लेबल वाला एरे है जो किसी भी डेटा प्रकार को धारण करने में सक्षम है।
2. एक पांडस सीरीज के दो प्राथमिक घटक इसके डेटा मान और इसके _____ लेबल हैं।
3. डिफ़ॉल्ट रूप से, यदि किसी सूची या NumPy एरे से सीरीज बनाते समय कोई सूचकांक निर्दिष्ट नहीं किया जाता है, तो पांडस शून्य से शुरू होने वाला एक _____ सूचकांक बनाएगा।
4. पायथन शब्दकोश से सीरीज बनाते समय, शब्दकोश की _____ डिफ़ॉल्ट रूप से सीरीज के सूचकांक लेबल बन जाती हैं।
5. एक सीरीज बनाने के लिए जहाँ सभी तत्वों का समान अदिश मान हो, आपको अदिश मान और _____ तर्क के लिए एक सूची दोनों प्रदान करने की आवश्यकता है।
6. सीरीज से डेटा मानों के अंतर्निहित NumPy एरे को पुनः प्राप्त करने के लिए उपयोग किया जाने वाला गुण ._____ है।
7. गुण ._____ सीरीज में तत्वों के डेटा प्रकार को लौटाता है।
8. असंरेखित सूचकांकों वाली दो सीरीज के बीच गणितीय संक्रियाएँ करते समय, पांडस डिफ़ॉल्ट रूप से लापता स्थितियों को _____ मानों से भर देगा।
9. गुण ._____ सीरीज में तत्वों की कुल संख्या लौटाता है।
10. यदि किसी सीरीज में एक अदिश मान जोड़ा जाता है, तो संक्रिया प्रत्येक तत्व पर _____-वार लागू होती है।
11. गुण ._____ सत्य (True) लौटाता है यदि सीरीज में कम से कम एक NaN मान है।

स. सत्य या असत्य बताएँ

1. पांडस सीरीज के सभी तत्वों का डेटा प्रकार समान होना चाहिए।
2. पांडस सीरीज का सूचकांक हमेशा 0 से शुरू होने वाले पूर्णांकों का एक क्रम होना चाहिए।
3. आप एक पायथन शब्दकोश से एक पांडस सीरीज बना सकते हैं।
4. विभिन्न सूचकांकों वाली दो सीरीज के बीच अंकगणितीय संक्रियाएँ करते समय, संक्रिया केवल सामान्य सूचकांक लेबलों पर ही की जाएगी।
5. सीरीज का .values गुण सूचकांक लेबल लौटाता है।

6. pd.Series() कंस्ट्रक्टर का उपयोग एक खाली सीरीज बनाने के लिए किया जा सकता है।
7. एक सीरीज में एक अदिश मान जोड़ने से अदिश मान सीरीज के प्रत्येक तत्व में जुड़ जाएगा।
8. .dtype गुण सीरीज में तत्वों की संख्या लौटाता है।
9. .name गुण आपको सीरीज को एक वर्णनात्मक नाम निर्दिष्ट करने की अनुमति देता है।
10. यदि आप 5 तत्वों की सूची से एक सीरीज बनाने का प्रयास करते हैं और 4 तत्वों की सूचकांक सूची प्रदान करते हैं, तो इसके परिणामस्वरूप एक त्रुटि उत्पन्न होगी।
11. सीरीज में NaN मान एक लापता या अपरिभाषित डेटा बिंदु को इंगित करता है।

द. लघु उत्तरीय प्रश्न

1. पांडस सीरीज अपने शब्दों में क्या है?
2. एक पांडस सीरीज बनाने हेतु दो मुख्य घटकों के नाम बताइए?
3. आप सीरीज के साथ काम करने के लिए आमतौर पर पांडस लाइब्रेरी को कैसे इम्पोर्ट करते हैं?
4. पायथन शब्दकोश से सीरीज बनाते समय सूचकांक कैसे निर्धारित किया जाता है?
5. एक पांडस सीरीज और एक एकल अदिश मान के बीच गणितीय संक्रिया (जैसे, जोड़, घटाव) करने पर क्या होता है?
6. सीरीज के .dtype गुण का उद्देश्य क्या है?
7. उस डिफ़ॉल्ट सूचकांक का वर्णन करें जो पांडस तब निर्दिष्ट करता है जब बिना स्पष्ट रूप से सूचकांक प्रदान किए किसी सूची या NumPy एरे से एक सीरीज बनाई जाती है।
8. एक सीरीज के भीतर लापता या अपरिभाषित डेटा बिंदुओं का प्रतिनिधित्व करने के लिए पांडस किस मान का उपयोग करता है और इसका विशिष्ट डेटा प्रकार निहितार्थ क्या है?
9. यदि आपके पास अलग-अलग सूचकांक लेबल वाली दो सीरीज हैं, और आप उन्हें एक साथ जोड़ते हैं, तो पांडस सूचकांक संरेखण को कैसे संभालता है, विशेष रूप से उन लेबलों के लिए जो दोनों में सामान्य नहीं हैं?
10. सीरीज के *.values* गुण और *.index* गुण के बीच अंतर संक्षेप में समझाइए।

सत्र 3. पांडस डेटाफ्रेम

(Pandas Dataframe)

एक शिक्षक की ग्रैड शीट की कल्पना कीजिए। इसमें 'Student Name', 'Math Marks', 'Science Marks' जैसे कॉलम होते हैं और प्रत्येक विद्यार्थी के लिए अलग-अलग पंक्तियाँ में होती हैं। जोकि टेबल एक उत्तम उदाहरण है। Pandas का DataFrame भी बिल्कुल ऐसा ही होता है—यह एक शक्तिशाली, डिजिटल अंक तालिका का रूप है। इस सत्र में, आप सीखेंगे कि DataFrame को कैसे बनाया जाए, उसे कैसे प्रबंधित किया जाए और उससे जानकारी कैसे प्राप्त की जाए।

3.1 परिचय

एक पांडस डेटाफ्रेम एक दो-आयामी, लेबल वाली डेटा संरचना है। आप इसे एक स्प्रेडशीट या SQL टेबल के रूप में कल्पना कर सकते हैं। यह डेटा विश्लेषण के लिए सबसे अधिक उपयोग की जाने वाली पांडस वस्तु है।

डेटाफ्रेम की प्रमुख विशेषताएँ:

- यह एक तालिका की तरह पंक्तियों और स्तंभों में संगठित हॉट है।
- लेबल वाले अक्ष: पंक्तियों का एक सूचकांक (पंक्ति संख्याओं की तरह) होता है और स्तंभों के नाम होते हैं।
- विभिन्न डेटा प्रकार: प्रत्येक स्तंभ एक अलग प्रकार का डेटा रख सकता है (जैसे, 'अंक' में संख्याएँ, 'नाम' में पाठ)।

3.2 पांडस डेटाफ्रेम बनाना (Creating Pandas Dataframe)

आप कई तरीकों से एक डेटाफ्रेम बना सकते हैं। यहाँ सबसे आसान तरीके दिया गया हैं।

1. एक खाली डेटाफ्रेम बनाना (Creating an Empty DataFrame)

```
Python
import pandas as pd
df =
pd.DataFrame()
print(df)
# Output:
# Empty DataFrame
# Columns: []
# Index: []
```

2. शब्दकोशों की सूची से डेटाफ्रेम बनाना (Creating a DataFrame from a List of Dictionaries)

यह बहुत आम है। प्रत्येक शब्दकोश एक पंक्ति है और कुंजियाँ स्तंभ नाम बन जाती हैं।

```
Python
import pandas as pd
data = [{'Name': 'Asha', 'Marks': 85}, {'Name': 'Boby', 'Marks': 92}]
df = pd.DataFrame(data)
print(df)
# Output:
#    Name  Marks
# 0  Asha     85
# 1  Boby     92
```

3. सूचियों के शब्दकोश से डेटाफ्रेम बनाना (Creating a DataFrame from a Dictionary of Lists)

इस विधि में, प्रत्येक कुंजी-मान युग्म एक स्तंभ बन जाता है। कुंजी स्तंभ का नाम है और सूची स्तंभ का डेटा है।

© PSSCIVE Draft Study Material Not be Published

```
Python
import pandas as pd
data = {
    'Name': ['Asha', 'Boby', 'Diya'],
    'Marks': [85, 92, 78],
    'City': ['Nagpur', 'Bhopal', 'Pune']
}
df = pd.DataFrame(data)
print(df)
# Output:
#    Name  Marks    City
# 0  Asha     85  Nagpur
# 1  Boby     92  Bhopal
# 2  Diya     78    Pune
```

4. सीरीज के शब्दकोश से डेटाफ्रेम बनाना (Creating a DataFrame from a Dictionary of Series)

यहाँ प्रत्येक सीरीज एक स्तंभ बन जाती है। सीरीज का सूचकांक डेटाफ्रेम की पंक्ति सूचकांक बन जाता है।

```
Python
import pandas as pd
data = {
    'Phy': pd.Series([78, 89, 78], index=['a', 'b', 'c']),
    'Che': pd.Series([87, 90, 79], index=['a', 'b', 'c'])
}
df = pd.DataFrame(data)
print(df)
# Output:
#    Phy  Che
# a   78   87
# b   89   90
# c   78   79
```

3.3 पंक्तियों और स्तंभों पर संक्रियाएँ (Operations on Rows and Columns)

© PSSCIVE Draft Study Material Not be Published

आइए पंक्तियों और स्तंभों को जोड़ने, बदलने और हटाने का तरीका सीखने के लिए उपरोक्त उदाहरण 3 (Name, Marks, City वाला df) का उपयोग करें।

a) एक नया स्तंभ जोड़ना (Adding a New Column)

एक नया स्तंभ जोड़ना बहुत आसान है। बस इसे एक नाम दें और मान निर्दिष्ट करें।

```
Python
df['Grade'] = ['A', 'A+', 'B']
print(df)
# Output:
#    Name  Marks    City Grade
# 0  Asha     85  Nagpur     A
# 1  Boby     92  Bhopal    A+
# 2  Diya     78    Pune     B
```

b) एक नई पंक्ति जोड़ना (Adding a New Row)

एक नई पंक्ति जोड़ने के लिए, आप .loc[] विधि का उपयोग कर सकते हैं। यदि सूचकांक मौजूद नहीं है, तो इसे जोड़ दिया जाएगा।

```
Python
df.loc[3] = ['Deepak', 95, 'Mumbai', 'A+']
print(df)
# Output:
#     Name  Marks    City Grade
# 0    Asha     85  Nagpur     A
# 1    Boby     92  Bhopal    A+
# 2    Diya     78    Pune     B
# 3  Deepak     95  Mumbai    A+
```

c) एक पंक्ति या स्तंभ हटाना (Deleting a Row or Column)

.drop() विधि का उपयोग करें। एक पंक्ति हटाने के लिए, axis=0 का उपयोग करें। एक स्तंभ हटाने के लिए, axis=1 का उपयोग करें।

```
Python
# Delete the row with index 2 (Diya)
df = df.drop(2, axis=0)
```

© PSSCIVE Draft Study Material Not be Published

```
print(df)
# Output:
#      Name  Marks    City Grade
# 0    Asha     85  Nagpur     A
# 1    Boby     92  Bhopal    A+
# 3  Deepak     95  Mumbai    A+

# Delete the 'City' column
df = df.drop('City', axis=1)
print(df)
# Output:
#      Name  Marks Grade
# 0    Asha     85     A
# 1    Boby     92    A+
# 3  Deepak     95    A+
```

d) स्तंभ लेबल का नाम बदलना (Renaming Column Labels)

.rename() विधि आपको स्तंभ नाम बदलने देती है।

```
Python
df = df.rename(columns={'Marks': 'Percentage', 'Grade': 'Letter'})
print(df)
# Output:
#      Name  Percentage Letter
# 0    Asha          85      A
# 1    Boby          92     A+
# 3  Deepak          95     A+
```

3.4 डेटाफ्रेम तत्वों तक पहुँचना (Accessing Dataframe Elements)

पांडस आपको डेटाफ्रेम से विशिष्ट डेटा प्राप्त करने के स्मार्ट तरीके देता है।

1. .loc[] का उपयोग करना (लेबल-आधारित) (Using .loc[] (Label-based))

आप पंक्ति के सूचकांक लेबल और स्तंभ के नाम से डेटा का चयन करते हैं।

© PSSCIVE Draft Study Material Not be Published

```
Python
# Select row with index '1' and the 'Name' column
print(df.loc[1, 'Name'])  # Output: Boby

# Select rows 0 to 1 and columns 'Name' and 'Percentage'
print(df.loc[0:1, ['Name', 'Percentage']])
# Output:
#    Name  Percentage
# 0  Asha         85
# 1  Boby         92
```

2. .iloc[] का उपयोग करना (स्थिति-आधारित) (Using .iloc[] (Position-based))

आप पंक्ति और स्तंभ की पूर्णांक स्थिति (सूची की तरह, 0 से शुरू) द्वारा डेटा का चयन करते हैं।

```
Python
# Select the element at row position 0 and column position 1 (which is 'Percentage')
print(df.iloc[0, 1])  # Output: 85

# Select the first two rows and the first column
print(df.iloc[0:2, 0])
# Output:
# 0    Asha
# 1    Boby
# Name: Name, dtype: object
```

3. बूलियन अनुक्रमण (फ़िल्टरिंग) (Boolean Indexing (Filtering))

यह एक शक्तिशाली तरीका है जो किसी विशिष्ट शर्त को पूरा करने वाली पंक्तियों का चयन करता है।

```
Python
# Select all rows where 'Percentage' is greater than 90
high_scorers = df[df['Percentage'] > 90]
print(high_scorers)
# Output:
```

SCIVE Draft Study Material Not Published

```
#     Name  Percentage Letter
# 1   Boby        92     A+
# 3  Deepak       95     A+
```

3.5 पांडस डेटाफ्रेम के गुण (Attributes of Pandas Dataframes)

गुण ऐसे गुण हैं जो आपको डेटाफ्रेम के बारे में कुछ बताते हैं। उनमें कोष्ठक () नहीं होते हैं।

गुण (Attribute)	उद्देश्य (Purpose)	हमारे df पर उदाहरण (Example on our df)
.shape	आपको पंक्तियों और स्तंभों की संख्या बताता है।	print(df.shape) -> (3, 3)
.size	आपको तत्वों की कुल संख्या बताता है।	print(df.size) -> 9
.ndim	आपको आयामों की संख्या बताता है (हमेशा 2)।	print(df.ndim) -> 2
.dtypes	आपको प्रत्येक स्तंभ का डेटा प्रकार बताता है।	print(df.dtypes)
.columns	आपको स्तंभ नाम देता है।	print(df.columns) -> Index(['Name', 'Percentage', 'Letter'], dtype='object')
.index	आपको पंक्ति लेबल (सूचकांक) देता है।	print(df.index) -> Index([0, 1, 3], dtype='int64')
.values	आपको डेटा को 2D NumPy एरे के रूप में देता है।	print(df.values) -> [['Asha' 85 'A'] ['Boby' 92 'A+'] ['Deepak' 95 'A+']]

CTWE Draft Study Material Not be Published

सारांश (Summary)

- एक डेटाफ्रेम एक 2D, लेबल वाली डेटा संरचना है, जो एक तालिका की तरह है।
- आप इसे सूचियों, शब्दकोशों या सीरीज से बना सकते हैं।
- आप आसानी से पंक्तियाँ और स्तंभ जोड़/हटा सकते हैं।
- लेबल-आधारित पहुँच के लिए .loc[] और स्थिति-आधारित पहुँच के लिए .iloc[] का उपयोग कीजिए।
- शर्तों के आधार पर डेटा फ़िल्टर करने के लिए बूलियन अनुक्रमण का उपयोग कीजिए।
- .shape, .columns, और .dtypes जैसे गुण आपको डेटाफ्रेम के बारे में जानकारी देते हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. निम्नलिखित में से कौन सा पांडस डेटाफ्रेम बनाने का वैध तरीका *नहीं* है? (अ) सीरीज के शब्दकोश से (ब) NumPy ndarray से (स) बिना सूचकांक और स्तंभ निर्दिष्ट किए एक अदिश मान से (द) शब्दकोशों की सूची से।
2. डेटाफ्रेम df से 'Age' नामक एकल स्तंभ का चयन करने के लिए, निम्नलिखित में से कौन सा सही सिंटैक्स है? (अ) df.Age (ब) df['Age'] (स) df.loc[:, 'Age'] (द) उपरोक्त सभी
3. आप डेटाफ्रेम में लेबल द्वारा पंक्तियों या स्तंभों के समूह तक कैसे पहुँचते हैं? (अ) iloc (ब) loc (स) at (द) iat
4. कौन सा गुण डेटाफ्रेम के आयाम (पंक्तियाँ, स्तंभ) को दर्शाने वाला एक टपल लौटाता है? (अ) .size (ब) .ndim (स) .shape (द) .axes
5. डेटाफ्रेम df की पहली n पंक्तियाँ लौटाने के लिए, आप किस विधि का उपयोग करेंगे? (अ) df.first(n) (ब) df.top(n) (स) df.head(n) (द) df.start(n)
6. डेटाफ्रेम का .columns गुण क्या लौटाता है? (अ) पंक्ति सूचकांक लेबलों की एक सूची (ब) स्तंभ लेबल (स्तंभ नाम) की एक सूची (स) स्तंभों के डेटा प्रकार (द) स्तंभों की संख्या।
7. यदि आप एक शब्दकोश से एक डेटाफ्रेम बनाते हैं जहाँ कुंजियाँ स्तंभ नाम हैं और मान असमान लंबाई की सूचियाँ हैं, तो क्या होगा? (अ) एक त्रुटि उत्पन्न होगी (ब) छोटी सूचियों को 0 से भर दिया जाएगा (स) छोटी सूचियों को NaN से भर दिया जाएगा (द) डेटाफ्रेम केवल सबसे छोटी सूची की लंबाई के साथ बनाया जाएगा
8. डेटाफ्रेम से लापता मानों (NaN) वाली पंक्तियों को हटाने के लिए किस विधि का उपयोग किया जाता है? (अ) df.drop_missing() (ब) df.remove_nan() (स) df.dropna() (द) df.fill_na()
9. डेटाफ्रेम df का ट्रांसपोज़ प्राप्त करने के लिए, आप लिख सकते हैं: (अ) df.Transpose() (ब) df.T (स) df.swap_axes() (द) df.transpose_dataframe()
10. कौन सा गुण आपको डेटाफ्रेम में पंक्तियों और स्तंभों की संख्या बताता है? (अ) .size (ब) .count (स) .shape (द) .ndim

ब. रिक्त स्थान भरें

1. डेटाफ्रेम पंक्तियों और _____ से मिलकर बनते हैं।
2. पांडस लाइब्रेरी को आयात करते समय उपयोग किया जाने वाला सामान्य उपनाम pd है, जैसे import pandas as _____।
3. सूचियों के शब्दकोश से डेटाफ्रेम बनाने के लिए, शब्दकोश की कुंजियाँ आमतौर पर डेटाफ्रेम के _____ नाम बन जाती हैं।
4. गुण ._____ एक टपल लौटाता है जो डेटाफ्रेम में पंक्तियों और स्तंभों की संख्या को इंगित करता है।
5. डेटाफ्रेम df की पहली 5 पंक्तियों को प्रदर्शित करने के लिए, आप df._____ विधि का उपयोग करेंगे।

6. डेटाफ्रेम my_df से 'Price' नामक स्तंभ का चयन करने के लिए, आप my_df[_____] का उपयोग कर सकते हैं।
7. गुण ._____ डेटाफ्रेम के स्तंभ लेबल (नाम) की एक सूची प्रदान करता है।
8. जब आप दो डेटाफ्रेम के बीच गणितीय संक्रियाएँ करते हैं, तो पांडस डेटा को पंक्ति _____ और स्तंभ _____ दोनों के आधार पर संरेखित करता है।
9. गुण ._____ डेटाफ्रेम की पंक्ति लेबल (सूचकांक) की एक सूची प्रदान करता है।
10. गुण ._____ डेटाफ्रेम में प्रत्येक स्तंभ का डेटा प्रकार लौटाता है, आमतौर पर एक सीरीज के रूप में।

स. सत्य या असत्य बताएँ

1. पांडस डेटाफ्रेम के सभी स्तंभों का डेटा प्रकार समान होना चाहिए।
2. आप सीरीज के शब्दकोश से एक डेटाफ्रेम बना सकते हैं।
3. डेटाफ्रेम का .index गुण स्तंभ लेबल लौटाता है।
4. df.head() विधि डिफ़ॉल्ट रूप से डेटाफ्रेम की पहली 5 पंक्तियाँ प्रदर्शित करती है।
5. iloc एक्सेसर का उपयोग पूर्णांक-स्थान-आधारित अनुक्रमण के आधार पर डेटा का चयन करने के लिए किया जाता है।
6. df.shape गुण डेटाफ्रेम में तत्वों की कुल संख्या लौटाता है।
7. डेटाफ्रेम में लापता मानों को आमतौर पर स्ट्रिंग "NA" द्वारा दर्शाया जाता है।
8. आप किसी मौजूदा डेटाफ्रेम में एक नया स्तंभ जोड़ सकते हैं।
9. df.T गुण डेटाफ्रेम का ट्रांसपोज़ लौटाता है (पंक्तियों और स्तंभों को स्वैप करता है)।
10. जब एक शब्दकोश से एक डेटाफ्रेम बनाया जाता है जहाँ मान असमान लंबाई की सूचियाँ हैं, तो हमेशा एक त्रुटि उत्पन्न होगी।

द. लघु उत्तरीय प्रश्न

1. पांडस सीरीज और पांडस डेटाफ्रेम के बीच मूलभूत अंतर क्या है?
2. पांडस डेटाफ्रेम बनाने के दो सामान्य तरीकों के नाम बताइए।
3. डेटाफ्रेम के .shape गुण के उद्देश्य की व्याख्या कीजिए।
4. डेटाफ्रेम से डेटा का चयन करते समय loc और iloc एक्सेसर के बीच प्राथमिक अंतर क्या है?
5. head() विधि कब उपयोगी होती है और यह डिफ़ॉल्ट रूप से क्या प्रदर्शित करती है?
6. df.columns गुण क्या प्रदान करता है?
7. पांडस डेटाफ्रेम में लापता मानों का प्रतिनिधित्व कैसे किया जाता है और उन्हें रखने वाली पंक्तियों या स्तंभों को हटाने के लिए किस विधि का उपयोग किया जा सकता है?
8. df.describe() विधि क्या आउटपुट देती है, और यह किस डेटा प्रकार के लिए सबसे अधिक प्रासंगिक है?

9. fillna() विधि की कार्यक्षमता का संक्षेप में वर्णन कीजिए।
10. पांडस गलत संरेखित सूचकांकों और/या स्तंभों वाले दो डेटाफ्रेम के बीच अंकगणितीय संक्रियाओं (जैसे जोड़) को कैसे संभालता है?

© PSSCIVE Draft Study Material Not be Published

सत्र 4. मैटप्लोटलिब का उपयोग करके डेटा विज़ुअलाइज़ेशन

(Data Visualization Using Matplotlib)

कल्पना कीजिए कि आपके पास पिछले 10 मैचों के लिए आपकी पसंदीदा क्रिकेट टीम के स्कोर हैं। संख्याओं की सूची को देखकर, एक पैटर्न देखना मुश्किल है। लेकिन यदि आप एक लाइन ग्राफ बनाते हैं, तो आप तुरंत देख सकते हैं कि टीम में सुधार हो रहा है या नहीं। यह संख्याओं को चित्रों में बदलना ही डेटा विज़ुअलाइज़ेशन है।

इस सत्र में, आप सीखेंगे कि विभिन्न प्रकार के चार्ट बनाने और उन्हें स्पष्ट और पेशेवर दिखने के लिए अनुकूलित करने के लिए Matplotlib का उपयोग कैसे करें।

4.1 परिचय

डेटा विज़ुअलाइज़ेशन सूचना और डेटा का चित्रमय प्रतिनिधित्व है। चार्ट, ग्राफ और मानचित्र जैसे दृश्य तत्वों का उपयोग करके, डेटा विज़ुअलाइज़ेशन उपकरण डेटा में रुझानों, बाहरी मूल्यों और पैटर्न को देखने और समझने का एक सुलभ तरीका प्रदान करते हैं।

4.2 मैटप्लोटलिब (Matplotlib)

Matplotlib स्थिर, एनिमेटेड और इंटरैक्टिव विज़ुअलाइज़ेशन बनाने के लिए सबसे लोकप्रिय पायथन लाइब्रेरी है। यह आपको बहुत अधिक नियंत्रण देता है कि आपके चार्ट कैसे दिखते हैं।

4.2.1 मैटप्लोटलिब इंस्टाल करना (Installing Matplotlib)

पांडस की तरह, आप अपने कमांड प्रॉम्प्ट में pip का उपयोग करके Matplotlib इंस्टाल कर सकते हैं:

Bash

```
pip install matplotlib
```

4.2.2 पाइप्लॉट (Pyplot) इम्पोर्ट करना

प्लॉटिंग के लिए Matplotlib का उपयोग करने के लिए, हम pyplot नामक एक विशिष्ट मॉड्यूल इम्पोर्ट करते हैं। हम आमतौर पर इसे एक छोटा उपनाम plt देते हैं।

Python

```python
import matplotlib.pyplot as plt
```

4.3 पाइप्लॉट के साथ मूल प्लॉटिंग (Basic Plotting with Pyplot)

आइए एक सरल लाइन चार्ट बनाएँ।

उदाहरण 4.1: दैनिक तापमान के लिए एक सरल लाइन चार्ट।

Python

```python
import matplotlib.pyplot as plt


days = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
```

```
temp = [35, 37, 38, 36, 39, 37, 38]  # temperatures in Celsius


plt.plot(days, temp)  # Create the line chart
plt.show()          # Display the chart
```

यह एक मूल लाइन ग्राफ बनाता है। plot() फ़ंक्शन लाइन खींचता है और show() विंडो को दिखाता है।

4.3.1 अपने प्लॉट को अनुकूलित करना (Customizing Your Plot)

शीर्षक और लेबल वाला चार्ट समझने में बहुत आसान होता है।

उदाहरण 4.2: शीर्षक, लेबल और एक ग्रिड जोड़ना।

```
Python
import matplotlib.pyplot as plt


days = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
temp = [35, 37, 38, 36, 39, 37, 38]


plt.plot(days, temp)


# Customizations
plt.xlabel("Day of the Week")
plt.ylabel("Temperature in °C")
plt.title("Weekly Temperature Report")
plt.grid(True)  # Adds a grid for easier reading


plt.show()
```

PSSCIVE Draft Study Material Not be Publis

4.4 विभिन्न प्रकार के चार्ट (Different Types of Charts)

Matplotlib कई अलग-अलग प्रकार के चार्ट बना सकता है। यहाँ सबसे उपयोगी चार्ट दिए गए हैं।

4.4.1. लाइन चार्ट (Line Chart)

समय के साथ रुझान दिखाने के लिए सबसे अच्छा।

```
Python
# Example from 4.3.1
```

4.4.2. बार चार्ट (Bar Chart)

विभिन्न श्रेणियों की तुलना करने के लिए सबसे अच्छा।

```
Python
import matplotlib.pyplot as plt

categories = ['Apples', 'Bananas', 'Oranges']
sales = [30, 45, 20]

plt.bar(categories, sales)
plt.xlabel('Fruit Type')
plt.ylabel('Units Sold')
plt.title('Fruit Sales')
plt.show()
```

4.4.3. स्कैटर प्लॉट (Scatter Plot)

दो चीजों के बीच संबंध दिखाने के लिए सबसे अच्छा।

```
Python
import matplotlib.pyplot as plt

# Hours studied vs. Exam score
hours_studied = [2, 3, 1, 4, 5, 2, 3, 4]
exam_score = [65, 70, 50, 80, 90, 60, 72, 85]

plt.scatter(hours_studied, exam_score)
plt.xlabel('Hours Studied')
plt.ylabel('Exam Score')
plt.title('Effect of Studying on Exam Scores')
plt.show()
```

4.4.4. हिस्टोग्राम (Histogram)

यह दिखाने के लिए सबसे अच्छा है कि डेटा कैसे वितरित किया गया है (जैसे, कितने विद्यार्थी प्रत्येक ग्रेड रेंज में आते हैं)।

© PSSCIVE Draft Study Material Not be Published

```
Python
import matplotlib.pyplot as plt

marks = [35, 28, 55, 62, 78, 41, 15, 85, 92, 58, 65, 71, 30, 48, 70, 25, 95, 50, 60, 75]

plt.hist(marks, bins=5, edgecolor='black')  # 'bins' are the number of groups
plt.xlabel('Marks Ranges')
plt.ylabel('Number of Students')
plt.title('Distribution of Marks')
plt.show()
```

4.4.5. पाई चार्ट (Pie Chart)

एक पूर्ण के भागों (प्रतिशत) को दिखाने के लिए सबसे अच्छा।

```
Python
import matplotlib.pyplot as plt

grades = ['A', 'B', 'C', 'D']
num_students = [40, 35, 15, 5]

plt.pie(num_students, labels=grades, autopct="%1.1f%%")  # autopct shows percentages
plt.title('Student Grades Distribution')
plt.show()
```

4.4.6. बॉक्स प्लॉट (Box Plot)

डेटा के प्रसार (न्यूनतम, अधिकतम, माध्यिका, और बाहरी मान) को दिखाने के लिए सबसे अच्छा।

```
Python
import matplotlib.pyplot as plt
import numpy as np

english = [78, 36, 89, 87, 79, 32, 57, 68, 65, 82]
maths = [78, 66, 39, 87, 79, 32, 57, 68, 65, 80]
```

```
science = [78, 59, 89, 87, 79, 62, 57, 68, 60, 82]

data_to_plot = [english, maths, science]

plt.boxplot(data_to_plot, labels=['English', 'Maths', 'Science'])
plt.ylabel('Marks')
plt.title('Subject-wise Marks Distribution')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

blished

सारांश (Summary)

- डेटा विज़ुअलाइज़ेशन आसान समझ के लिए डेटा को चित्रों में बदल देता है।
- Matplotlib चार्ट बनाने के लिए एक पायथन लाइब्रेरी है।
- पाइप्लॉट (plt) वह मॉड्यूल है जिसका उपयोग हम प्लॉटिंग के लिए करते हैं।
- विभिन्न कार्यों के लिए अलग-अलग चार्ट होते हैं: रुझानों के लिए लाइन चार्ट, तुलनाओं के लिए बार चार्ट, संबंधों के लिए स्कैटर प्लॉट, आदि।
- आप प्लॉट को शीर्षकों (plt.title()), लेबल (plt.xlabel/ylabel), लेजेंड (plt.legend()) और ग्रिड (plt.grid()) के साथ अनुकूलित कर सकते हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. Matplotlib का प्राथमिक उद्देश्य क्या है? (अ) डेटा हेरफेर और विश्लेषण (ब) मशीन लर्निंग मॉडल बनाना (स) पायथन में स्थिर, एनिमेटेड और इंटरैक्टिव विज़ुअलाइज़ेशन बनाना (द) वेब विकास
2. Matplotlib के भीतर किस मॉड्यूल को आमतौर पर प्लॉट बनाने के लिए इम्पोर्ट किया जाता है, आमतौर पर उपनाम plt के साथ? (अ) matplotlib.graph (ब) matplotlib.figure (स) matplotlib.pyplot (द) matplotlib.visualize
3. Matplotlib में एक सरल लाइन प्लॉट बनाने के लिए किस फ़ंक्शन का उपयोग किया जाता है? (अ) plt.line() (ब) plt.plot() (स) plt.draw() (द) plt.graph()
4. प्लॉट को स्क्रीन पर प्रदर्शित करने के लिए, प्लॉट बनाने के बाद किस फ़ंक्शन को कॉल किया जाना चाहिए? (अ) plt.show() (ब) plt.display() (स) plt.render() (द) plt.view()

5. आप Matplotlib में प्लॉट का शीर्षक कैसे सेट कर सकते हैं? (अ) plt.set_title("My Plot") (ब) plt.plot_title("My Plot") (स) plt.title("My Plot") (द) plt.add_title("My Plot")
6. Matplotlib प्लॉट में x-अक्ष पर लेबल जोड़ने के लिए किस फ़ंक्शन का उपयोग किया जाता है? (अ) plt.ylabel() (ब) plt.x_label() (स) plt.set_xlabel() (द) plt.xlabel()
7. Matplotlib प्लॉट को छवि फ़ाइल (जैसे, PNG, JPEG) के रूप में सहेजने के लिए, किस विधि का उपयोग किया जाता है? (अ) plt.save() (ब) plt.export() (स) plt.save_image() (द) plt.savefig()
8. plt.legend() फ़ंक्शन का उद्देश्य क्या है? (अ) प्लॉट में एक शीर्षक जोड़ना। (ब) प्लॉट पर ग्रिड लाइनें प्रदर्शित करना। (स) प्लॉट के भीतर विभिन्न तत्वों (लाइनों, बार आदि) को लेबल करना (द) प्लॉट का पृष्ठभूमि रंग बदलना।
9. स्कैटर प्लॉट बनाने के लिए निम्नलिखित में से किस फ़ंक्शन का उपयोग किया जाता है? (अ) plt.line() (ब) plt.scatter() (स) plt.plot(kind='scatter') (द) Both b and c
10. आप Matplotlib में ग्राफ पर ग्रिड लाइनें कैसे जोड़ सकते हैं? (अ) plt.gridlines() (ब) plt.add_grid() (स) plt.grid() (द) plt.show_grid()

ब. रिक्त स्थान भरें

1. Matplotlib का उपयोग मुख्य रूप से पायथन में स्थिर, एनिमेटेड और _____ विज़ुअलाइज़ेशन बनाने के लिए किया जाता है।
2. Matplotlib के भीतर प्लॉटिंग फलनों के लिए आमतौर पर आयात किया जाने वाला मॉड्यूल matplotlib._____ है।
3. एक सरल लाइन प्लॉट बनाने के लिए, फ़ंक्शन plt._____ का उपयोग किया जाता है।
4. एक प्लॉट बनाने के बाद, प्लॉट विंडो प्रदर्शित करने के लिए आपको plt._____ को कॉल करना होगा।
5. एक प्लॉट में शीर्षक जोड़ने के लिए, फ़ंक्शन plt._____() का उपयोग किया जाता है।
6. y-अक्ष पर लेबल जोड़ने के लिए फ़ंक्शन plt._____() का उपयोग किया जाता है।
7. वर्तमान प्लॉट को छवि फ़ाइल (जैसे, PNG, PDF) के रूप में सहेजने के लिए, फ़ंक्शन plt._____() का उपयोग किया जाता है।
8. प्लॉट पर विभिन्न लाइनों या तत्वों के लिए लेबल प्रदर्शित करने के लिए फ़ंक्शन plt._____() का उपयोग किया जाता है।
9. स्कैटर प्लॉट बनाने के लिए, आमतौर पर फ़ंक्शन plt._____() का उपयोग किया जाता है।
10. आप प्लॉट में ग्रिड लाइनें जोड़ सकते हैं, फ़ंक्शन plt._____() का उपयोग करके।

स. सत्य या असत्य बताएँ

1. Matplotlib का उपयोग मुख्य रूप से एरे के साथ संख्यात्मक गणना के लिए किया जाता है।
2. matplotlib.pyplot मॉड्यूल Matplotlib की एक उप-लाइब्रेरी है।
3. plt.show() फ़ंक्शन वैकल्पिक है और प्लॉट प्रदर्शित करने के लिए इसे कॉल करने की आवश्यकता नहीं है।
4. आप plt.title() का उपयोग करके एक प्लॉट का शीर्षक सेट कर सकते हैं।

5. Matplotlib केवल 2D प्लॉट बनाने में सक्षम है।
6. plt.xlabel() और plt.ylabel() फलनों का उपयोग क्रमशः x और y अक्षों पर लेबल जोड़ने के लिए किया जाता है।
7. plt.legend() फ़ंक्शन का उपयोग प्लॉट में ग्रिड लाइनें जोड़ने के लिए किया जाता है।
8. आप plt.savefig() का उपयोग करके Matplotlib प्लॉट को PNG छवि फ़ाइल के रूप में सहेज सकते हैं।
9. Matplotlib प्लॉट हमेशा स्थिर होते हैं और उन्हें इंटरैक्टिव नहीं बनाया जा सकता है।
10. plt.grid() फ़ंक्शन का उपयोग प्लॉट में ग्रिड जोड़ने के लिए किया जा सकता है।

द. लघु उत्तरीय प्रश्न

1. पायथन में Matplotlib लाइब्रेरी का मुख्य उद्देश्य क्या है?
2. Matplotlib का कौन सा मॉड्यूल MATLAB-जैसे इंटरफ़ेस में प्लॉट बनाने के लिए सबसे अधिक उपयोग किया जाता है?
3. Matplotlib का उपयोग करके एक सरल प्लॉट बनाने और प्रदर्शित करने में शामिल मूल चरण क्या हैं?
4. आप Matplotlib प्लॉट में शीर्षक और x तथा y अक्षों के लिए लेबल कैसे जोड़ सकते हैं?
5. Matplotlib में स्कैटर प्लॉट और लाइन प्लॉट के बीच अंतर स्पष्ट कीजिए।
6. आप Matplotlib प्लॉट को फ़ाइल में सहेजने के लिए किस फ़ंक्शन का उपयोग करेंगे और आप इसे किस प्रकार की फ़ाइलों के रूप में सहेज सकते हैं?
7. plt.legend() फ़ंक्शन का उद्देश्य क्या है?
8. आप Matplotlib का उपयोग करके एक ही आकृति (figure) के भीतर कई प्लॉट कैसे बनाते हैं?
9. plt.grid() फ़ंक्शन के उद्देश्य का संक्षेप में वर्णन कीजिए।
10. Matplotlib में "प्लॉट को अनुकूलित करना" का क्या अर्थ है और यह महत्वपूर्ण क्यों है?

© PSSCIVE Dra

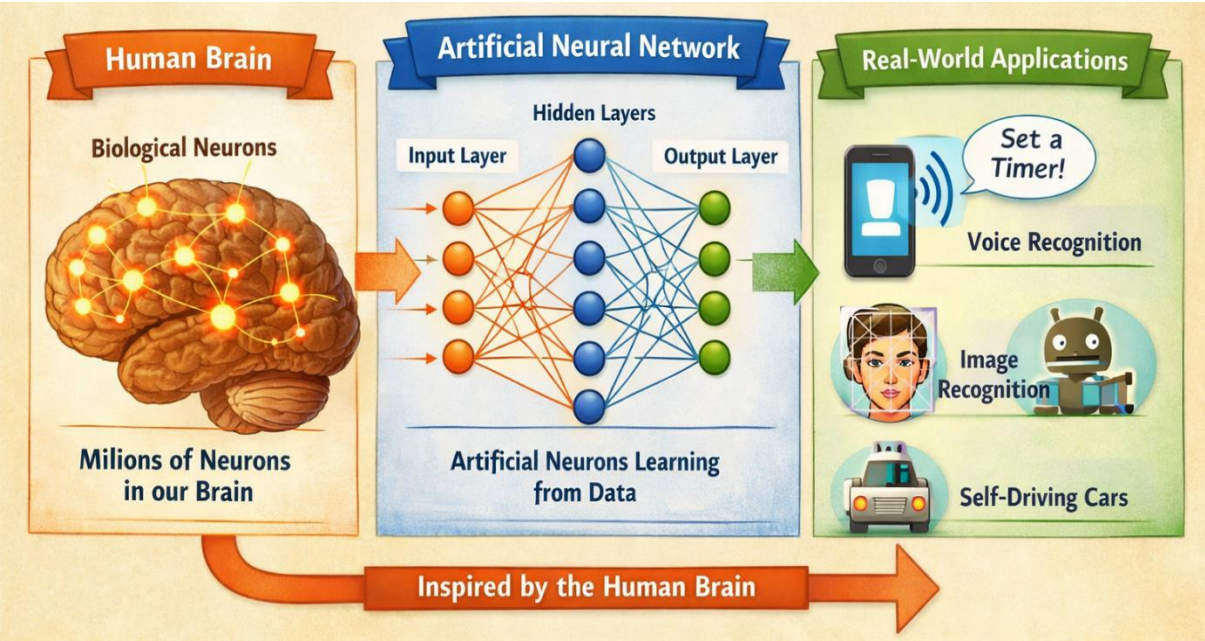
मॉड्यूल 4. न्यूरल नेटवर्क

(Neural Network)

मॉड्यूल संक्षिप्त परिचय (Module Overview)

अपने दिन के बारे में सोचिए। जब आप दूर से किसी दोस्त को देखते हैं, तो आप तुरंत पहचान लेते हैं—हर छोटी-छोटी बात जाँचकर नहीं, बल्कि उनकी मुस्कान, चलने का तरीका या आवाज़ देखकर। जब आप अपने फोन से कहते हैं, "10 मिनट का टाइमर सेट करो," तो वह आपको समझकर काम कर देता है। आपका फोन यह सब कैसे करता है?

इन स्मार्ट कार्यों के पीछे **न्यूरल (तंत्रिका) नेटवर्क (Neural Networks)** होते हैं। ये ऐसे कंप्यूटर प्रोग्राम हैं, जो हमारे मस्तिष्क (brain) के काम करने के तरीके से प्रेरित होते हैं। जैसे हमारा मस्तिष्क लाखों जुड़े हुए कोशिकाओं (neurons) का उपयोग करके सोचता और याद रखता है, उसी तरह एक न्यूरल नेटवर्क कृत्रिम न्यूरॉन्स (artificial neurons) का उपयोग करके डेटा से सीखता है और निर्णय लेता है।



चित्र 4.0 न्यूरल (तंत्रिका) नेटवर्क *को समझना*

इस इकाई के अंत तक, आप इन स्मार्ट प्रणालियों के पीछे के जादू को समझ जाएंगे और यहां तक कि पायथन का उपयोग करके स्वयं एक सरल न्यूरल (तंत्रिका) नेटवर्क बनाना भी सीखेंगे।

अधिगम के परिणाम (Learning Outcomes)

इस मॉड्यूल को पूरा करने के बाद, आप निम्न में सक्षम होंगे:

- न्यूरल नेटवर्क की मूल अवधारणा और वे मानव मस्तिष्क से कैसे प्रेरित हैं, इसे समझेंगे।
- सूचना प्रसंस्करण में कृत्रिम न्यूरॉन की भूमिका की व्याख्या करना।
- पहचानना कि वास्तविक जीवन के अनुप्रयोगों में न्यूरल नेटवर्क का उपयोग कैसे किया जाता है।

- सक्रियण फलनों के महत्व को समझना।
- न्यूरल नेटवर्क मॉडल को प्रशिक्षित करने में शामिल चरणों को सीखना।
- पायथन लाइब्रेरी का उपयोग करके सरल न्यूरल नेटवर्क मॉडल बनाना।
- बुनियादी वर्गीकरण समस्याओं के लिए न्यूरल नेटवर्क लागू करना।

मॉड्यूल संरचना (Module Structure)

सत्र 1. न्यूरल नेटवर्क का परिचय (Introduction to Neural Networks)

सत्र 2. एक न्यूरॉन के अंदर: सक्रियण फ़ंक्शन (Inside A Neuron: Activation Functions)

सत्र 3. क्रियाशील न्यूरल नेटवर्क: वास्तविक दुनिया के अनुप्रयोग (Neural Networks in Action: Real-World Applications)

सत्र 4. पायथन लाइब्रेरी के साथ न्यूरल नेटवर्क का निर्माण (Building Neural Networks with Python Libraries)

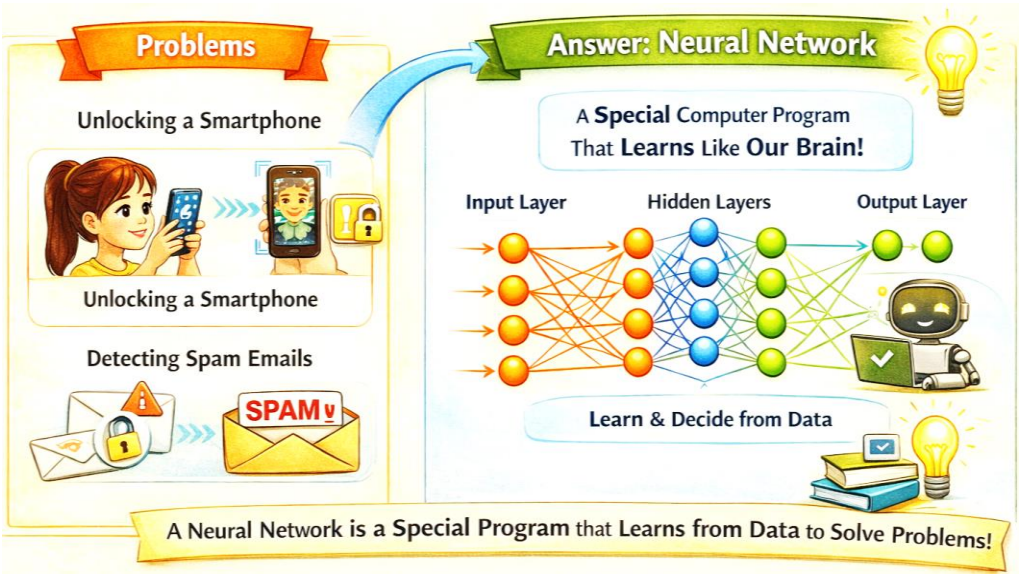
सत्र 5. डेटा तैयार करना और अपने मॉडल को प्रशिक्षित करना (Preparing Data and Training Your Model)

सत्र 6. एक वर्गीकरण मॉडल का निर्माण (Building A Classification Model)

सत्र 1. न्यूरल नेटवर्क का परिचय

(Introduction to Neural Networks)

क्या आपने कभी सोचा है कि स्मार्टफोन सिर्फ एक नज़र से कैसे अनलॉक हो जाता है? या ईमेल ऐप कैसे जानते हैं कि कौन से संदेश स्पैम हैं? इसका उत्तर एक विशेष प्रकार का कंप्यूटर प्रोग्राम है जिसे न्यूरल नेटवर्क (Neural Network) कहा जाता है।



चित्र 1.1: न्यूरल नेटवर्क

इस सत्र में, हम जानेंगे कि न्यूरल नेटवर्क क्या हैं, वे हमारे मस्तिष्क से कैसे प्रेरित हैं और वे डेटा से कैसे सीखते हैं।

1.1 न्यूरल नेटवर्क की अवधारणा (Concept of Neural Network)

एक **न्यूरल नेटवर्क (Neural Network)** कृत्रिम बुद्धिमत्ता (AI) की एक विधि है जो कंप्यूटरों को डेटा को उस तरह से संसाधित करना सिखाती है जो मानव मस्तिष्क से प्रेरित है। यह एक प्रकार का मशीन लर्निंग मॉडल है जो पैटर्न पहचानना सीखता है।

नियमों के एक निश्चित समूह का पालन करने के बजाय, एक न्यूरल नेटवर्क **उदाहरणों** से सीखता है। इसे बिल्ली की पहचान करने के लिए एक बच्चे को पढ़ाने की तरह समझें। आप उन्हें एक सख्त परिभाषा नहीं देते हैं; आप उन्हें तब तक बिल्लियों की कई तस्वीरें दिखाते हैं जब तक वे समझ नहीं जाते कि बिल्ली कैसी दिखती है। एक न्यूरल नेटवर्क डेटा के साथ भी ऐसा ही करता है।

1.2 न्यूरल नेटवर्क के वास्तविक जीवन अनुप्रयोग (Real-Life Applications of Neural Networks)

तंत्रिका नेटवर्क केवल एक प्रयोगशाला प्रयोग नहीं हैं; वे हमारे चारों ओर हैं, प्रौद्योगिकी को स्मार्ट बना रहे हैं। कुछ सामान्य उदाहरण चित्र 1.2 में दिखाए गए हैं।

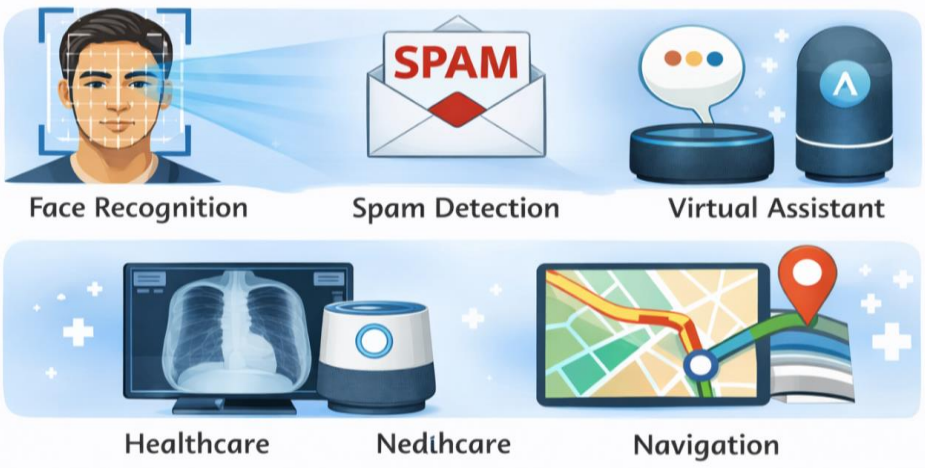
चेहरा पहचान (Face Recognition): आपका फोन या फेसबुक इसका उपयोग फ़ोटो में लोगों की पहचान करने के लिए करता है।

स्पैम का पता लगाना (Spam Detection): जीमेल इसका उपयोग अवांछित ईमेल को स्वचालित रूप से आपके स्पैम फ़ोल्डर में भेजने के लिए करता है।

आभासी सहायक (Virtual Assistants): सिरी (Siri), एलेक्सा (Alexa) और गूगल असिस्टेंट (Google Assistant) आपकी आवाज़ को समझने और आपके सवालों का जवाब देने के लिए इसका उपयोग करते हैं।

स्वास्थ्य सेवा (Healthcare): डॉक्टर एक्स-रे जैसी चिकित्सा छवियों में बीमारियों का पता लगाने में सहायता के लिए AI का उपयोग करते हैं।

नेविगेशन (Navigation): गूगल मैप्स जैसे ऐप इसका उपयोग ट्रैफ़िक की भविष्यवाणी करने और सबसे तेज़ मार्ग खोजने के लिए करते हैं।



चित्र 1.2: न्यूरल नेटवर्क के वास्तविक जीवन अनुप्रयोग

ये उदाहरण दिखाते हैं कि कैसे न्यूरल नेटवर्क कंप्यूटरों को अधिक मानव-जैसे तरीके से देखने, सुनने और दुनिया को समझने में मदद करते हैं।

1.3 प्रेरणा: जैविक बनाम कृत्रिम न्यूरॉन (The Inspiration: Biological vs. Artificial Neurons)

कृत्रिम न्यूरल नेटवर्क का विचार मानव मस्तिष्क के अध्ययन से आया है।

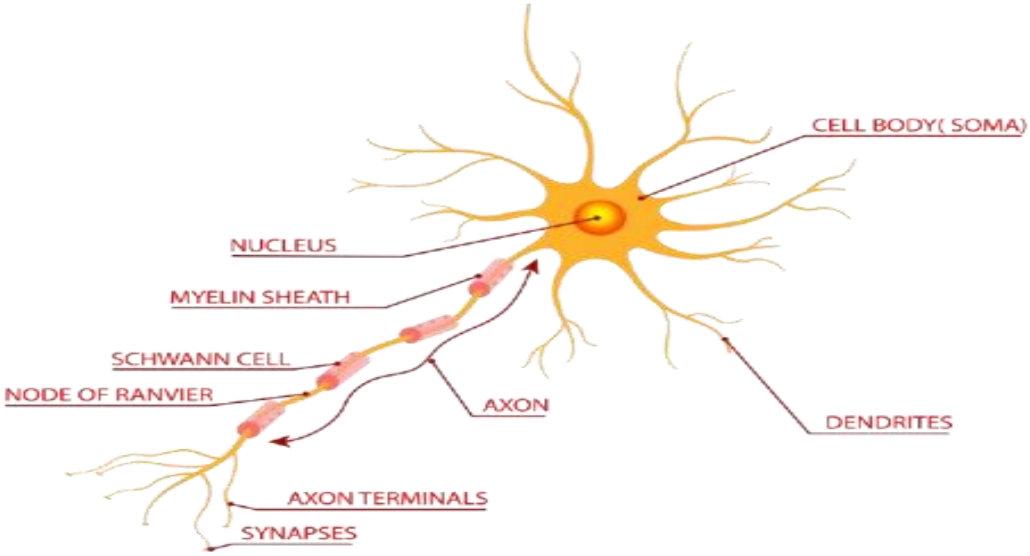
1.3.1 जैविक न्यूरॉन (Biological Neuron) (मस्तिष्क कोशिका)

हमारे मस्तिष्क में अरबों सूक्ष्म न्यूरल कोशिकाएँ होती हैं जिन्हें **न्यूरॉन (neurons)** कहा जाता है। जैसा कि चित्र 4.1.2 में दिखाया गया है, एक न्यूरॉन के तीन मुख्य भाग होते हैं:

डेंड्राइट्स (Dendrites): ये शाखाएँ अन्य न्यूरॉन्स से संकेत प्राप्त करती हैं (जैसे कान जानकारी सुन रहे हों)।

सेल बॉडी (Soma) (कोशिका शरीर): यह भाग आने वाले संकेतों को संसाधित करता है (जैसे मस्तिष्क सोच रहा हो)।

एक्सॉन (Axon): यह लंबी पूंछ संसाधित संकेत को अगले न्यूरॉन तक पहुँचाती है (जैसे मुँह बोल रहा हो)।

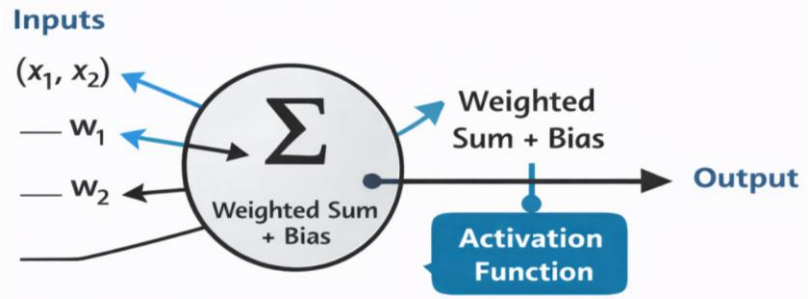


चित्र 1.3: सूक्ष्म न्यूरल कोशिकाएँ न्यूरॉन के रूप में

1.3.2 कृत्रिम न्यूरॉन (Artificial Neuron) (कंप्यूटर की कोशिका)

एक **कृत्रिम न्यूरॉन (Artificial Neuron)** , जिसे **नोड (node)** भी कहा जाता है, एक सरल गणितीय मॉडल है जो जैविक न्यूरॉन की नकल करता है, जैसा कि चित्र 1.3 में दिखाया गया है।

- यह एक या अधिक **इनपुट (inputs)** (जैसे **x1**, **x2**) प्राप्त करता है।
- यह प्रत्येक इनपुट को **भार (weight)** (**w1**, **w2**) नामक संख्या से गुणा करता है, जो न्यूरॉन को बताता है कि वह इनपुट कितना महत्वपूर्ण है।
- यह उन सभी को एक साथ जोड़ता है, साथ ही एक विशेष मान जिसे **बायस (bias)** कहा जाता है।
- फिर यह तय करने के लिए **सक्रियण फ़ंक्शन (activation function)** का उपयोग करता है कि अंतिम **आउटपुट (output)** क्या होगा, जिसे फिर अगले न्यूरॉन को भेज दिया जाता है।



चित्र 1.4: एक कृत्रिम न्यूरॉन की संरचना

तालिका 1.1 दिखाती है कि एक जैविक न्यूरॉन के भागों की तुलना कृत्रिम न्यूरॉन से कैसे की जाती है।

तालिका 1.1: जैविक न्यूरॉन बनाम कृत्रिम न्यूरॉन

जैविक न्यूरॉन (Biological Neuron)	**कृत्रिम न्यूरॉन (Artificial Neuron)**
डेंड्राइट्स (रिसीवर)	इनपुट (संख्यात्मक डेटा)
सोमा (प्रोसेसर)	(भार * इनपुट) + बायस का योग
एक्सॉन (प्रेषक)	सक्रियण फ़ंक्शन के बाद आउटपुट

1.4 न्यूरल नेटवर्क कैसे काम करता है? (How Does a Neural Network Work?)

एक न्यूरल नेटवर्क का "मस्तिष्क" उसकी संरचना और सीखने का तरीका है।

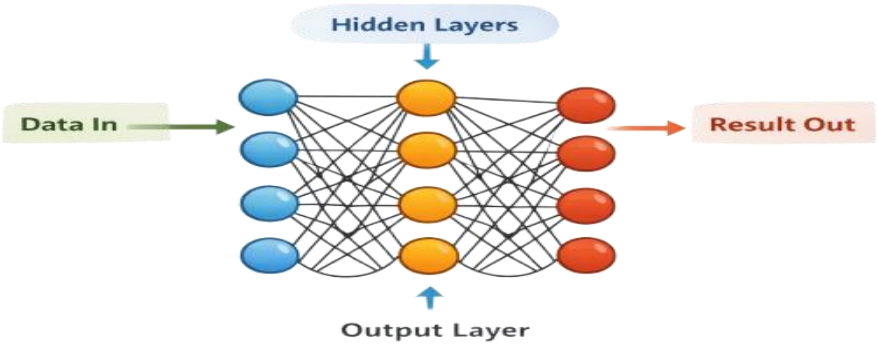
1.4.1 न्यूरल नेटवर्क की संरचना (Structure of a Neural Network)

एक बुनियादी न्यूरल नेटवर्क परतों से बना होता है, जैसा कि चित्र 4.1.4 में दिखाया गया है।

इनपुट परत (Input Layer): यह वह जगह है जहाँ डेटा नेटवर्क में प्रवेश करता है।

छिपी हुई परत(एँ) (Hidden Layer(s)): ये इनपुट और आउटपुट के बीच की परतें हैं। वे पैटर्न खोजने का भारी काम करती हैं। उन्हें "छिपी हुई" कहा जाता है क्योंकि हम सीधे उनके साथ बातचीत नहीं करते हैं।

आउटपुट परत (Output Layer): यह परत अंतिम परिणाम देती है, जैसे "बिल्ली" या "कुत्ता"।



चित्र 1.5 न्यूरल नेटवर्क की मूल संरचना दिखाता है

1.4.2 फीडफॉरवर्ड प्रसार (Feedforward Propagation) (अग्रगामी पास)

यह नेटवर्क के माध्यम से डेटा की यात्रा है।

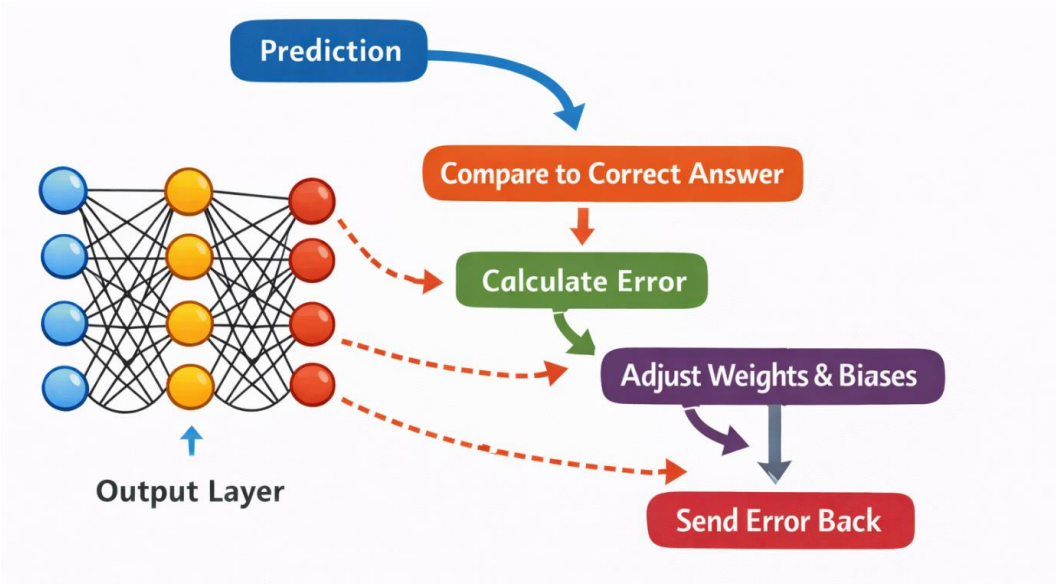
1. डेटा इनपुट परत में प्रवेश करता है।

2. यह छिपी हुई परतों के माध्यम से आगे बढ़ता है। प्रत्येक न्यूरॉन में, इनपुट को उनके भार से गुणा किया जाता है, जोड़ा जाता है और एक बायस जोड़ा जाता है।
3. एक **सक्रियण फ़ंक्शन (activation function)** तय करता है कि क्या संकेत अगली परत तक पहुँचने के लिए पर्याप्त मजबूत है।
4. यह तब तक जारी रहता है जब तक डेटा आउटपुट परत तक नहीं पहुँच जाता, जो भविष्यवाणी (prediction) उत्पन्न करती है।

1.4.3 बैकप्रोपगेशन (Backpropagation) (गलतियों से सीखना)

क्या होता है यदि भविष्यवाणी गलत है? नेटवर्क बैकप्रोपगेशन नामक प्रक्रिया के माध्यम से अपनी गलती से सीखता है (चित्र 1.5 देखें)।

1. नेटवर्क अपनी भविष्यवाणी की तुलना सही उत्तर से करता है और त्रुटि की गणना करता है।
2. यह त्रुटि नेटवर्क के माध्यम से पीछे की ओर भेजी जाती है।
3. जैसे-जैसे यह पीछे जाती है, नेटवर्क सभी भारों और बायस को थोड़ा-थोड़ा समायोजित करता है ताकि अगली बार त्रुटि थोड़ी कम हो।
4. प्रशिक्षण (training) के दौरान इस प्रक्रिया को हजारों बार दोहराया जाता है जब तक कि नेटवर्क भविष्यवाणियाँ करने में बहुत अच्छा नहीं हो जाता।



चित्र 1.6: बैकप्रोपगेशन का उपयोग करके एक न्यूरल नेटवर्क में सीखने की प्रक्रिया

1.4.4 भार (Weights) और बायस (Bias) क्यों महत्वपूर्ण हैं?

भार (Weights): उन्हें वॉल्यूम नॉब (volume knobs) की तरह समझें। किसी इनपुट पर अधिक भार का मतलब है कि वह अंतिम निर्णय के लिए अधिक महत्वपूर्ण है।

बायस (Bias): इसे एक समायोजन नॉब (adjustment knob) की तरह समझें। यह न्यूरॉन को तब भी सक्रिय होने की अनुमति देता है जब सभी इनपुट कमजोर हों, जिससे मॉडल को अधिक लचीलापन मिलता है।

तंत्रिका नेटवर्क को प्रशिक्षित करने का पूरा उद्देश्य सटीक भविष्यवाणियाँ करने के लिए भार और बायस का सही सेट ढूंढना है।

व्यावहारिक गतिविधि 1.1. जैविक न्यूरॉन और कृत्रिम न्यूरॉन का चित्रण और तुलना

उद्देश्य

जैविक न्यूरॉन और कृत्रिम न्यूरॉन की संरचना को समझना और उनके कार्यों की तुलना करना।

आवश्यक सामग्री

- A4 शीट या नोटबुक
- पेंसिल और रबर
- स्केल
- रंगीन पेंसिल (वैकल्पिक)

प्रक्रिया

1. पृष्ठ के बाईं ओर एक जैविक न्यूरॉन बनाएँ।
    - निम्नलिखित भागों को लेबल करें:
    - डेंड्राइट्स
    - कोशिका शरीर (सोमा)
    - केन्द्रक (न्यूक्लियस)
    - एक्सॉन
    - एक्सॉन टर्मिनल्स
2. पृष्ठ के दाईं ओर, एक कृत्रिम न्यूरॉन मॉडल बनाएँ।
    - निम्नलिखित भागों को लेबल करें:
    - इनपुट संकेत
    - भार
    - योग (Σ)
    - सक्रियण फ़ंक्शन
    - आउटपुट
3. यह दिखाने के लिए तीरों का उपयोग करें कि संकेत न्यूरॉन के माध्यम से कैसे यात्रा करते हैं।

दोनों आरेखों को ध्यान से देखें और जैविक और कृत्रिम न्यूरॉन के बीच दो समानताएँ और दो अंतर लिखें।

अवलोकन तालिका

विशेषता	जैविक न्यूरॉन	कृत्रिम न्यूरॉन

इनपुट	डेंड्राइट्स संकेत प्राप्त करते हैं	इनपुट डेटा मान
प्रसंस्करण	कोशिका शरीर संकेतों को संसाधित करता है	योग और सक्रियण फ़ंक्शन
आउटपुट	एक्सॉन संकेत भेजता है	आउटपुट मान
प्रणाली	मानव मस्तिष्क का भाग	कंप्यूटरों में न्यूरल नेटवर्क का भाग

परिणाम / निष्कर्ष

एक जैविक न्यूरॉन मानव मस्तिष्क में एक प्राकृतिक कोशिका है जो विद्युत संकेतों के माध्यम से सूचना को संसाधित और प्रसारित करती है।

एक कृत्रिम न्यूरॉन कंप्यूटरों में उपयोग किया जाने वाला एक गणितीय मॉडल है जो इनपुट प्राप्त करता है, उन्हें संसाधित करता है और मानव मस्तिष्क के काम करने के समान एक आउटपुट उत्पन्न करता है।

सारांश (Summary)

इस सत्र में न्यूरल नेटवर्क्स को मानव मस्तिष्क से प्रेरित प्रणालियों के रूप में परिचित कराया गया है। विद्यार्थी इसमें विभिन्न परतों (layers) जैसे इनपुट लेयर (input layer), हिडन लेयर (hidden layer) और आउटपुट लेयर (output layer) के बारे में सीखते हैं, तथा यह समझते हैं कि नेटवर्क जानकारी को कैसे प्रोसेस करते हैं और डेटा से पैटर्न (patterns) कैसे सीखते हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. न्यूरल (तंत्रिका) नेटवर्क का प्राथमिक कार्य क्या है? (अ) उच्च-गति गणना करना (ब) डेटा में पैटर्न पहचानना (स) बड़ी मात्रा में डेटा संग्रहीत करना (द) इंटरनेट से कनेक्ट करना
2. एक जैविक न्यूरॉन में, कौन सा भाग अन्य न्यूरॉन्स से संकेत प्राप्त करता है? (अ) एक्सॉन (ब) सोमा (स) डेंड्राइट्स (द) न्यूक्लियस
3. एक कृत्रिम न्यूरॉन में भार (weights) की क्या भूमिका होती है? (अ) अंतिम आउटपुट संग्रहीत करना (ब) किसी इनपुट के महत्व को निर्धारित करना (स) प्रशिक्षण प्रक्रिया शुरू करना (द) इंटरनेट से कनेक्ट करना
4. इनपुट से आउटपुट परत तक डेटा अग्रेषित करने की प्रक्रिया को कहा जाता है (अ) बैकप्रोपगेशन (ब) फीडफॉरवर्ड (स) डीप लर्निंग (द) डेटा विभाजन
5. न्यूरल (तंत्रिका) नेटवर्क का वास्तविक दुनिया में उपयोग निम्नलिखित में से कौन सा है? (अ) MS Word में दस्तावेज़ टाइप करना (ब) कैलकुलेटर ऐप का उपयोग करना (स) अपने चेहरे से फोन अनलॉक करना (द) फ़ाइल एक्सप्लोरर ब्राउज़ करना

ब. रिक्त स्थान भरें

1. एक न्यूरल नेटवर्क _____के काम करने के तरीके से प्रेरित है।
2. न्यूरल (तंत्रिका) नेटवर्क की पहली परत को _____परत कहा जाता है।
3. जो संख्याएँ किसी इनपुट के महत्व को निर्धारित करती हैं, उन्हें _____कहा जाता है।
4. त्रुटि को पीछे भेजकर भारों को समायोजित करने की विधि को _____कहा जाता है।
5. न्यूरल (तंत्रिका) नेटवर्क का अंतिम परिणाम _____परत द्वारा निर्मित होता है।

स. सत्य या असत्य बताएँ

1. न्यूरल (तंत्रिका) नेटवर्क नियमों के एक निश्चित समूह का पालन करते हैं जो कभी नहीं बदलते।
2. ईमेल में स्पैम का पता लगाना न्यूरल नेटवर्क का एक अनुप्रयोग है।
3. वास्तविक सीखना छिपी हुई परतों में होता है।
4. भार में एक बदलाव कभी भी भविष्यवाणी को प्रभावित नहीं करता है।
5. एक जैविक न्यूरॉन और एक कृत्रिम न्यूरॉन अवधारणात्मक रूप से समान तरीके से काम करते हैं।

द. लघु उत्तरीय प्रश्न

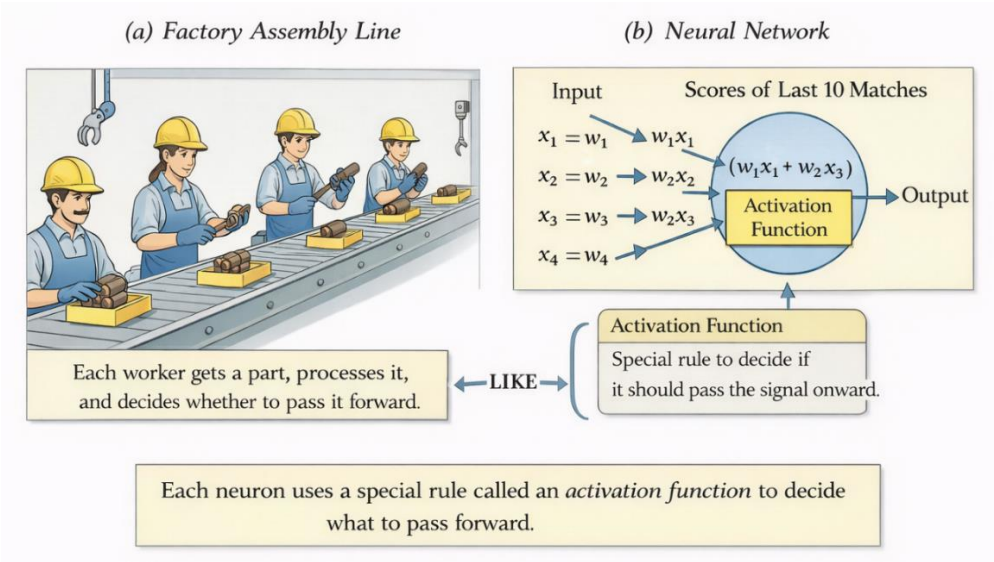
1. एक जैविक न्यूरॉन के तीन मुख्य भागों के नाम बताइए।
2. एक कृत्रिम न्यूरल नेटवर्क क्या है इसे अपने शब्दों में लिखिए ?
3. एक न्यूरॉन में बायस पद (bias term) क्यों महत्वपूर्ण है?
4. दैनिक जीवन में आप न्यूरल नेटवर्क के साथ कैसे बातचीत करते हैं? इसके दो उदाहरण दीजिए।
5. फीडफॉरवर्ड और बैकप्रोपगेशन के बीच अंतर संक्षेप में समझाइए।

© PSSCIVE Draft S

सत्र 2. एक न्यूरॉन के अंदर: सक्रियण फ़ंक्शन

(Inside A Neuron: Activation Functions)

एक फैक्ट्री की असेंबली लाइन की कल्पना कीजिए। यंहा हर कर्मचारी (worker) एक हिस्सा प्राप्त करता है, उस पर काम करता है और तय करता है कि उसे अगले व्यक्ति को भेजना है या नहीं। न्यूरल नेटवर्क में हर न्यूरॉन भी यही काम करता है। लेकिन वह यह कैसे तय करता है कि आगे क्या भेजना है? इसके लिए वह एक विशेष गणितीय नियम का उपयोग करता है, जिसे **सक्रियण (एक्टिवेशन) फ़ंक्शन (Activation Function)** कहते है।

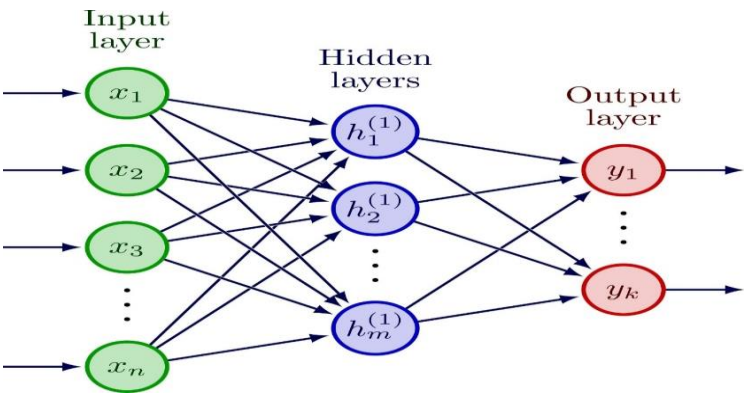


चित्र 2.1: सादृश्य के माध्यम से सक्रियण फ़ंक्शन की व्याख्या

इस सत्र में हम न्यूरल नेटवर्क की संरचना (structure), भार (weights) और बायस (bias) की भूमिका तथा विभिन्न प्रकार के सक्रियण या एक्टिवेशन फ़ंक्शन्स का अध्ययन करेंगे।

2.1 न्यूरल नेटवर्क की संरचना (The Structure of a Neural Network)

जैसा कि हमने सीखा, एक न्यूरल नेटवर्क परतों से बना होता है। इसे एक बहु-चरणीय फिल्टर की तरह समझें। इनपुट परत कच्चा माल है, छिपी हुई परतें वे मशीनें हैं जो इसे परिष्कृत करती हैं और आउटपुट परत अंतिम उत्पाद है। चित्र 2.2 इस संरचना को दर्शाता है।



चित्र 2.2: एक न्यूरल नेटवर्क की परतें

इनपुट परत (Input Layer): डेटा यहाँ प्रवेश करता है। यदि आप हस्तलिखित अंक की छवि का उपयोग कर रहे हैं, तो प्रत्येक पिक्सेल की चमक एक इनपुट है।

छिपी हुई परते (Hidden Layer(s)): ये परतें पैटर्न ढूंढती हैं। पहली छिपी हुई परत किसी छवि में किनारों का पता लगा सकती है। अगली परत उन किनारों को आकृतियों में जोड़ सकती है।

आउटपुट परत (Output Layer): यह अंतिम उत्तर देती है, जैसे कि छवि के '8' अंक होने की संभावना।

2.2 कृत्रिम न्यूरॉन: एक करीब से देखें (The Artificial Neuron: A Closer Look)

एक कृत्रिम न्यूरॉन एक सरल गणना इकाई है। इसका काम इनपुट लेना, उन्हें संसाधित करना और एक आउटपुट उत्पन्न करना है।

2.2.1 एक न्यूरॉन सूचना कैसे संसाधित करता है (गणितीय अभिव्यक्ति)

आउटपुट = सक्रियण फ़ंक्शन ($w_1x_1 + w_2x_2 + ... + b$)

आइए इसे समझें:

इनपुट (x_1, x_2): न्यूरॉन को दिए गए मान, जैसे किसी छवि में पिक्सेल मान या किसी विद्यार्थी द्वारा पढ़े गए घंटों की संख्या।

भार (w_1, w_2): संख्याएँ जो प्रत्येक इनपुट के महत्व को दर्शाती हैं। प्रशिक्षण के दौरान इन भारों को समायोजित किया जाता है ताकि न्यूरल नेटवर्क डेटा में पैटर्न सीख सके।

बायस (b): भारित इनपुट के योग में जोड़ा गया एक स्थिर मान। बायस न्यूरॉन को आउटपुट समायोजित करने में मदद करता है और मॉडल को अधिक लचीला बनाता है।

2.2.2 वास्तविक जीवन सादृश्य: फिल्म चुनना (Real-Life Analogy: Choosing a Movie)

कल्पना कीजिए कि एक न्यूरॉन यह तय करता है कि क्या आपको कोई फिल्म पसंद आएगी, जो दो इनपुट पर आधारित है: **has_funny_actors (x1)** and **is_action_packed (x2)** ।

- यदि आप **एक्शन** पसंद करते हैं, तो **x2** (**w2**) के लिए भार अधिक होगा।
- **b** आपके सामान्य मूड की तरह है। यदि आप ऊब चुके हैं, तो किसी भी फिल्म को पसंद करने का आपका 'बायस' अधिक हो सकता है।

न्यूरॉन कुल स्कोर की गणना करता है और सक्रियण फ़ंक्शन तय करता है कि क्या यह स्कोर "हाँ, मुझे यह फिल्म पसंद आएगी!" कहने के लिए पर्याप्त है।

2.3 सक्रियण फ़ंक्शन: निर्णय निर्माता (Activation Functions: The Decision-Maker)

एक सक्रियण फ़ंक्शन एक गणितीय द्वार है जो तय करता है कि अगली परत को क्या भेजा जाए। यह **गैर-रैखिकता (non-linearity)** का परिचय देता है, जो नेटवर्क को जटिल पैटर्न सीखने की अनुमति देता है, न कि केवल सरल सीधी रेखाओं को।

2.3.1 सिग्मॉइड फ़ंक्शन (Sigmoid Function) (संभाव्यता निर्माता)

यह फ़ंक्शन किसी भी संख्या को लेता है और उसे **0 और 1** के बीच के मान में दबा देता है। यह उन निर्णयों के लिए एकदम सही है जहाँ उत्तर एक संभाव्यता है, जैसे "क्या यह बिल्ली है?" (1 हाँ के लिए, 0 नहीं के लिए)।

$f(x) = 1 / (1 + e^{-x})$

*(**e** के बारे में चिंता न करें, यह सिर्फ एक विशेष गणितीय स्थिरांक है ≈ 2.718)*

उदाहरण:

- यदि किसी न्यूरॉन का कुल स्कोर एक बड़ी धनात्मक संख्या है, तो सिग्मॉइड इसे 1 के करीब बना देता है।
- यदि स्कोर एक बड़ी ऋणात्मक संख्या है, तो यह इसे 0 के करीब बना देता है।
- 0 का स्कोर बिल्कुल 0.5 बन जाता है।

2.3.2 ReLU (लोकप्रिय किड) (The Popular Kid)

ReLU (Rectified Linear Unit) सबसे लोकप्रिय सक्रियण फ़ंक्शन है। इसका नियम सरल है: यदि इनपुट धनात्मक है, तो उसे पास कर दो। यदि यह ऋणात्मक है, तो उसे रोक दो और 0 आउटपुट करो।

$f(x) = \max(0, x)$

उदाहरण:

- 5 का इनपुट 5 का आउटपुट देता है।
- -3 का इनपुट 0 का आउटपुट देता है।
- यह तेज़ है और नेटवर्क को जटिल पैटर्न कुशलतापूर्वक सीखने में मदद करता है।

2.3.3 सॉफ्टमैक्स (Softmax) (संभाव्यता वितरक)

सॉफ्टमैक्स विशेष है। इसका उपयोग आउटपुट परत में तब किया जाता है जब हमारे पास **दो से अधिक श्रेणियाँ (categories)** हों। यह अंतिम परत से स्कोर लेता है और उन्हें संभाव्यताओं में बदल देता है जिनका योग 1 होता है।

$\text{Softmax}(x_i) = e^{x_i} / \Sigma e^{x_j}$

या

$\text{Softmax}(x_i) = e^{x_i} / (e^{x_1} + e^{x_2} + \ldots + e^{x_n})$

उदाहरण:

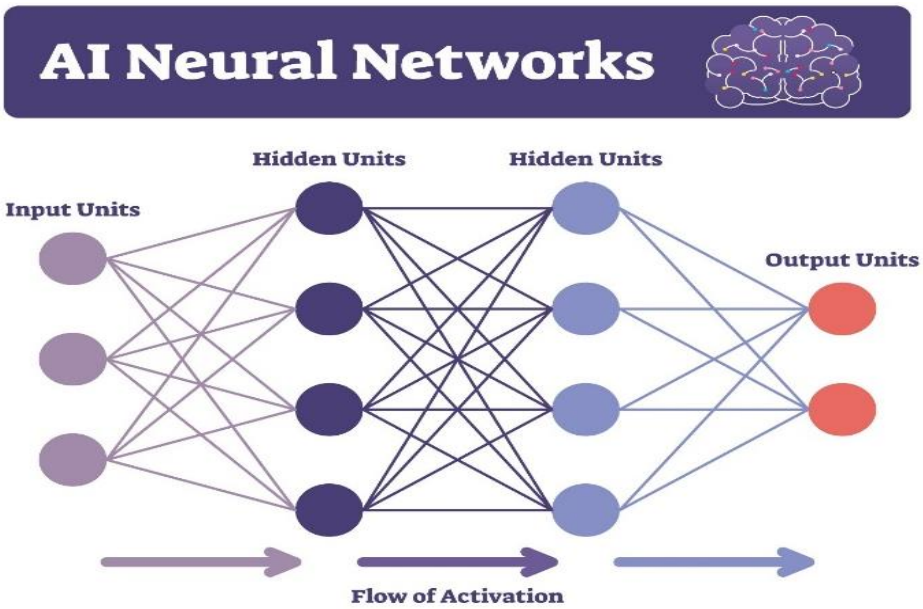
यदि एक नेटवर्क किसी फल को सेब, केला या संतरे के रूप में वर्गीकृत करने का प्रयास कर रहा है, तो अंतिम स्कोर **[2.0, 1.0, 0.1]** हो सकते हैं। सॉफ्टमैक्स इसे **[0.7, 0.2, 0.1]** जैसी संभाव्यताओं में बदल देगा, जिसका अर्थ है कि नेटवर्क 70% सुनिश्चित है कि यह एक सेब है।

2.4 न्यूरल नेटवर्क के प्रकार (Types of Neural Networks)

उनकी संरचना के आधार पर, न्यूरल नेटवर्क विभिन्न प्रकार के हो सकते हैं।

2.4.1 फीडफॉरवर्ड न्यूरल नेटवर्क (Feedforward Neural Network (FNN))

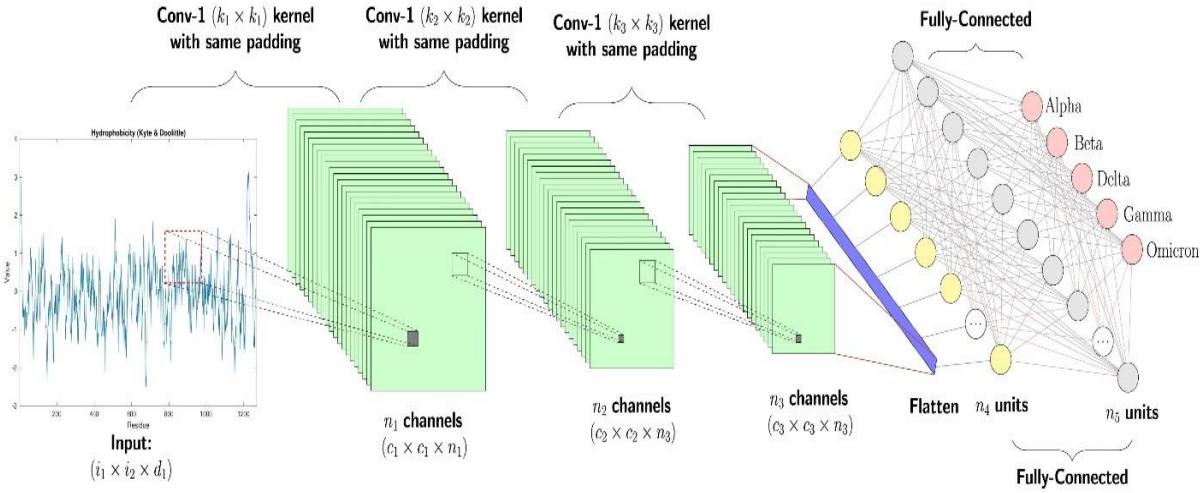
यह सबसे सरल प्रकार है। डेटा केवल एक ही दिशा में यात्रा करता है: इनपुट से आउटपुट तक। कोई लूप या फीडबैक कनेक्शन नहीं होते हैं। यह एक वन-वे सड़क की तरह है। (चित्र 2.3)



चित्र 2.3: AI न्यूरल नेटवर्क संरचना

2.4.2 कनवोल्यूशनल न्यूरल नेटवर्क (Convolutional Neural Network (CNN))

CNN विशेष रूप से छवि डेटा के लिए डिज़ाइन किए गए हैं। वे चित्रों में किनारों, आकृतियों और बनावट जैसे पैटर्न का पता लगाने में बहुत अच्छे हैं। यही कारण है कि चेहरे की पहचान इतनी अच्छी तरह से काम करती है (चित्र 2.4)।



चित्र 2.4: कनवोल्यूशनल न्यूरल नेटवर्क (CNN) की संरचना

2.4.3 आवर्ती न्यूरल नेटवर्क (Recurrent Neural Network (RNN))

RNN में एक विशेष विशेषता होती है: स्मृति (memory)। उनके पास ऐसे लूप होते हैं जो जानकारी को बनाए रखने की अनुमति देते हैं, जिससे वे समय, पाठ या भाषण जैसे अनुक्रमिक डेटा के लिए एकदम सही हो जाते हैं। जब आप सिरी से बात करते हैं, तो एक RNN आपके शब्दों के संदर्भ को समझने में मदद करता है। (चित्र 2.4)

2.5 व्यावहारिक गतिविधि: पायथन में एक सरल न्यूरॉन (Hands-on Activity: A Simple Neuron in Python)

आइए कुछ सरल पायथन कोड के साथ एक न्यूरॉन को क्रियाशील देखें। यदि पायथन स्थापित है तो आप इसे अपने कंप्यूटर पर चला सकते हैं।

```
import numpy as np

# Our inputs: [hours studied, hours slept]
x = np.array([5, 8])

# Our weights: how important each input is
w = np.array([0.8, 0.4])

# Our bias
b = 0.2

# Calculate the weighted sum (like a score)
z = np.dot(x, w) + b
# z = (5*0.8 + 8*0.4) + 0.2 = (4.0 + 3.2) + 0.2 = 7.4

# Apply the ReLU activation function
output = max(0, z)

print("Neuron's Score (z):", z)
print("Neuron's Output after ReLU:", output)
```

Output:

Neuron's Score (z): 7.4

Neuron's Output after ReLU: 7.4

Since the score is positive, ReLU lets it pass through.

© PSSCIVE Draft Study Material Not be Published

सारांश (Summary)

- एक न्यूरल नेटवर्क विभिन्न परतों (layers): इनपुट (input), हिडन (hidden) और आउटपुट (output) से बना होता है।
- एक न्यूरॉन अपने इनपुट्स का वेटेड सम (weighted sum) निकालता है, उसमें बायस (bias) जोड़ता है और फिर एक एक्टिवेशन फ़ंक्शन (activation function) लागू करता है।
- Sigmoid मानों (values) को 0 और 1 के बीच सीमित (squish) कर देता है (हाँ/ना जैसे निर्णयों के लिए उपयोगी)।
- ReLU सकारात्मक (positive) मानों को आगे बढ़ाता है और नकारात्मक (negative) मानों को रोक देता है (बहुत कुशल/efficient)।
- Softmax स्कोर को कई श्रेणियों (categories) के लिए प्रायिकताओं (probabilities) में बदल देता है।
- FNN, CNN, और RNN अलग-अलग प्रकार के नेटवर्क हैं, जो विभिन्न कार्यों के लिए उपयोग किए जाते हैं।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. किसी न्यूरॉन का कौन सा भाग किसी इनपुट के महत्व को निर्धारित करता है? (अ) सक्रियण फ़ंक्शन (ब) बायस (स) भार (द) आउटपुट
2. कौन सा सक्रियण फ़ंक्शन 0 और 1 के बीच मान आउटपुट करता है? (अ) ReLU (ब) सिग्मॉइड (स) सॉफ्टमैक्स (द) रैखिक
3. किस प्रकार के न्यूरल नेटवर्क में "स्मृति" होती है और यह भाषण के लिए अच्छा होता है? (अ) CNN (ब) FNN (स) RNN (द) ANN
4. एक न्यूरॉन में बायस की क्या भूमिका होती है? (अ) इनपुट को गुणा करना (ब) सक्रियण फ़ंक्शन को स्थानांतरित करके लचीलापन प्रदान करना (स) आउटपुट संग्रहीत करना (द) अगली परत से जुड़ना
5. किसी छवि को कुत्ते, बिल्ली या पक्षी के रूप में वर्गीकृत करने के लिए, आउटपुट परत के लिए कौन सा सक्रियण फ़ंक्शन सबसे अच्छा है? (अ) ReLU (ब) सिग्मॉइड (स) सॉफ्टमैक्स (द) स्टेप

ब. रिक्त स्थान भरें

1. _____ वह लेयर है जो इनपुट और आउटपुट के बीच होती है, जहाँ सीखना (learning) होता है।
2. _____ फ़ंक्शन हिडन लेयर्स के लिए एक लोकप्रिय विकल्प है, क्योंकि यह तेज़ और सरल होता है।
3. _____ फ़ंक्शन का उपयोग आउटपुट लेयर में बाइनरी (दो-श्रेणी) वर्गीकरण के लिए किया जाता है।
4. _____ न्यूरल नेटवर्क्स विशेष रूप से इमेज रिकग्निशन के लिए बनाए जाते हैं।
5. एक न्यूरॉन के सूत्र में, वेटेड सम के बाद जो स्थिर मान जोड़ा जाता है, उसे _____ कहा जाता है।

स. सत्य या असत्य बताएँ

1. एक्टिवेशन फ़ंक्शन्स के बिना, एक न्यूरल नेटवर्क केवल सरल रैखिक (linear) पैटर्न ही सीख सकता है।
2. ReLU नकारात्मक (negative) संख्याएँ आउटपुट कर सकता है।
3. Softmax यह सुनिश्चित करता है कि सभी आउटपुट प्रायिकताओं (probabilities) का योग 1 हो।
4. फीडफॉरवर्ड न्यूरल नेटवर्क (Feedforward Neural Network) में लूप्स होते हैं, जो जानकारी को पीछे की ओर बहने देते हैं।
5. किसी इनपुट पर अधिक वेट (weight) होने का मतलब है कि वह अंतिम निर्णय के लिए कम महत्वपूर्ण है।

द. लघु उत्तरीय प्रश्न

1. सक्रियण फ़ंक्शन का उद्देश्य क्या है?
2. सिग्मॉइड और सॉफ्टमैक्स फ़ंक्शन्स के बीच अंतर स्पष्ट करें।
3. आप CNN के स्थान पर RNN का उपयोग कब करेंगे? एक उदाहरण दीजिए।
4. एक एकल कृत्रिम न्यूरॉन (artificial neuron) का मूल सूत्र लिखिए।
5. हिडन लेयर क्या कार्य करती है? इसे अपने शब्दों में बताइए।

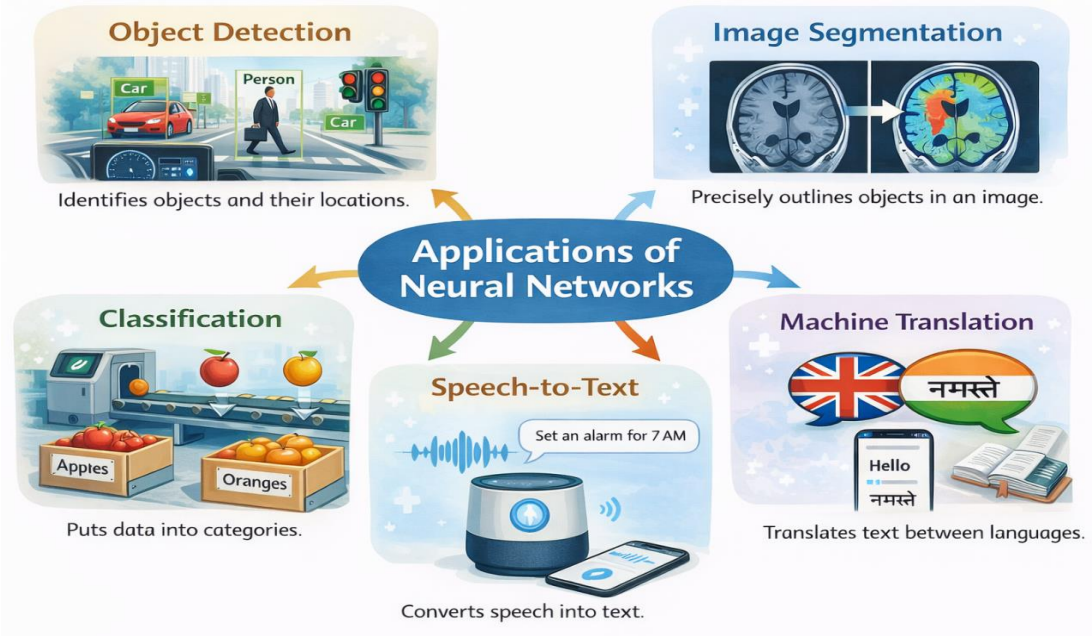
सत्र 3. क्रियाशील न्यूरल नेटवर्क: वास्तविक दुनिया के अनुप्रयोग

(Neural Networks in Action: Real-World Applications)

हमने देखा है कि न्यूरल नेटवर्क बहुत शक्तिशाली होते हैं। लेकिन वास्तविक दुनिया में वे आपके फोन से लेकर अस्पताल तक जटिल समस्याओं को हल कर रहे हैं। इस सत्र में, हम कुछ महत्वपूर्ण अनुप्रयोगों (applications) जैसे वस्तु पहचान (ऑब्जेक्ट डिटेक्शन), छवि विभाजन (इमेज सेगमेंटेशन) और भाषण-से-पाठ (स्पीच-टू-टेक्स्ट) का अध्ययन करेंगे।

3.1 अनुप्रयोगों का परिचय (Introduction to Applications)

क्योंकि न्यूरल नेटवर्क पैटर्न (patterns) खोजने में बहुत अच्छे होते हैं, इसलिए इन्हें कई अलग-अलग कार्यों में उपयोग किया जा सकता है। इनमें से कुछ सबसे सामान्य कार्य **दृष्टि (vision/देखना)**, **भाषा (language/शब्दों को समझना)** और **निर्णय लेने (decision-making)** से संबंधित होते हैं।



चित्र 3.1: वास्तविक दुनिया में न्यूरल नेटवर्क के प्रमुख अनुप्रयोग

3.2 वस्तु पहचान (Object Detection) (क्या और कहाँ?)

ऑब्जेक्ट डिटेक्शन एक ऐसी तकनीक है, जिसमें कंप्यूटर किसी चित्र (image) या वीडियो में वस्तुओं (objects) की पहचान और उनका स्थान निर्धारित करता है। यह दो सवालों के जवाब देता है: "यह क्या है?" और "यह कहाँ है?"

- क्या: यह वस्तु की पहचान करता है (जैसे कार, व्यक्ति, गेंद)।
- कहाँ: यह वस्तु के चारों ओर एक बॉक्स बनाता है (जिसे बाउंडिंग बॉक्स कहा जाता है)।
- न्यूरल नेटवर्क, विशेष रूप से CNNs, इसमें बहुत कुशल होते हैं। वे चित्र को छोटे-छोटे भागों में स्कैन करते हैं, विशेषताओं (features) की पहचान करते हैं और वस्तुओं का स्थान निर्धारित करते हैं।

वास्तविक जीवन उदाहरण:

- **सेल्फ-ड्राइविंग कारें** अन्य वाहनों, पैदल यात्रियों और ट्रैफिक संकेतों को पहचानने के लिए इसका उपयोग करती हैं।
- **सुरक्षा कैमरे** घुसपैठियों (intruders) का पता लगाने के लिए इसका उपयोग करते हैं।

3.3 छवि विभाजन (Image Segmentation) (पिक्सेल-स्तरीय समझ)

- छवि विभाजन (इमेज सेगमेंटेशन) वस्तु पहचान (ऑब्जेक्ट डिटेक्शन) का एक सुपर-पावर्ड संस्करण है। इसमें केवल वस्तु के चारों ओर बॉक्स बनाने के बजाय, नेटवर्क चित्र के **हर एक पिक्सेल (pixel)** को वर्गीकृत (classify) करता है। इससे वस्तु की बहुत सटीक रूपरेखा (outline) बनती है।

वास्तविक जीवन उदाहरण:

- **चिकित्सा इमेजिंग:** डॉक्टर एमआरआई स्कैन में ट्यूमर की सटीक रूपरेखा प्राप्त करने के लिए इसका उपयोग करते हैं।

- **स्वचालित कारें (Autonomous Cars):** यह एक कार को यह समझने में मदद करता है कि कौन से पिक्सेल सड़क हैं, कौन से फुटपाथ (sidewalk) हैं और कौन से आकाश या आसमान हैं।

3.4 वर्गीकरण (Classification) (यह क्या है?)

- वर्गीकरण इन कार्यों में सबसे सरल है। यह संपूर्ण छवि या डेटा के टुकड़े को देखता है और तय करता है कि यह किस श्रेणी से संबंधित है।

वास्तविक जीवन उदाहरण:

- **ईमेल स्पैम फ़िल्टरिंग**: नेटवर्क एक ईमेल पढ़ता है और उसे "स्पैम" या "स्पैम नहीं" के रूप में वर्गीकृत करता है।
- **उत्पाद छँटाई**: कारखाने में कन्वेयर बेल्ट पर एक प्रणाली फल को "सेब" या "संतरा" के रूप में वर्गीकृत करती है।

3.5 भाषण-से-पाठ (Speech-to-Text) (आवाज़ से शब्दों में)

जब आप अपने फोन पर कोई संदेश बोलकर लिखवाते हैं या किसी स्मार्ट स्पीकर से सवाल पूछते हैं, तब यही प्रक्रिया होती है। आपकी आवाज़ की ध्वनि तरंगों (sound waves) को संख्याओं (numbers) में बदला जाता है। इसके बाद एक RNN या विशेष प्रकार का नेटवर्क जिसे LSTM कहा जाता है, इन संख्याओं के क्रम (sequence) को प्रोसेस करता है और उसे टेक्स्ट (text) में बदल देता है।

वास्तविक जीवन उदाहरण:

- **आभासी सहायक (वर्चुअल असिस्टेंट्स):** सिरी, गूगल असिस्टेंट और एलेक्सा।
- **लाइव कैप्शन:** यूट्यूब और वीडियो कॉन्फ्रेंसिंग ऐप्स वास्तविक समय (रियल-टाइम) में कैप्शन बना सकते हैं।

3.6 मशीन अनुवाद (Machine Translation) (एक भाषा से दूसरी भाषा में)

क्या आपने कभी गूगल ट्रांसलेट (Google Translate) का उपयोग करके अंग्रेज़ी को हिंदी में बदला है? यही मशीन अनुवाद (Machine Translation) है। इसमें एक न्यूरल नेटवर्क एक भाषा में लिखे गए वाक्य को पढ़ता है, उसका अर्थ समझता है और दूसरी भाषा में उसी अर्थ का वाक्य बना देता है। यह कार्य अक्सर एक विशेष प्रकार के नेटवर्क, जिसे ट्रांसफॉर्मर (Transformer) कहा जाता है, द्वारा किया जाता है।

वास्तविक जीवन उदाहरण:

- **गूगल ट्रांसलेट:** वेब पेज या बातचीत का तुरंत अनुवाद करना।
- **भाषा सीखने वाले ऐप्स:** विदेशी वाक्यों को समझने में मदद करना।
- **वस्तु पहचान (Object Detection)** वस्तुओं को पहचानता है और उनके चारों ओर बॉक्स बनाता है।
- **छवि विभाजन (Image Segmentation)** हर एक पिक्सेल को लेबल करके सटीक रूपरेखा देता है।
- **वर्गीकरण (Classification)** डेटा को एक श्रेणी (category) में रखता है।
- **भाषण-से-पाठ (Speech-to-Text)** बोले गए शब्दों को लिखित पाठ (टेक्स्ट) में बदलता है।
- **मशीन अनुवाद (Machine Translation)** पाठ (टेक्स्ट) को एक भाषा से दूसरी भाषा में बदलता है।

सारांश (Summary)

इस सत्र में दैनिक जीवन में न्यूरल नेटवर्क जैसे चेहरा पहचान (फेस रिकग्निशन), आवाज सहायक (वॉइस असिस्टेंट्स) और अनुशंसा प्रणाली (रिकमेंडेशन सिस्टम्स) का उपयोग किया गया है। इस तरह यह सत्र सिद्धांत (theory) को व्यवहार (practice) से जोड़ता है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. किस अनुप्रयोग का उपयोग किसी चिकित्सा छवि में ट्यूमर के चारों ओर एक सटीक रूपरेखा बनाने के लिए किया जाएगा? (अ) वर्गीकरण (ब) वस्तु पहचान (स) छवि विभाजन (द) भाषण-से-पाठ
2. कोई ईमेल "स्पैम" है या "स्पैम नहीं" यह तय करना एक उदाहरण है। (अ) मशीन अनुवाद (ब) वस्तु पहचान (स) विभाजन (द) वर्गीकरण
3. भाषण-से-पाठ के लिए न्यूरल नेटवर्क सबसे उपयुक्त है? (अ) CNN (ब) FNN (स) RNN (द) इनमें से कोई नहीं
4. एक वाक्य को फ्रेंच से अंग्रेजी में बदलना कहलाता है। (अ) भाषण-से-पाठ (ब) मशीन अनुवाद (स) वस्तु पहचान (द) छवि प्रसंस्करण
5. वस्तु पहचान किन दो प्रश्नों का उत्तर देती है? (अ) "क्यों?" और "कब?" (ब) "क्या?" और "कहाँ?" (स) "कौन?" और "कैसे?" (द) "क्या?" और "क्यों?"

ब. रिक्त स्थान भरें

1. किसी छवि में किसी वस्तु के चारों ओर एक बॉक्स बनाने की तकनीक को _____ कहा जाता है।
2. _____ किसी छवि में प्रत्येक पिक्सेल को लेबल करने की प्रक्रिया है।
3. गूगल ट्रांसलेट _____ का एक अनुप्रयोग है।
4. सिरी जैसे आवाज सहायक कमांड समझने के लिए _____ का उपयोग करते हैं।
5. किसी ईमेल को "स्पैम" या "इनबॉक्स" फ़ोल्डर में रखना एक _____ कार्य है।

स. सत्य या असत्य बताएँ

1. वस्तु पहचान आपको बता सकती है कि छवि में एक कार है, लेकिन यह नहीं कि वह कहाँ है।
2. छवि विभाजन वस्तु पहचान से अधिक विस्तृत है।
3. भाषण-से-पाठ प्रणालियाँ माइक्रोफोन और न्यूरल नेटवर्क का उपयोग करती हैं।
4. मशीन अनुवाद केवल अंग्रेजी के लिए काम करता है।
5. वर्गीकरण कार्यों में हमेशा दो से अधिक श्रेणियाँ होती हैं।

द. लघु उत्तरीय प्रश्न

1. छवि वर्गीकरण और वस्तु पहचान के बीच मुख्य अंतर स्पष्ट कीजिए।
2. स्व-ड्राइविंग कारों में छवि विभाजन कैसे सहायक है?

3. दैनिक जीवन में आप मशीन अनुवाद का उपयोग कैसे कर सकते हैं, इसका एक उदाहरण दीजिए।
4. भाषण-से-पाठ प्रणाली में पहला चरण क्या है?
5. वस्तु पहचान में "कहाँ" की जानकारी क्यों उपयोगी है?

सत्र 4. पायथन लाइब्रेरी के साथ न्यूरल नेटवर्क का निर्माण

(Building Neural Networks with Python Libraries)

कल्पना कीजिए कि आप एक घर बनाना चाहते हैं। आप शुरुआत से अपनी ईंटें और सीमेंट बना सकते हैं या आप इसे तेज़ी से बनाने के लिए स्टोर से तैयार सामग्री का उपयोग कर सकते हैं। पायथन लाइब्रेरी न्यूरल नेटवर्क बनाने के लिए उस स्टोर की तरह हैं। वे हमें तैयार उपकरण देते हैं ताकि हम डिजाइन पर ध्यान केंद्रित कर सकें।

इस सत्र में, हम AI के लिए आवश्यक पायथन लाइब्रेरी के बारे में सीखेंगे और हस्तलिखित अंकों को पहचानने के लिए अपना पहला न्यूरल नेटवर्क बनाएंगे।

4.1 पायथन लाइब्रेरी का परिचय (Introduction to Python Libraries)

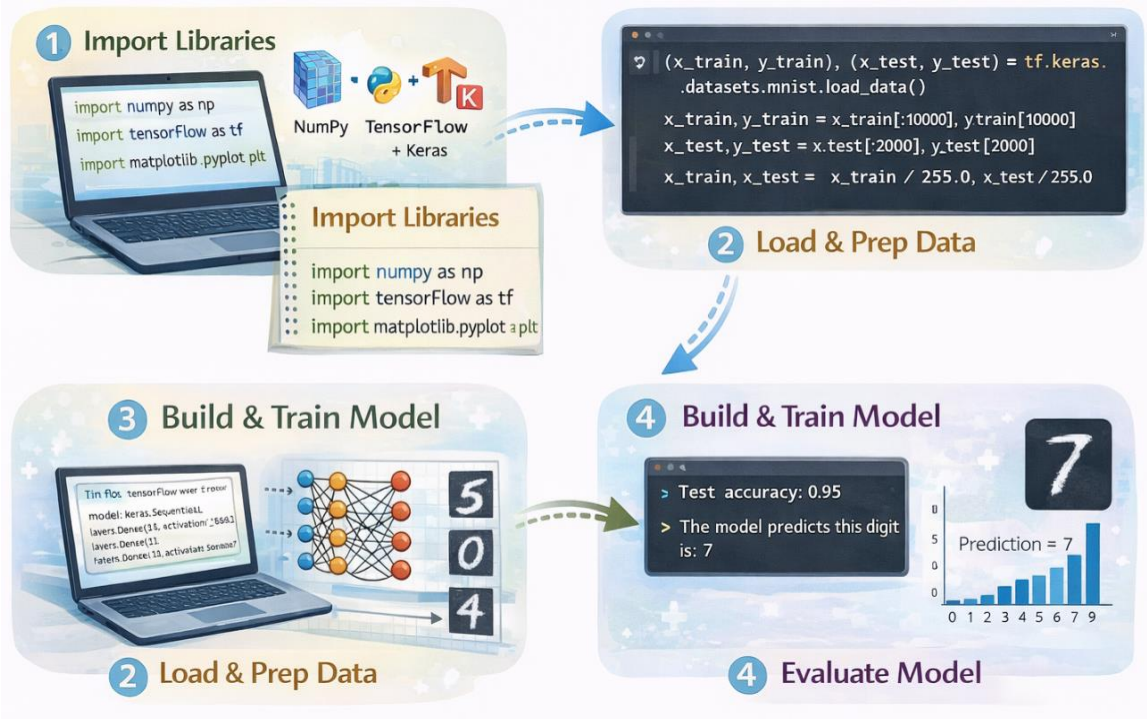
AI के लिए पायथन एक लोकप्रिय भाषा है क्योंकि इसमें कई शक्तिशाली और उपयोग में आसान लाइब्रेरी हैं। लाइब्रेरी को पूर्व-लिखित कोड के संग्रह के रूप में सोचें जो आपको शून्य से शुरू किए बिना विशिष्ट कार्य करने में मदद करता है। हम जिन मुख्य लाइब्रेरी का उपयोग करेंगे, वे हैं:

NumPy: गणितीय संक्रियाओं के लिए मौलिक लाइब्रेरी। यह एरे और मैट्रिसेस के साथ काम करने में हमारी सहायता करता है, जो न्यूरल नेटवर्क में डेटा का प्रतिनिधित्व करने का तरीका है।

TensorFlow: Google द्वारा विकसित बड़े पैमाने पर न्यूरल नेटवर्क बनाने और प्रशिक्षित करने के लिए एक शक्तिशाली लाइब्रेरी।

Keras: एक उपयोगकर्ता-अनुकूल API (एप्लीकेशन प्रोग्रामिंग इंटरफेस) जो TensorFlow के शीर्ष पर चलता है। यह न्यूरल नेटवर्क बनाने को लेगो ब्लॉक्स को जोड़ने जितना सरल बनाता है। हम Keras का उपयोग करेंगे।

Matplotlib: ग्राफ और चार्ट बनाने के लिए एक लाइब्रेरी, जो हमारे डेटा और परिणामों को देखने में हमारी सहायता करती है।



चित्र 4.1: पायथन लाइब्रेरी का उपयोग करके एक न्यूरल नेटवर्क बनाने की कार्यप्रवाह

4.2 अपना पायथन वातावरण स्थापित करना (Setting Up Your Python Environment)

निर्माण शुरू करने के लिए, आपको इन लाइब्रेरी को इंस्टाल करना होगा। यदि आपके पास पायथन इंस्टाल है, तो आप अपना कमांड प्रॉम्प्ट या टर्मिनल खोल सकते हैं और इन कमांड को टाइप कर सकते हैं:

```
Bash
pip install numpy tensorflow matplotlib
```

बस इतना ही! यह वह सब कुछ डाउनलोड और इंस्टॉल करेगा जो आपको चाहिए।

4.3 निर्माण (Build) के दो तरीके: निम्न-स्तरीय बनाम उच्च-स्तरीय

4.3.1 निम्न-स्तरीय (NumPy के साथ): गणित स्वयं करना

निम्न-स्तरीय दृष्टिकोण में, हम एक न्यूरॉन के लिए गणित को चरण-दर-चरण लिखते हैं।

उदाहरण Python कोड के साथ

```
import numpy as np
# Inputs (e.g., pixels of an image)
inputs = np.array([0.5, 0.8, 0.2])
# Weights (importance of each pixel)
weights = np.array([0.4, 0.1, -0.7])
# Bias
bias = 0.3
```

```
# The neuron's calculation
z = np.dot(inputs, weights) + bias
output = max(0, z) # ReLU activation
print("Neuron output:", output)
```

4.3.2 उच्च-स्तरीय (Keras के साथ): तैयार ब्लॉक्स का उपयोग करना

Keras के साथ, हम एक पंक्ति के कोड में न्यूरॉन्स की एक पूरी परत बना सकते हैं। हमें प्रत्येक न्यूरॉन के लिए गणित लिखने की आवश्यकता नहीं है।

उदाहरण Python कोड के साथ

```
from tensorflow.keras.layers import Dense
# This one line creates a layer with 10 neurons!
# It automatically handles all the weights, biases, and math.
layer = Dense(10, activation='relu')
```

Not be Publis

4.4 एक नेटवर्क बनाने के लिए मुख्य घटक (Key Components for Building a Network)

4.4.1 लेयर्स (Layers)

डेंस परत (Dense Layer): सबसे सामान्य परत (लेयर)। डेंस परत में प्रत्येक न्यूरॉन पिछली परत के प्रत्येक न्यूरॉन से जुड़ा होता है।

फ्लैटन परत (Flatten Layer): यह एक विशेष परत है जिसका उपयोग छवियों के लिए किया जाता है। यह पिक्सेल (जैसे 28x28 छवि) के 2D ग्रिड को एक एकल, लंबी 1D सूची (784 संख्याओं की) में समतल करता है ताकि इसे डेंस परत में डाला जा सके।

- **हानि फ़ंक्शन (Loss Functions) (गलतियों को मापना):** हानि फ़ंक्शन नेटवर्क को बताता है कि उसकी भविष्यवाणी कितनी गलत थी। प्रशिक्षण का लक्ष्य इस संख्या को जितना संभव हो उतना छोटा बनाना है।
- **sparse_categorical_crossentropy:** हम इसका उपयोग वर्गीकरण समस्याओं के लिए करते हैं जहाँ उत्तर केवल एक संख्या है (जैसे अंक '7')।
- **ऑप्टिमाइज़र (Optimizers) (सीखने की रणनीति):** ऑप्टिमाइज़र वह एल्गोरिथ्म है जो हानि को कम करने के लिए भारों और बायस को अद्यतन करता है।
- **adam:** एक बहुत लोकप्रिय और प्रभावी ऑप्टिमाइज़र। यह अधिकांश समस्याओं के लिए एक बेहतरीन शुरुआती बिंदु है।

4.5 आपका पहला न्यूरल नेटवर्क: एमएनआईएसटी अंक वर्गीकरणकर्ता (Your First Neural Network: The MNIST Digit Classifier)

आइए एक वास्तविक प्रोजेक्ट बनाएँ! हम प्रसिद्ध MNIST डेटासेट का उपयोग करके हस्तलिखित अंकों (0-9) को पहचानने के लिए एक न्यूरल नेटवर्क को प्रशिक्षित करेंगे।

चरण 1. लाइब्रेरी इम्पोर्ट करें (Import Libraries)

उदाहरण Python कोड के साथ

```
from tensorflow import keras
from tensorflow.keras import layers
import numpy as np
import matplotlib.pyplot as plt
```

चरण 2. डेटा लोड करें और तैयार करें (Load and Prepare the Data)

हम MNIST डेटासेट लोड करते हैं और कुछ सरल प्री-प्रोसेसिंग करते हैं।

उदाहरण Python कोड के साथ

```
# Load the data
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
# To make it faster on slower computers, let's use a smaller subset
x_train = x_train[:10000]
y_train = y_train[:10000]
x_test = x_test[:2000]
y_test = y_test[:2000]
# Normalize the pixel values from 0-255 to 0-1. This helps the network learn better.
x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
# Reshape the data: flatten the 28x28 images into 1D arrays of 784 numbers.
x_train = x_train.reshape(-1, 28*28)
x_test = x_test.reshape(-1, 28*28)
```

चरण 3. मॉडल बनाएँ (Build the Model)

हम लेयर्स (परतों) को ढेर करने के लिए Keras के **अनुक्रमिक (Sequential)** मॉडल का उपयोग करते हैं।

उदाहरण Python कोड के साथ

```
model = keras.Sequential([
    layers.Dense(64, activation='relu', input_shape=(784,)), # Hidden layer with 64 neurons
    layers.Dense(10, activation='softmax')                   # Output layer with 10 neurons (one
for each digit 0-9)
```

SCIVE Draft Study Material Not be Pu

```
])
```

चरण 4. मॉडल संकलित करें (Compile the Model)

हम मॉडल को बताते हैं कि कैसे सीखना है।

उदाहरण Python कोड के साथ

```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

चरण 5. मॉडल को प्रशिक्षित करें (Train the Model)

उदाहरण Python कोड के साथ

```
history = model.fit(x_train, y_train, epochs=5, batch_size=128)
```

चरण 6. मॉडल का मूल्यांकन करें (Evaluate the Model)

उदाहरण Python कोड के साथ

```
loss, acc = model.evaluate(x_test, y_test)
print(f'Test accuracy: {acc:.2f}')
```

आपको लगभग 0.95-0.97 की सटीकता देखनी चाहिए, जिसका अर्थ है कि यह नए, अनदेखे अंकों पर लगभग 95-97% सही है!

चरण 7. एक भविष्यवाणी करें (Make a Prediction)

आइए देखें कि मॉडल हमारे परीक्षण सेट में पहली छवि के लिए क्या भविष्यवाणी करता है।

उदाहरण Python कोड के साथ

```
import numpy as np
img = x_test[0].reshape(1, 784)
prediction = model.predict(img)
predicted_digit = np.argmax(prediction)
print(f"The model predicts this digit is: {predicted_digit}")
```

सारांश (Summary)

- NumPy गणित के लिए है, TensorFlow/Keras नेटवर्क बनाने के लिए है और Matplotlib प्लॉटिंग के लिए है।
- Keras न्यूरल नेटवर्क बनाने को सरल और तेज़ बनाता है।

- MNIST डेटासेट छवि पहचान के लिए एक क्लासिक "हैलो वर्ल्ड" है।
- कार्यप्रवाह है: डेटा लोड करें → मॉडल बनाएँ → संकलित करें → प्रशिक्षित करें → मूल्यांकन करें → भविष्यवाणी करें।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. कौन सी पायथन लाइब्रेरी सरल, लेगो-जैसे इंटरफ़ेस के साथ न्यूरल नेटवर्क बनाने के लिए सबसे अच्छी है? (अ) NumPy (ब) Matplotlib (स) Keras (द) Pandas
2. फ्लैटन (Flatten) परत (लेयर) क्या करती है? (अ) यह डेटा को सामान्यीकृत करती है। (ब) यह 2D छवि को 1D एरे में बदल देती है। (स) यह नेटवर्क में अधिक न्यूरॉन जोड़ती है। (द) यह मॉडल को संकलित करती है।
3. MNIST उदाहरण में, आउटपुट परत के सक्रियण फ़ंक्शन 'सॉफ्टमैक्स' का क्या कार्य है? (अ) यह ऋणात्मक मानों को रोकता है। (ब) यह आउटपुट को संभाव्यताओं में बदल देता है जिनका योग 1 होता है। (स) यह हानि की गणना करता है। (द) यह छवि को समतल करता है।
4. हमारे MNIST मॉडल में हानि फ़ंक्शन था? (अ) adam (ब) accuracy (स) relu (द) sparse_categorical_crossentropy
5. model.fit() फ़ंक्शन क्या करता है? (अ) यह परीक्षण डेटा पर मॉडल का मूल्यांकन करता है। (ब) यह मॉडल की वास्तुकला का निर्माण करता है। (स) यह डेटा पर मॉडल को प्रशिक्षित करता है। (द) यह एक भविष्यवाणी करता है।

ब. रिक्त स्थान भरें

1. पायथन में गणितीय संक्रियाओं के लिए लाइब्रेरी _____ है।
2. Keras में, आप एक _____ मॉडल के अंदर परतों को ढेर करते हैं।
3. पिक्सेल मानों को 0-255 से 0-1 तक स्केल करने की प्रक्रिया को _____ कहा जाता है।
4. _____ फ़ंक्शन हमें बताता है कि प्रशिक्षण के दौरान मॉडल कितना अच्छा प्रदर्शन कर रहा है।
5. _____ वह एल्गोरिथ्म है जो हानि को कम करने के लिए मॉडल के भारों को अद्यतन (अपडेट) करता है।

स. सत्य या असत्य बताएँ

1. Keras को आपको प्रत्येक न्यूरॉन के लिए गणितीय सूत्र लिखने की आवश्यकता होती है।
2. MNIST डेटासेट में कपड़ों की छवियाँ होती हैं।
3. एक ऑप्टिमाइज़र का काम हानि फ़ंक्शन को न्यूनतम करना है।
4. डेटा को सामान्यीकृत करने से न्यूरल नेटवर्क को तेज़ और बेहतर प्रशिक्षित करने में मदद मिलती है।

5. प्रशिक्षण के बाद, evaluate फ़ंक्शन मॉडल का परीक्षण उसी डेटा पर करता है जिस पर उसे प्रशिक्षित किया गया था।

द. लघु उत्तरीय प्रश्न

1. मशीन लर्निंग में उपयोग की जाने वाली दो पायथन लाइब्रेरी के नाम बताइए और संक्षेप में बताइए कि प्रत्येक क्या करती है।
2. Keras में न्यूरल नेटवर्क बनाने और उपयोग करने के छह मुख्य चरणों की सूची बनाएँ।
3. हम छवि डेटा को 0 और 1 के बीच के मानों में क्यों सामान्यीकृत करते हैं?
4. प्रशिक्षण डेटा और परीक्षण डेटा के बीच क्या अंतर अपने शब्दों में दीजिए ?
5. 0.96 की सटीकता का क्या अर्थ है?

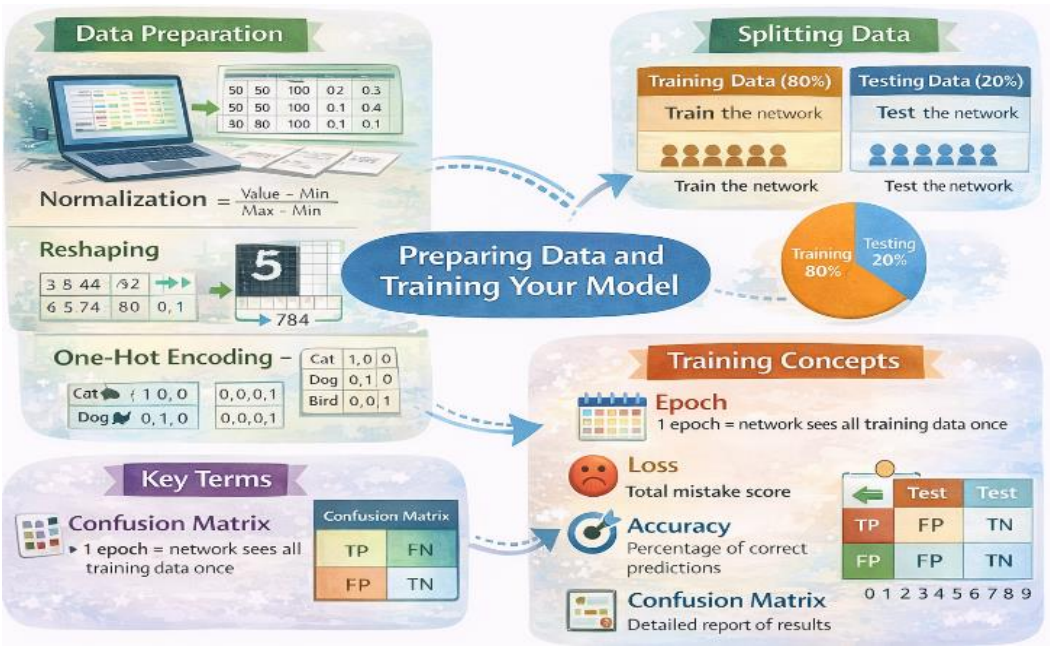
सत्र 5. डेटा तैयार करना और अपने मॉडल को प्रशिक्षित करना

(Preparing Data and Training Your Model)

आप किसी विद्यार्थी को पहले पढ़ाए बिना परीक्षा नहीं लेंगे। इसी तरह, आप केवल कच्चा डेटा एक न्यूरल नेटवर्क को नहीं दे सकते। आपको इसे तैयार करना होगा। इस सत्र में, हम सीखेंगे कि डेटा को प्रशिक्षण के लिए कैसे तैयार किया जाए और यह कैसे समझा जाए कि क्या मॉडल ने अच्छी तरह से सीखा है।

5.1 डेटा तैयारी: डेटा को तैयार करना (Data Preparation: Getting the Data Ready)

कच्चा डेटा अक्सर गन्दा होता है। इसमें अलग-अलग पैमाने हो सकते हैं या ऐसे प्रारूप में हो सकता है जिसे कंप्यूटर नहीं समझता। हमें इसे साफ करने की आवश्यकता है।



चित्र 5.1: न्यूरल नेटवर्क में डेटा तैयारी और मॉडल प्रशिक्षण प्रक्रिया

5.1.1 सामान्यीकरण (Normalization) (सब कुछ एक ही पैमाने पर रखना)

दो विषयों से अंकों की तुलना करने की कल्पना करें: एक का अधिकतम 50 अंक है और दूसरे का अधिकतम 100 अंक है। उनकी निष्पक्ष तुलना करने के लिए, आप उन्हें प्रतिशत में बदल देंगे। सामान्यीकरण डेटा के लिए भी ऐसा ही करता है। यह सभी मानों को एक मानक सीमा में स्केल करता है, आमतौर पर 0 से 1 तक।

यह महत्वपूर्ण है क्योंकि यदि एक इनपुट बहुत बड़ा है (जैसे लाखों में आय) और दूसरा बहुत छोटा है (जैसे आयु), तो बड़ी संख्या सीखने की प्रक्रिया पर हावी हो सकती है। सूत्र सरल है:

सामान्यीकृत = (मूल – न्यूनतम) / (अधिकतम – न्यूनतम)

उदाहरण Python कोड के साथ

```
import numpy as np
# Student scores out of different totals
scores = np.array([45, 80, 90, 60, 30]) # Max score here is 90, min is 30
normalized_scores = (scores - 30) / (90 - 30)
print(normalized_scores) # Output: [0.25 0.83 1.0  0.5  0.  ]
```

सभी अंक अब 0 और 1 के बीच हैं।

5.1.2 पुनः आकार देना (Reshaping) (आकार बदलना)

तंत्रिका नेटवर्क डेटा को एक विशिष्ट प्रारूप में अपेक्षा करते हैं। हमारे अंक वर्गीकरणकर्ता के लिए, हमारे पास 28x28 छवियाँ थीं। लेकिन पहली डेंस परत संख्याओं की एक एकल, सपाट सूची की अपेक्षा करती है। हम आकार को (28, 28) से (784,) में बदलने के लिए reshape का उपयोग करते हैं।

5.1.3 वन-हॉट एन्कोडिंग (One-Hot Encoding) (श्रेणियों के लिए)

यदि आप जानवरों को "बिल्ली", "कुत्ता" या "पक्षी" के रूप में वर्गीकृत करना चाहते हैं, तो आप नेटवर्क को केवल शब्द नहीं दे सकते। आपको उन्हें संख्याओं में बदलने की आवश्यकता है। वन-हॉट एन्कोडिंग प्रत्येक श्रेणी के लिए एक नया स्तंभ बनाकर और उसे 1 या 0 से चिह्नित करके ऐसा करती है।

```
"Cat" ⟶ [1, 0, 0]
"Dog" ⟶ [0, 1, 0]
"Bird" ⟶ [0, 0, 1]
```

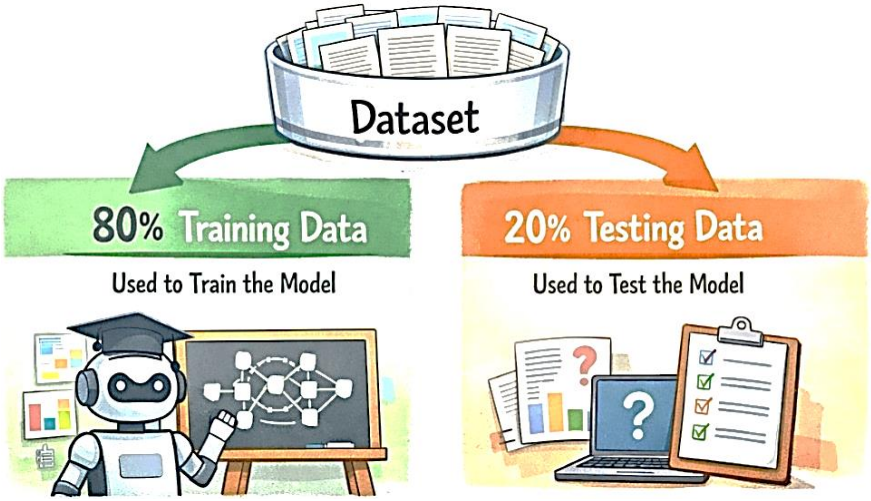
हमारे MNIST उदाहरण में, हमें ऐसा करने की आवश्यकता नहीं थी क्योंकि हमने **sparse_categorical_crossentropy** हानि का उपयोग किया था, जो हमारे लिए इसे संभालती है।

5.2 डेटा विभाजन: प्रशिक्षण बनाम परीक्षण (Splitting Data: Training vs. Testing)

आप किसी विद्यार्थी को पढ़ाने और परीक्षण करने के लिए एक ही प्रश्न का उपयोग नहीं करते है। न्यूरल नेटवर्क के लिए भी ऐसा ही है। हमें अपने डेटा को दो अलग-अलग सेटों में विभाजित करना चाहिए:

प्रशिक्षण डेटा (80%): नेटवर्क सिखाने और उसके भारों को समायोजित करने के लिए उपयोग किया जाता है।

परीक्षण डेटा (20%): केवल अंत में यह जांचने के लिए उपयोग किया जाता है कि नेटवर्क नए, अनदेखे डेटा पर कितना अच्छा प्रदर्शन करता है। यह हमें बताता है कि क्या मॉडल ने वास्तव में सीखा है या सिर्फ याद किया है।



चित्र 5.2: एक डेटासेट को प्रशिक्षण और परीक्षण अनुभागों में विभाजित करना दिखाने वाली छवि

5.3 मुख्य प्रशिक्षण अवधारणाएँ (Key Training Concepts)

5.3.1 युग (Epoch)

एक युग (एपोच) का मतलब है कि नेटवर्क ने सभी प्रशिक्षण डेटा को एक बार देख लिया है और उसने अपने भारों को समायोजित कर लिया है। प्रशिक्षण आमतौर पर कई युगों (जैसे, 5, 10, 50) के लिए चलता है। जैसे-जैसे युगों की संख्या बढ़ती है, नेटवर्क अपने कार्य में बेहतर और बेहतर होता जाता है।

5.3.2 हानि (Loss)

हानि एक संख्या है जो मापती है कि मॉडल की भविष्यवाणियाँ कितनी गलत हैं। प्रशिक्षण का लक्ष्य इस संख्या को जितना संभव हो उतना छोटा बनाना है। आप इसे "कुल गलती स्कोर" के रूप में सोच सकते हैं।

5.3.3 सटीकता (Accuracy)

सटीकता सही भविष्यवाणियों का प्रतिशत है। यदि मॉडल 100 छवियों में से 90 का सही अनुमान लगाता है, तो उसकी सटीकता 90% है।

5.3.4 भ्रम मैट्रिक्स (Confusion Matrix) (एक विस्तृत रिपोर्ट)

सटीकता कभी-कभी भ्रामक हो सकती है। एक दुर्लभ बीमारी के परीक्षण की कल्पना करें जो 1% लोगों को प्रभावित करती है। एक मॉडल जो हमेशा "कोई बीमारी नहीं" की भविष्यवाणी करता है, वह 99% सटीक होगा, लेकिन यह बेकार है! एक भ्रम मैट्रिक्स अधिक विस्तृत विवरण देता है।

यह एक सरल 2x2 तालिका है जो मॉडल की भविष्यवाणियों की तुलना वास्तविक उत्तरों से करती है।

तालिका 5.3.1: भ्रम मैट्रिक्स संरचना (Confusion Matrix Structure)

	भविष्यवाणी की गई सकारात्मक	भविष्यवाणी की गई नकारात्मक
वास्तविक सकारात्मक	सच्चा सकारात्मक (True Positive - TP)	गलत नकारात्मक (False Negative - FN)
वास्तविक नकारात्मक	गलत सकारात्मक (False Positive - FP)	सच्चा नकारात्मक (True Negative - TN)

सच्चा सकारात्मक (True Positive (TP)): मॉडल ने "हाँ" कहा, और यह सही था।

सच्चा नकारात्मक (True Negative (TN)): मॉडल ने "नहीं" कहा, और यह सही था।

गलत सकारात्मक (False Positive (FP)): मॉडल ने "हाँ" कहा, लेकिन यह गलत था (एक झूठा अलार्म)।

गलत नकारात्मक (False Negative (FN)): मॉडल ने "नहीं" कहा, लेकिन यह गलत था (एक चूक)।

5.4 सामान्य सक्रियण फ़ंक्शन (Common Activation Functions)

इसे पहले भी हम पढ़ चुके हैं, लेकिन आइए प्रशिक्षण के संदर्भ में उन पर नज़र डालें।

- **ReLU:** छिपी हुई परतों के लिए कार्यशील घोड़ा। यह तेज़ है और केवल सकारात्मक संकेतों को पारित करके नेटवर्क को जटिल पैटर्न सीखने में मदद करता है।
- **सिग्मॉइड (Sigmoid):** द्विआधारी वर्गीकरण में आउटपुट परत के लिए निर्णायक। यह 0 और 1 के बीच एक स्पष्ट संभाव्यता देता है।
- **Tanh:** सिग्मॉइड के समान, लेकिन इसका आउटपुट -1 से 1 की सीमा में होता है। यह तब उपयोगी होता है जब डेटा में मजबूत नकारात्मक मान हों।

5.5 इसे एक साथ रखना: एक सरल प्रशिक्षण उदाहरण (Putting It All Together: A Simple Training Example)

आइए कल्पना करें कि हम यह भविष्यवाणी करना चाहते हैं कि कोई विद्यार्थी पढ़ाई किए गए घंटों के आधार पर पास (1) होगा या फेल (0)। हमारे पास 10 विद्यार्थियों का डेटा है।

उदाहरण Python कोड के साथ

```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.model_selection import train_test_split

# Data: [Hours Studied] -> [Pass/Fail]
X = np.array([[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]])
y = np.array([0, 0, 0, 1, 1, 1, 1, 1, 1, 1]) # 0 = Fail, 1 = Pass
```

© PSSCIVE

```
# 1. Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 2. Build the model
model = Sequential([
    Dense(4, activation='relu', input_dim=1),
    Dense(1, activation='sigmoid')
])

# 3. Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# 4. Train the model
history = model.fit(X_train, y_train, epochs=50, verbose=0)

# 5. Evaluate the model
loss, acc = model.evaluate(X_test, y_test, verbose=0)
print(f"Test Accuracy: {acc:.2f}")
```

यह सरल उदाहरण कच्चे डेटा से प्रशिक्षित और परीक्षण किए गए मॉडल तक की पूरी प्रक्रिया दिखाता है।

Study Material Not be Published

सारांश (Summary)

- प्रशिक्षण से पहले डेटा को सामान्यीकृत (स्केल) और पुनः आकार दिया जाना चाहिए।
- डेटा को प्रशिक्षण (सीखने के लिए) और परीक्षण (मूल्यांकन के लिए) सेट में विभाजित किया जाता है।
- एक युग प्रशिक्षण डेटा का एक पूरा पास होता है।
- हानि गलतियों को मापती है, सटीकता शुद्धता को मापती है।
- एक भ्रम मैट्रिक्स सही और गलत भविष्यवाणियों का विस्तृत विवरण देता है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. हम डेटा को सामान्यीकृत क्यों करते हैं? (अ) डेटासेट को छोटा बनाने हेतु (ब) सभी डेटा को एक समान पैमाने पर रखने हेतु ताकि कोई एक विशेषता हावी न हो (स) डेटा को अधिक रंगीन बनाने हेतु (द) डेटा को प्रशिक्षण और परीक्षण सेट में विभाजित करने हेतु
2. प्रशिक्षण में एक युग (epoch) का अर्थ है? (अ) मॉडल ने एक एकल छवि देखी है (ब) मॉडल ने एक बार अपने भारों को समायोजित किया है (स) मॉडल ने सभी प्रशिक्षण डेटा को एक बार देख लिया है (द) मॉडल का परीक्षण किया गया है
3. एक भ्रम मैट्रिक्स (confusion matrix) में, एक "गलत सकारात्मक (False Positive)" का अर्थ है? (अ) मॉडल ने सही ढंग से "हाँ" की भविष्यवाणी की। (ब) मॉडल ने "हाँ" की भविष्यवाणी की, लेकिन वास्तव में यह "नहीं" था। (स) मॉडल ने "नहीं" की भविष्यवाणी की, लेकिन वास्तव में यह "हाँ" था। (द) मॉडल ने सही ढंग से "नहीं" की भविष्यवाणी की।
4. द्विआधारी (हाँ/नहीं) वर्गीकरणकर्ता की आउटपुट परत के लिए कौन सा सक्रियण फ़ंक्शन सबसे अच्छा है? (अ) ReLU (ब) Softmax (स) Sigmoid (द) Tanh
5. हमें एक अलग परीक्षण सेट की आवश्यकता क्यों है? (अ) मॉडल को और प्रशिक्षित करने हेतु (ब) अनदेखे डेटा पर मॉडल के प्रदर्शन की जांच करने हेतु (स) डेटा को सामान्यीकृत करने हेतु (द) मॉडल वास्तुकला बनाने हेतु।

ब. रिक्त स्थान भरें

1. मानों को 0 से 1 की सीमा में स्केल करने की प्रक्रिया को _____ कहा जाता है।
2. हम अपने डेटा को एक _____ सेट और एक _____ सेट में विभाजित करते हैं।
3. _____ एक तालिका है जो मॉडल की सही और गलत भविष्यवाणियों का विस्तृत विवरण दिखाती है।
4. कम _____ मान का अर्थ है कि मॉडल कम गलतियाँ कर रहा है।
5. _____ फ़ंक्शन छिपी हुई परतों के लिए एक लोकप्रिय विकल्प है क्योंकि यह सरल और तेज़ है।

स. सत्य या असत्य बताएँ

1. अपने मॉडल का परीक्षण उसी डेटा पर करना ठीक है जिसका उपयोग आपने प्रशिक्षण के लिए किया था।
2. सटीकता हमेशा किसी मॉडल के प्रदर्शन का सही माप होती है।
3. सामान्यीकरण एक न्यूरल नेटवर्क को तेज़ी से प्रशिक्षित करने में मदद कर सकता है।
4. सर्वोत्तम मॉडल प्राप्त करने के लिए युगों की संख्या हमेशा अधिकतम होनी चाहिए।
5. एक भ्रम मैट्रिक्स हमें मॉडल द्वारा की जा रही त्रुटियों के प्रकार को समझने में मदद करता है।

द. लघु उत्तरीय प्रश्न

1. डेटा को प्रशिक्षण और परीक्षण सेटों में विभाजित करना क्यों महत्वपूर्ण है?
2. हानि (loss) और सटीकता (accuracy) के बीच क्या अंतर है?
3. एक सरल उदाहरण के साथ भ्रम मैट्रिक्स (confusion matrix) के चार शब्दों (TP, TN, FP, FN) की व्याख्या करें।

4. यदि हम किसी न्यूरल नेटवर्क को उसके पिक्सेल मानों को सामान्यीकृत किए बिना एक छवि खिलाते हैं, तो क्या हो सकता है?
5. यदि किसी मॉडल में उच्च प्रशिक्षण सटीकता है लेकिन कम परीक्षण सटीकता है, तो समस्या क्या हो सकती है?

सत्र 6. वर्गीकरण मॉडल का निर्माण

(Building A Classification Model)

आइए शुरू से एक पूर्ण वर्गीकरण मॉडल बनाकर सब कुछ एक साथ रखते है। हम एक मजेदार, यंहा वास्तविक जीवन के उदाहरण का उपयोग करेंगे: यह भविष्यवाणी करना कि रिया (Riya) इस आधार पर "केंद्रित (Focused)" होगी या "विचलित (Distracted)" होगी कि उसके अंतिम भोजन (खाना) के कितने समय बीत चुका है।

6.1 समस्या: रिया की एकाग्रता (The Problem: Riya's Focus)

रिया देखती है कि कक्षा में उसकी एकाग्रता उसके भोजन के समय पर निर्भर करती है। वह:

विचलित (0) है भोजन के बाद पहले 2 घंटों के लिए (बहुत भरी हुई)।

केंद्रित (1) है भोजन के 3 से 5 घंटे बाद (बिल्कुल सही)।

विचलित (0) है फिर से 5 घंटे के बाद (बहुत भूखी)।

यह एक आदर्श **द्विआधारी वर्गीकरण (binary classification)** समस्या है।

6.2 चरण-दर-चरण मॉडल बनाना (Building the Model Step-by-Step)

चरण 1. डेटासेट बनाएँ (Create the Dataset)

हम रिया के अवलोकन के आधार पर एक छोटा डेटासेट बनाएंगे।

उदाहरण Python कोड के साथ

```
import numpy as np
# Input: Minutes after meal
X = np.array([30, 60, 90, 120, 150, 180, 210, 240, 270, 300, 330, 360, 390, 420, 450])
# Output: 0 = Distracted, 1 = Focused
y = np.array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0])
```

चरण 2. डेटा विभाजित करें (Split the Data)

उदाहरण Python कोड के साथ

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

चरण 3. Keras के साथ मॉडल बनाएँ (Build the Model with Keras)

हम एक छिपी हुई परत वाले सरल नेटवर्क का उपयोग करेंगे।

उदाहरण Python कोड के साथ

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
model = Sequential([
    Dense(8, activation='relu', input_dim=1), # Hidden layer with 8 neurons
    Dense(1, activation='sigmoid')          # Output layer for binary classification
])
```

चरण 4. मॉडल संकलित करें (Compile the Model)

उदाहरण Python कोड के साथ

```
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

चरण 5. मॉडल को प्रशिक्षित करें (Train the Model)

उदाहरण Python कोड के साथ

```
history = model.fit(X_train, y_train, epochs=200, verbose=0)
```

चरण 6. मॉडल का मूल्यांकन करें (Evaluate the Model)

उदाहरण Python कोड के साथ

```
loss, acc = model.evaluate(X_test, y_test, verbose=0)
print(f"Test Accuracy: {acc:.2f}")
```

चरण 7: विस्तृत मेट्रिक्स प्राप्त करें (Get Detailed Metrics)

उदाहरण Python कोड के साथ

```
from sklearn.metrics import confusion_matrix, precision_score, recall_score
# Get predictions
y_pred_prob = model.predict(X_test)
y_pred = (y_pred_prob > 0.5).astype(int).flatten()
# Calculate metrics
cm = confusion_matrix(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
print("Confusion Matrix:\n", cm)
```

```
print(f"Precision: {precision:.2f}, Recall: {recall:.2f}")
```

6.3 मेट्रिक्स को समझना (Understanding the Metrics)

सटीकता (Accuracy): (TP + TN) / (Total)। मॉडल कितनी बार समग्र रूप से सही है?

परिशुद्धता (Precision): TP / (TP + FP)। जब मॉडल "केंद्रित" की भविष्यवाणी करता है, तो यह कितनी बार सही होता है? (गुणवत्ता का माप)।

पुनर्स्मरण (Recall): TP / (TP + FN)। उन सभी बारों में से जब रिया वास्तव में "केंद्रित" थी, मॉडल ने कितने को पकड़ा? (मात्रा का माप)।

6.4 एक सामान्य समस्या: अति-अनुकूलन (Overfitting) और इसे कैसे ठीक करें (A Common Problem: Overfitting and How to Fix It)

कभी-कभी, एक मॉडल प्रशिक्षण डेटा को बहुत अच्छी तरह से सीख लेता है, जिसमें उसका शोर और बाहरी मान भी शामिल हो जाते हैं। इसे अति-अनुकूलन (overfitting) कहा जाता है। यह प्रशिक्षण डेटा पर बहुत अच्छा प्रदर्शन करता है, लेकिन नए, अनदेखे परीक्षण डेटा पर खराब प्रदर्शन करता है। यह एक विद्यार्थी की तरह है जो विषय को समझे बिना उत्तरों को याद कर लेता है।

ड्रॉपआउट (Dropout) अति-अनुकूलन को रोकने का एक सरल और शक्तिशाली तरीका है।

ड्रॉपआउट (Dropout): प्रशिक्षण के दौरान, यह एक परत में न्यूरॉन्स के एक प्रतिशत को बेतरतीब ढंग से "बंद" कर देता है। यह नेटवर्क को किसी एक न्यूरॉन पर बहुत अधिक निर्भर न रहने के लिए मजबूर करता है और अधिक मजबूत विशेषताएँ सीखता है।

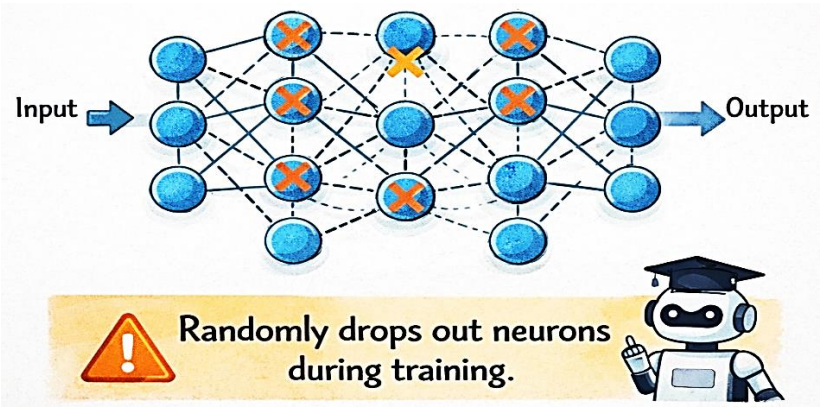
आइए अपने मॉडल में एक ड्रॉपआउट परत जोड़ें और देखें कि क्या इससे मदद मिलती है।

उदाहरण Python कोड के साथ

```
from tensorflow.keras.layers import Dropout
model_dropout = Sequential([
    Dense(8, activation='relu', input_dim=1),
    Dropout(0.5), # This will randomly turn off 50% of the neurons during training!
    Dense(1, activation='sigmoid')
])

model_dropout.compile(optimizer='adam',                    loss='binary_crossentropy',
metrics=['accuracy'])
history_dropout = model_dropout.fit(X_train, y_train, epochs=200, verbose=0)
loss_do, acc_do = model_dropout.evaluate(X_test, y_test, verbose=0)
print(f"With Dropout - Test Accuracy: {acc_do:.2f}")
```

© PSSCIVE Draft

Dropout helps make the model more general and reliable.



चित्र 6.1 एक न्यूरल नेटवर्क में ड्रॉपआउट की अवधारणा दिखाने वाली छवि]

सारांश (Summary)

- हमने एक पूर्ण द्विआधारी वर्गीकरण मॉडल बनाया।
- मूल्यांकन के लिए प्रमुख मेट्रिक्स सटीकता, परिशुद्धता, पुनर्स्मरण और भ्रम मैट्रिक्स हैं।
- अति-अनुकूलन (Overfitting) तब होता है जब कोई मॉडल प्रशिक्षण डेटा पर अच्छा प्रदर्शन करता है लेकिन परीक्षण डेटा पर खराब प्रदर्शन करता है।
- ड्रॉपआउट अति-अनुकूलन को रोकने के लिए एक सरल और प्रभावी तकनीक है, जो प्रशिक्षण के दौरान न्यूरॉन्स को बेतरतीब ढंग से बंद कर देती है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. कौन सी मीट्रिक इस प्रश्न का उत्तर देती है: "जितनी बार मॉडल ने 'केंद्रित' कहा, उनमें से कितनी बार वह सही था?" (अ) सटीकता (ब) परिशुद्धता (स) पुनर्स्मरण (द) हानि
2. एक मॉडल जो प्रशिक्षण डेटा को याद कर लेता है लेकिन नए डेटा पर विफल हो जाता है, वह निम्न में से पीड़ित है। (अ) अल्प-अनुकूलन (ब) अति-अनुकूलन (स) उच्च पूर्वाग्रह (द) कम भिन्नता
3. ड्रॉपआउट (Dropout) परत क्या करती है? (अ) यह नेटवर्क में अधिक न्यूरॉन जोड़ती है। (ब) यह अति-अनुकूलन को रोकने के लिए प्रशिक्षण के दौरान न्यूरॉन्स को बेतरतीब ढंग से बंद कर देती है। (स) यह सीखने की दर को बढ़ाती है। (द) यह इनपुट डेटा को सामान्यीकृत करती है।
4. द्विआधारी वर्गीकरण के लिए, आउटपुट परत में किस सक्रियण फ़ंक्शन का उपयोग किया जाना चाहिए? (अ) ReLU (ब) Tanh (स) Sigmoid (द) Linear
5. भ्रम मैट्रिक्स हमें समझने में मदद करता है? (अ) केवल समग्र सटीकता (ब) सही और गलत भविष्यवाणियों के प्रकार (स) हानि मान (द) युगों की संख्या।

ब. रिक्त स्थान भरें

1. रिया के उदाहरण में, आउटपुट या तो "केंद्रित" या "विचलित" था, जिसे _____ वर्गीकरण कहा जाता है।
2. मीट्रिक _____ उन वास्तविक सकारात्मक मामलों के अनुपात को मापता है जिन्हें सही ढंग से पहचाना गया था।
3. _____ एक नियमितीकरण (regularization) तकनीक है जो अति-अनुकूलन को रोकने में मदद करती है।
4. एक मॉडल जो _____ है, वह प्रशिक्षण डेटा पर अच्छा प्रदर्शन करता है लेकिन परीक्षण डेटा पर खराब प्रदर्शन करता है।
5. रिया की द्विआधारी वर्गीकरण समस्या के लिए उपयोग किया जाने वाला हानि फ़ंक्शन _____ था।

स. सत्य या असत्य बताएँ

1. प्रशिक्षण सेट पर पूर्ण सटीकता वाला मॉडल हमेशा सबसे अच्छा मॉडल होता है।
2. कुछ मामलों में परिशुद्धता (precision) और पुनर्स्मरण (recall) अकेले सटीकता की तुलना में अधिक जानकारीपूर्ण होते हैं।
3. ड्रॉपआउट का उपयोग प्रशिक्षण और परीक्षण दोनों के दौरान किया जाता है।
4. अति-अनुकूलन (Overfitting) का अर्थ है कि मॉडल डेटा सीखने के लिए बहुत सरल है।
5. द्विआधारी वर्गीकरण के लिए भ्रम मैट्रिक्स 2x2 तालिका है।

द. लघु उत्तरीय प्रश्न

1. परिशुद्धता (precision) और पुनर्स्मरण (recall) के बीच अंतर स्पष्ट कीजिए।
2. अति-अनुकूलन (overfitting) क्या है और यह एक समस्या क्यों है?
3. ड्रॉपआउट (Dropout) परत जोड़ने से न्यूरल नेटवर्क को कैसे मदद मिलती है?
4. एक ऐसी स्थिति का वर्णन करें जहाँ उच्च सटीकता भ्रामक हो सकती है और भ्रम मैट्रिक्स अधिक उपयोगी होगा।
5. एक वर्गीकरण मॉडल के निर्माण और मूल्यांकन की पूरी कार्यप्रवाह का वर्णन अपने शब्दों में कीजिए।

© PSSCI

मॉड्यूल 5. एआई परियोजना

(AI Project)

मॉड्यूल संक्षिप्त परिचय (Module Overview)

आज के डेटा-संचालित दुनिया में, कृत्रिम बुद्धिमत्ता (AI) जटिल समस्याओं को हल करने और उद्योगों को बदलने के लिए एक शक्तिशाली उपकरण बन गया है। AI परियोजनाओं पर यह इकाई उन बुद्धिमान प्रणालियों के विकास की पड़ताल करती है जो उन कार्यों को कर सकती हैं जिनके लिए आमतौर पर मानव संज्ञानात्मक क्षमताओं, जैसे सीखना, तर्क करना और निर्णय लेना की आवश्यकता होती है।

इकाई एक AI प्रोजेक्ट के संपूर्ण जीवनचक्र को कवर करती है। इसमें मूल समस्या को परिभाषित करना, डेटा एकत्र करना, एक मॉडल चुनना, उसे प्रशिक्षित करना, उसके प्रदर्शन का मूल्यांकन करना और वास्तविक दुनिया के अनुप्रयोगों के लिए उसे तैनात करना शामिल है। यह व्यवस्थित प्रक्रिया सफल AI विकास के लिए आवश्यक है।

इकाई यह पता लगाएगी कि AI परियोजनाएँ पारंपरिक सॉफ्टवेयर विकास से कैसे भिन्न होती हैं। प्रोजेक्ट विकास प्रक्रिया, प्रोजेक्ट प्रारूप और प्रोजेक्ट रिपोर्ट कैसे लिखी जाए। इस इकाई में एक सैम्पल प्रोजेक्ट भी शामिल है, जो विद्यार्थियों को यह विचार देगी कि AI प्रोजेक्ट कैसे विकसित की जाए।

इस इकाई को पूरा करने पर, आपको संपूर्ण AI प्रोजेक्ट पारिस्थितिकी तंत्र की व्यापक समझ होगी। यह आपको अपना AI विकास शुरू करने और कृत्रिम बुद्धिमत्ता के तेजी से विकसित हो रहे क्षेत्र में योगदान करने के लिए आवश्यक ज्ञान और कौशल प्रदान करेगा।

अधिगम के परिणाम (Learning Outcomes)

इस मॉड्यूल को पूरा करने के बाद, आप निम्नलिखित में सक्षम होंगे:

- कृत्रिम बुद्धिमत्ता (AI) परियोजनाओं की अवधारणा और महत्व को समझेंगे।
- AI प्रोजेक्ट चक्र (AI Project Cycle) और इसके विभिन्न चरणों की व्याख्या करना।
- AI का उपयोग करके वास्तविक दुनिया की समस्याओं को हल करने में शामिल चरणों की पहचान करना।
- AI विकास और पारंपरिक सॉफ्टवेयर विकास के बीच अंतर करना।
- एक सरल AI-आधारित प्रोजेक्ट की योजना बनाना, डिजाइन करना और उसे क्रियान्वित करना।
- एक संरचित AI प्रोजेक्ट रिपोर्ट तैयार करना।
- AI मॉडल के प्रदर्शन का विश्लेषण और मूल्यांकन करना।

मॉड्यूल संरचना (Module Structure)

सत्र 1. AI प्रोजेक्ट चक्र (AI Project Cycle)

सत्र 2. प्रोजेक्ट कार्य (Project Work)

सत्र 3. सैम्पल प्रोजेक्ट (Sample Project)

सत्र 1. AI प्रोजेक्ट चक्र

(AI Project Cycle)

1.1 परिचय

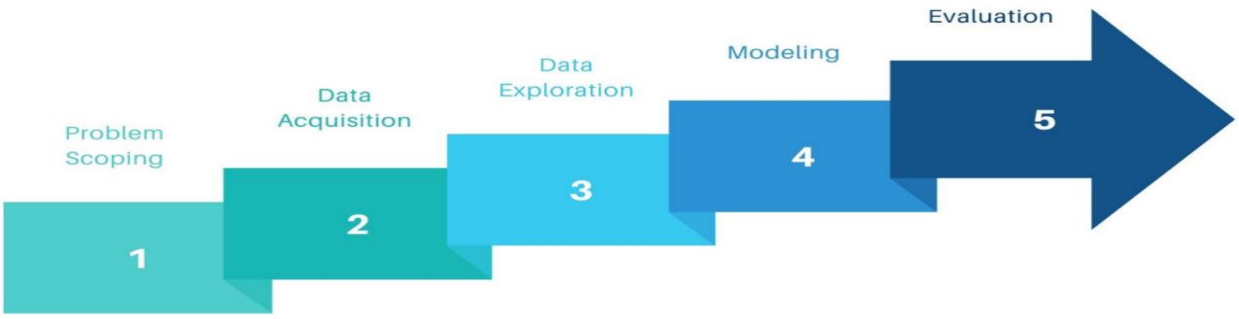
AI प्रोजेक्ट चक्र (साइकिल) संगठनों और व्यक्तियों को किसी समस्या का समाधान करने में सहायता करती है, साथ ही एक AI प्रोजेक्ट विकसित करने में मार्गदर्शन देती है। यह चरण-दर-चरण प्रक्रिया AI विकास जीवनचक्र (AI Development Lifecycle) के रूप में भी जानी जाती है, जो लक्ष्यों को प्राप्त करने के लिए एक संरचित रणनीति प्रदान करती है।

AI प्रोजेक्ट जीवनचक्र कृत्रिम बुद्धिमत्ता (Artificial Intelligence) आधारित समाधानों के विकास और उनके कार्यान्वयन (deployment) के लिए एक व्यवस्थित विधि है। AI प्रोजेक्ट मुख्य रूप से डेटा पर निर्भर होते हैं, इनमें निरंतर सुधार (iterative improvements) शामिल होते हैं और अक्सर प्रयोगात्मक (experimental) दृष्टिकोण की आवश्यकता होती है। यह जीवनचक्र सुनिश्चित करता है कि AI समाधान मजबूत (robust) हों और लगातार उपयोगी मूल्य प्रदान करें।

AI प्रोजेक्ट साइकिल एक संरचित प्रक्रिया है, जिसमें समस्या की पहचान (problem definition), डेटा संग्रह (data collection), मॉडल विकास (model development), मूल्यांकन (evaluation), कार्यान्वयन (deployment) और निगरानी (monitoring) जैसे चरण शामिल होते हैं, ताकि प्रभावी कृत्रिम बुद्धिमत्ता आधारित समाधान तैयार और प्रबंधित किए जा सकें।

1.2 AI प्रोजेक्ट चक्र के चरण (Stages of the AI Project Cycle)

AI प्रोजेक्ट चक्र के पाँच चरण हैं। आइए प्रत्येक चरण को एक-एक करके देखें।



चित्र 1.1: AI प्रोजेक्ट चक्र के चरण

1.2.1. समस्या क्षेत्र निर्धारण (Problem Scoping)

समस्या हर जगह मौजूद है। यह छोटी या बड़ी हो सकती है। यह आसान या गंभीर हो सकती है। हर समस्या का समाधान होता है। उसके लिए उसे हल करना होता है। प्रोजेक्ट का मुख्य उद्देश्य किसी समस्या का समाधान खोजना होता है। किसी भी समस्या को हल करने के लिए, पहले हमें उसका विश्लेषण करना होगा। समस्या का विश्लेषण करने के लिए किसी को समस्या के दायरे को स्पष्ट रूप से पहचानना चाहिए और उसे हल करने का एक दृष्टिकोण होना चाहिए। समस्या क्षेत्र निर्धारण (Problem Scoping) समस्या के दायरे का पता लगाने के बारे में है। समस्या क्षेत्र निर्धारण एक चिंता या मुद्दे को उजागर करता है जिसे कृत्रिम बुद्धिमत्ता के साथ संभालने की आवश्यकता है। इसमें उद्देश्यों को स्पष्ट करना और उन्हें प्राप्त करने की रणनीति भी शामिल है। इसे सफलतापूर्वक लागू करने के लिए विशिष्ट मुद्दे की मजबूत समझ की आवश्यकता होती है। 4Ws समस्या कैनवास (4Ws problem canvas) एक दृष्टिकोण है जो चिंता या समस्या की बेहतर समझ प्राप्त करने में सहायता करता है।

4WS समस्या कैनवास (The 4WS Problem Canvas)

4Ws समस्या कैनवास समस्या से संबंधित प्रमुख तत्वों की पहचान करने में मदद करता है। 4Ws में *Who*, *What*, *Where* और *Why* शामिल हैं।

चित्र 1.2 4WS समस्या कैनवास

- **Who (कौन):** पहला W उन लोगों का मूल्यांकन करता है जो किसी मुद्दे से प्रभावित होते हैं और समाधान से लाभान्वित होंगे। इन लोगों को हितधारक (stakeholders) भी कहा जाता है और उनके बारे में जानकारी इकट्ठा करना अनिवार्य है। हितधारक वे लोग होते हैं जो इस समस्या का सामना करते हैं और समाधान से लाभान्वित होंगे। यह ब्लॉक निम्नलिखित प्रश्नों को संबोधित करेगा।

  हितधारक कौन हैं?

  आप उनके बारे में क्या जानते हैं?

- **What (क्या):** इस खंड में समस्या का मूल्यांकन करना और उसका गहन अध्ययन करना शामिल है। इस स्तर पर, आपको समस्या की प्रकृति निर्धारित करने की आवश्यकता है। समस्या क्या है और आपको कैसे पता चलेगा कि यह एक समस्या है? उल्लिखित उद्देश्यों को प्राप्त करने के लिए जानकारी में गहराई से जाना महत्वपूर्ण है। चयनित समस्या वास्तव में मौजूद है यह साबित करने के लिए आपको साक्ष्य भी एकत्र करने की आवश्यकता है। समाचार पत्र लेख, मीडिया, घोषणाएँ आदि कुछ उदाहरण हैं। यह ब्लॉक निम्नलिखित प्रश्नों को संबोधित करेगा।

  समस्या क्या है?

  आपको कैसे पता चलेगा कि यह एक समस्या है? साक्ष्य दें।

- **Where (कहाँ):** इसमें यह शामिल है कि समस्या कहाँ और कब हुई है ताकि स्थिति को समझा जा सके और आवश्यक पैटर्न का पता लगाया जा सके। आपको समस्या के संदर्भ/स्थिति/स्थान पर ध्यान केंद्रित करने की आवश्यकता है। इससे समस्या उत्पन्न होने वाली स्थिति, उसके संदर्भ और उन स्थानों को देखने में मदद मिलेगी जहाँ यह प्रमुख है। यह ब्लॉक निम्नलिखित प्रश्नों को संबोधित करेगा।

  हितधारकों को समस्या का अनुभव किस संदर्भ/स्थिति में होता है?

  समस्या कहाँ स्थित है?

- **Why (क्यों):** अंतिम W यह निर्धारित करता है कि वास्तव में समाधान से किसे लाभ होगा, किस समस्या को ठीक करने की आवश्यकता है और समाधान कहाँ लागू किया जाना चाहिए। इस कैनवास में, सोचें कि समाधान से हितधारकों के साथ-साथ समाज को कैसे लाभ होगा। यह ब्लॉक निम्नलिखित प्रश्नों को संबोधित करेगा।

  यह समाधान हितधारकों के लिए क्यों मूल्यवान होगा?

  समाधान उनकी स्थिति में कैसे सुधार करेगा?

1.2.2 डेटा अधिग्रहण (Data Acquisition)

प्रोजेक्ट प्रक्रिया में दूसरा चरण उस जानकारी को एकत्र करने के बारे में है जो प्रोजेक्ट के लिए आवश्यक है। AI प्रणाली को सटीक डेटा के साथ प्रशिक्षित करना चाहिए ताकि वह भविष्यवाणियाँ करने में सक्षम हो सके।

मान लीजिए, कोई एक ऐसी प्रणाली बनाना चाहता है जो किसी कर्मचारी की भविष्य की आय की भविष्यवाणी करने में सक्षम हो। इसे प्राप्त करने के लिए, किसी के पास उस विशेष कर्मचारी की ऐतिहासिक वेतन जानकारी भी होनी चाहिए। हम इस पिछले वेतन डेटा को 'प्रशिक्षण डेटा' कहते हैं, जबकि भविष्य की भविष्यवाणी करने के लिए उपयोग किए जाने वाले डेटा को 'परीक्षण डेटा' के रूप में जाना जाता है।

आप जिन विशिष्ट विवरणों को एकत्र करना चाहते हैं, उन्हें डेटा विशेषताएँ (data features) कहा जाता है। हमारे उदाहरण में, इनमें कर्मचारी का वेतन, उन्हें मिली प्रतिशत वृद्धि, वृद्धि के बीच का समय, कोई बोनस आदि शामिल हो सकते हैं। यह डेटा एकत्र करने के विभिन्न तरीके हैं, जैसे:

चित्र 1.3: यह डेटा एकत्र करने के विभिन्न तरीके

1.2.3. डेटा अन्वेषण (Data Exploration)

डेटा कन्फ़्युशन हो सकता है और भ्रम पैदा कर सकता है, खासकर संख्याओं से जुड़ा डेटा। डेटा विज़ुअलाइज़ेशन का उपयोग जटिल जानकारी का निरीक्षण करने के लिए किया जा सकता है। यह संख्याओं को दृश्यों में बदल देता है जिन्हें समझना आसान होता है। डेटा को अधिक उपयोगकर्ता-अनुकूल तरीके से प्रस्तुत करने के लिए इसे *बार ग्राफ, हिस्टोग्राम, पाई* और *लाइन चार्ट* जैसे दृश्यों में प्रस्तुत किया जाता है।

चित्र 1.4 : डेटा विज़ुअलाइज़ेशन

यह सक्षम बनाता है:

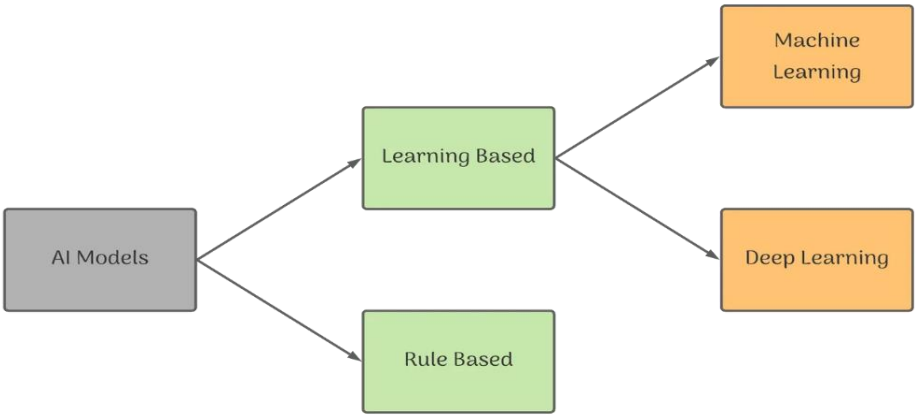
- **सही उपकरण चुनना** - विज़ुअलाइज़ेशन यह पता लगाने में सहायता करता है कि विश्लेषण के लिए कौन सा सही मॉडल या विधि काम करती है।
- **रुझानों का पता लगाना** - ग्राफ या चार्ट जैसे दृश्य यह प्रदर्शित करने के लिए उपयोगी होते हैं कि क्या कुछ बढ़ रहा है, घट रहा है या स्थिर है, जिससे आपको रुझानों की पहचान करने में मदद मिलती है।
- **निष्कर्ष साझा करना** - एक बार जब आप अंतर्दृष्टि प्राप्त कर लेते हैं, तो दृश्य अपने निष्कर्षों को दूसरों को समझाना आसान बनाते हैं।

1.2.4. मॉडलिंग (Modeling)

मॉडलिंग कंप्यूटर के लिए जटिल डेटा को सरल बनाने के लिए AI प्रोजेक्ट प्रक्रिया में एक महत्वपूर्ण तत्व है। यह कंप्यूटर को समझने और सही भविष्यवाणियाँ करने में सक्षम बनाता है। प्रारंभ में, डेटा को पैटर्न स्पॉटिंग के लिए चार्ट या ग्राफ़ में दिखाया जा सकता है। AI सिस्टम के कार्य करने के लिए, हमें इस डेटा को ऐसे प्रारूप में बदलना होगा जिसके साथ कंप्यूटर काम कर सकें, जिसका अर्थ आमतौर पर इसे बाइनरी (0 और 1) में परिवर्तित करना होता है।

मॉडलिंग एक गणितीय ढांचा बनाने के बारे में है जो दर्शाता है कि विभिन्न डेटा बिंदु एक दूसरे से कैसे संबंधित हैं। यह उसी तरह है जैसे कंप्यूटर को पैटर्न पहचानने या विकल्प चुनने के लिए सिखाया जाता है। ये मॉडल सरल समीकरणों से लेकर जटिल न्यूरल नेटवर्क तक हो सकते हैं, जो इस बात पर निर्भर करता है कि आपको क्या हासिल करना है। एक बार जब सिस्टम ऐतिहासिक डेटा से नियम सीख लेता है, तो वह नए डेटा के साथ भविष्यवाणियाँ कर सकता है।

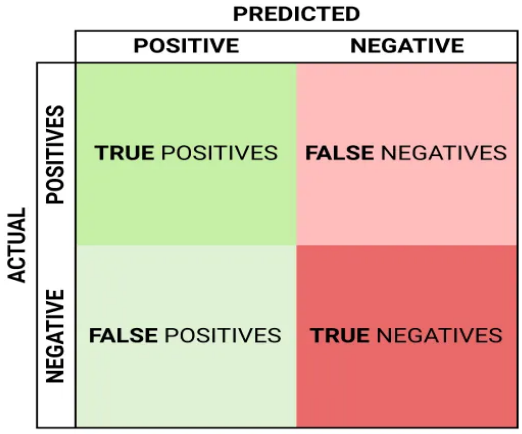
सामान्य तौर पर, AI मॉडल को निम्नानुसार वर्गीकृत किया जा सकता है:



चित्र 1.5 : AI मॉडलों का वर्गीकरण

1.2.5. मूल्यांकन (Evaluation)

मूल्यांकन AI प्रोजेक्ट प्रक्रिया का अंतिम चरण है। एक मॉडल बनाने और प्रशिक्षित करने के बाद, यह देखने के लिए इसे अच्छी तरह से परखना आवश्यक है कि यह कितना अच्छा प्रदर्शन करता है। इसके लिए, हम परीक्षण डेटा नामक एक अलग डेटासेट का उपयोग करते हैं। हम मॉडल के प्रदर्शन का मूल्यांकन कई मानदंडों के आधार पर करते हैं:



चित्र 1.6: मॉडल के प्रदर्शन के मूल्यांकन के मानदंड

- **सटीकता (Accuracy)** - यह संपूर्ण डेटासेट में से सही भविष्यवाणियों के प्रतिशत की गणना करता है और मॉडल की सटीकता की पूरी तस्वीर प्रदान करता है।
- **परिशुद्धता (Precision)** - यह मापता है कि उसकी सकारात्मक भविष्यवाणियों में से कितनी सही थीं।
- **F1 स्कोर (F1 Score)** - यह मीट्रिक परिशुद्धता और पुनर्स्मरण को जोड़ती है, जो तब उपयोगी होता है जब एक वर्ग दूसरे पर भारी रूप से दर्शाया जाता है। यह गलत सकारात्मक और गलत नकारात्मक के बीच संतुलन बनाने में सक्षम बनाता है।
- **पुनर्स्मरण (Recall)** - संवेदनशीलता के रूप में भी जाना जाता है, यह मीट्रिक दिखाता है कि मॉडल सभी सकारात्मक उदाहरणों को कितनी अच्छी तरह से सही ढंग से पहचानता है। यह सकारात्मक मामलों को पकड़ने की इसकी क्षमता को इंगित करता है।

सारांश (Summary)

यह सत्र AI प्रोजेक्ट साइकिल का परिचय कराता है, जो AI समाधान बनाने के लिए उपयोग की जाने वाली एक चरण-दर-चरण प्रक्रिया है। इसमें विद्यार्थी समस्या की पहचान (problem definition), डेटा संग्रह (data collection), डेटा विश्लेषण (data analysis), मॉडल निर्माण (model building) और मूल्यांकन (evaluation) के बारे में सीखते हैं। यह AI की सहायता से वास्तविक जीवन की समस्याओं को हल करने के लिए एक संरचित दृष्टिकोण प्रदान करता है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. AI प्रोजेक्ट चक्र में आम तौर पर पहला चरण निम्नलिखित में से कौन सा होता है? (अ) डेटा अधिग्रहण (ब) मॉडल मूल्यांकन (स) समस्या क्षेत्र निर्धारण (द) मॉडल तैनाती

2. समस्या क्षेत्र निर्धारण चरण के दौरान, समस्या को उसके प्रमुख तत्वों पर विचार करके तोड़ने के लिए अक्सर किस उपकरण का उपयोग किया जाता है? (अ) डेटा विज़ुअलाइज़ेशन चार्ट (ब) सिस्टम मैप (स) 4Ws समस्या कैनवास (द) न्यूरल नेटवर्क
3. कौन सा चरण AI मॉडल को प्रशिक्षित करने के लिए उच्च-गुणवत्ता वाले डेटासेट को एकत्र करने और तैयार करने पर केंद्रित है? (अ) डेटा अन्वेषण (ब) डेटा अधिग्रहण और तैयारी (स) मॉडल प्रशिक्षण (द) समस्या क्षेत्र निर्धारण
4. AI प्रोजेक्ट चक्र में डेटा अन्वेषण (Data Exploration) का उद्देश्य क्या है? (अ) समस्या विवरण को परिभाषित करना (ब) AI मॉडल को प्रशिक्षित करना (स) डेटा विशेषताओं को समझना, पैटर्न की पहचान करना और विसंगतियों का पता लगाना (द) AI मॉडल को तैनात करना
5. निम्नलिखित में से कौन सा AI मॉडलिंग में एक विशिष्ट प्रकार का शिक्षण-आधारित दृष्टिकोण *नहीं* है? (अ) पर्यवेक्षित शिक्षण (ब) अप्रशिक्षित शिक्षण (स) सुदृढीकरण शिक्षण (द) नियम-आधारित शिक्षण
6. जिस चरण में प्रशिक्षित मॉडल के प्रदर्शन का अनदेखे डेटा का उपयोग करके आकलन किया जाता है, उसे क्या कहा जाता है? (अ) डेटा अधिग्रहण (ब) मॉडल प्रशिक्षण (स) मॉडल मूल्यांकन (द) समस्या क्षेत्र निर्धारण
7. AI परियोजनाओं में निरंतर निगरानी और रखरखाव (Continuous Monitoring and Maintenance) क्यों आवश्यक है? (अ) यह सुनिश्चित करता है कि मॉडल जल्दी से तैनात किया जाए। (ब) यह मॉडल ड्रिफ्ट (model drift) का पता लगाने और समय के साथ प्रदर्शन बनाए रखने में मदद करता है। (स) यह केवल छवि पहचान कार्यों के लिए आवश्यक है। (द) यह मुख्य रूप से प्रारंभिक मॉडल प्रशिक्षण के लिए है।
8. AI प्रोजेक्ट निगरानी के संदर्भ में "मॉडल ड्रिफ्ट (model drift)" का क्या अर्थ है? (अ) मॉडल बहुत जटिल होता जा रहा है। (ब) मॉडल को नई सुविधाओं के साथ अद्यतन किया जा रहा है। (स) वास्तविक दुनिया के डेटा में परिवर्तन के कारण मॉडल का प्रदर्शन समय के साथ कम हो जाता है। (द) मॉडल एक अलग सर्वर पर जा रहा है।
9. निम्नलिखित में से कौन सा AI मॉडल का मूल्यांकन करने के लिए उपयोग किया जाने वाला एक सामान्य मीट्रिक है, विशेष रूप से वर्गीकरण कार्यों के लिए? (अ) माध्य वर्ग त्रुटि (MSE) (ब) सटीकता (स) कम्प्यूटेशनल लागत (द) विलंबता (Latency)
10. AI प्रोजेक्ट जीवनचक्र को एक पुनरावृत्तीय (iterative) प्रक्रिया के रूप में वर्णित किया गया है। इसका क्या अर्थ है? (अ) प्रत्येक चरण केवल एक बार पूरा किया जाता है। (ब) प्रक्रिया रैखिक है और चरणों पर पुनर्विचार करने की अनुमति नहीं देती है। (स) मॉडल को परिष्कृत करने और परिणामों में सुधार करने के लिए पूरी प्रक्रिया के दौरान अक्सर चरणों पर पुनर्विचार किया जाता है। (द) केवल तैनाती चरण पुनरावृत्तीय है।

ब. रिक्त स्थान भरें

1. AI प्रोजेक्ट चक्र का पहला चरण _____ कहलाता है।
2. _____ के दौरान, व्यावसायिक समस्या, प्रोजेक्ट के उद्देश्यों और सफलता के मानदंडों को परिभाषित किया जाता है।
3. _____ प्रारंभिक चरण में समस्या को कौन, क्या, कहाँ और क्यों पर विचार करके तोड़ने के लिए अक्सर उपयोग किया जाने वाला एक उपकरण है।
4. _____ चरण में, प्रोजेक्ट का आधार बनाने के लिए विभिन्न स्रोतों से डेटा एकत्र किया जाता है।

5. _____ में एकत्रित डेटा में लापता मानों, असंगतियों और त्रुटियों को संभालना शामिल है।
6. डेटा को चित्रमय रूप से प्रस्तुत करके रुझानों और पैटर्न की पहचान करने के चरण को _____ कहा जाता है।
7. _____ में उपयुक्त AI मॉडल का चयन करना और उसे तैयार किए गए डेटा में फिट करना शामिल है।
8. _____ प्रशिक्षित मॉडल के प्रदर्शन का अनदेखे डेटा पर परीक्षण करने की प्रक्रिया है।
9. _____ वह चरण है जहाँ AI मॉडल को उत्पादन वातावरण में एकीकृत किया जाता है।
10. निरंतर _____ मॉडल ड्रिफ्ट का पता लगाने और समय के साथ प्रदर्शन बनाए रखने में मदद करता है।
11. AI प्रोजेक्ट चक्र स्वाभाविक रूप से एक _____ प्रक्रिया है, जो मॉडल को परिष्कृत करने के लिए चरणों पर पुनर्विचार करने की अनुमति देती है।

स. सत्य या असत्य बताएँ

1. AI प्रोजेक्ट चक्र एक रैखिक प्रक्रिया है जहाँ प्रत्येक चरण केवल एक बार पूरा किया जाता है।
2. समस्या क्षेत्र निर्धारण (Problem Scoping) AI प्रोजेक्ट चक्र का पहला चरण है।
3. डेटा अधिग्रहण (Data Acquisition) मुख्य रूप से AI मॉडल को एल्गोरिदम के साथ प्रशिक्षित करने पर केंद्रित है।
4. मॉडल तैनाती (Model Deployment) AI प्रोजेक्ट का अंतिम और पूर्ण समापन बिंदु है, जिसके बाद कोई और चरण नहीं होते हैं।
5. 4Ws समस्या कैनवास एक उपकरण है जिसका उपयोग डेटा अन्वेषण चरण के दौरान किया जाता है।
6. प्रशिक्षण डेटा और परीक्षण डेटा AI मॉडल के मूल्यांकन के लिए समान हो सकते हैं।
7. मॉडल मूल्यांकन (Model evaluation) AI मॉडल में संभावित पूर्वाग्रहों (biases) की पहचान करने में मदद करता है।
8. मॉडल ड्रिफ्ट (Model drift) से तात्पर्य है कि वास्तविक दुनिया के डेटा के संपर्क में आने के कारण मॉडल समय के साथ अधिक कुशल हो जाता है।
9. डीप लर्निंग (Deep learning) एक प्रकार का नियम-आधारित AI मॉडल है।
10. डेटा गुणवत्ता और मात्रा का AI मॉडल के प्रदर्शन पर कोई महत्वपूर्ण प्रभाव नहीं पड़ता है।

द. लघु उत्तरीय प्रश्न

1. AI प्रोजेक्ट साइकिल क्या है और इसका मुख्य उद्देश्य क्या है?
2. 4Ws Problem Canvas में शामिल चार तत्व कौन-कौन से हैं और उनका महत्व क्या है?
3. डेटा अधिग्रहण (Data Acquisition) में “training data” और “testing data” में क्या अंतर होता है?
4. डेटा अन्वेषण (Data Exploration) में डेटा विज़ुअलाइज़ेशन की क्या भूमिका होती है?

5. मॉडल मूल्यांकन (Evaluation) के लिए किन-किन मानकों (metrics) का उपयोग किया जाता है?

सत्र 2. प्रोजेक्ट कार्य
(Project Work)

2.1 परिचय

प्रोजेक्ट वर्क व्यक्तिगत विकास के लिए महत्वपूर्ण है। यह सीखी गई अवधारणाओं को वास्तविक दुनिया की स्थिति में लागू करने का एक व्यावहारिक अवसर प्रदान करता है। परियोजनाएँ कौशल की एक विस्तृत श्रृंखला विकसित करती हैं, जिसमें प्रोग्रामिंग, मशीन लर्निंग एल्गोरिदम और डेटा विश्लेषण तकनीकों में तकनीकी विशेषज्ञता शामिल है। किसी भी पाठ्यक्रम से उत्तीर्ण होने वाले विद्यार्थी को कम से कम एक प्रोजेक्ट पूरी करनी चाहिए। इससे उन्हें विषय को अच्छी तरह से जानने में मदद मिलेगी और व्यावहारिक अनुभव मिलेगा।

2.2 प्रोजेक्ट कार्य के बारे में (About the Project Work)

प्रोजेक्ट में कुछ क्रेडिट पॉइंट होते हैं और परीक्षा में इसका ग्रेडिंग किया जाता है। प्रोजेक्ट का मार्गदर्शन स्कूल शिक्षक द्वारा आंतरिक मार्गदर्शक के रूप में और उद्योग या व्यावसायिक दुनिया के विशेषज्ञ द्वारा बाह्य मार्गदर्शक के रूप में किया जाता है। विद्यार्थी को प्रोजेक्ट की प्रगति दिखाने के लिए स्कूल शिक्षक को रिपोर्ट करना होता है। प्रोजेक्ट के अंत में, विद्यार्थियों को अपने काम का एक दस्तावेज़ प्रोजेक्ट रिपोर्ट के रूप में तैयार करना होगा।

प्रोजेक्ट कार्य के उद्देश्य (Objectives of Project Work)

प्रोजेक्ट कार्य के अंतिम उद्देश्यों को इस प्रकार कहा जा सकता है:

- विद्यार्थियों को AI का उपयोग करके स्वतंत्र रूप से समस्याओं को तैयार करने और हल करने और परिणामों को लिखित और मौखिक दोनों रूपों में प्रस्तुत करने के लिए प्रशिक्षित करना।
- विद्यार्थियों को कार्य दुनिया (World of Work) में वास्तविक जीवन की समस्याओं से अवगत कराना।
- विद्यार्थियों को लोगों के साथ बातचीत करने और मानवीय संबंधों को समझने के अवसर प्रदान करना।

2.3 प्रोजेक्ट विकास प्रक्रिया (Project Development Process)

विद्यार्थियों को अपना प्रोजेक्ट कार्य कुछ ही महीने की अवधि के भीतर पूरा करना होता है। विकास प्रक्रिया में निम्नलिखित बिंदु शामिल होते हैं।

कृत्रिम बुद्धिमत्ता (AI) प्रोजेक्ट विकसित करने के लिए एक संरचित दृष्टिकोण की आवश्यकता होती है। AI प्रोजेक्ट विकास प्रक्रिया अवधारणा से लेकर तैनाती तक के चरणों को निम्नानुसार रेखांकित करती है।

2.3.1. समस्या परिभाषा और लक्ष्य निर्धारण (Problem definition and goal setting)

- उस समस्या या अवसर को स्पष्ट रूप से पहचानें जिसे AI प्रोजेक्ट संबोधित करना चाहती है।
- AI प्रणाली के लिए विशिष्ट, मापने योग्य, प्राप्त करने योग्य, प्रासंगिक और समयबद्ध (SMART) उद्देश्यों को परिभाषित करें।
- AI प्रोजेक्ट के लक्ष्यों को व्यापक उद्देश्यों और हितधारकों की अपेक्षाओं के साथ संरेखित करें।

- कौन प्रभावित है, समस्या क्या है, यह कहाँ और कब होती है और इसे हल करना क्यों महत्वपूर्ण है, इसका विश्लेषण करके समस्या की गहन समझ हासिल करने के लिए "4Ws समस्या कैनवास" जैसे उपकरणों का उपयोग करें।
- सफलता के मानदंड और प्रमुख प्रदर्शन संकेतक (KPIs) स्थापित करें।

2.3.2. डेटा संग्रह और अन्वेषण (Data collection and exploration)

- AI मॉडल को प्रशिक्षित और मूल्यांकन करने के लिए आवश्यक डेटा की पहचान करें।
- डेटाबेस, सर्वेक्षण, वेब स्क्रैपिंग, सेंसर, कैमरे और API जैसे विभिन्न स्रोतों से डेटा एकत्र करें।
- लापता मानों, असंगतियों और शोर को संभालकर डेटा गुणवत्ता सुनिश्चित करें।
- डेटा विज़ुअलाइज़ेशन और सांख्यिकीय विश्लेषण जैसी तकनीकों का उपयोग करके डेटा की विशेषताओं, पैटर्न और संभावित मुद्दों का पता लगाएँ और समझें।
- डेटा को प्रशिक्षण, सत्यापन और परीक्षण सेटों में विभाजित कीजिए।
- डेटा गुणवत्ता, मात्रा, नैतिक और कानूनी मानकों का अनुपालन, डेटा गोपनीयता नियमों सहित चुनौतियों का समाधान कीजिए।

2.3.3. डेटा तैयारी और प्रीप्रोसेसिंग (Data preparation and preprocessing)

- त्रुटियों, असंगतियों, डुप्लिकेट और अप्रासंगिक जानकारी को हटाकर डेटा साफ़ कीजिए।
- डेटा सामान्यीकरण, स्केलिंग और लापता मानों को संभालने जैसे कार्यों के माध्यम से मॉडल प्रशिक्षण के लिए डेटा को पूर्व-प्रक्रिया कीजिए।
- फीचर इंजीनियरिंग (feature engineering) कीजिए, जिसमें मॉडल के प्रदर्शन को बढ़ाने के लिए डेटा विशेषताओं का चयन, निर्माण या परिवर्तन करना शामिल है।
- विशेष रूप से पर्यवेक्षित शिक्षण परियोजनाओं के लिए, आवश्यकतानुसार डेटा को लेबल या एनोटेट कीजिए।

2.3.4. मॉडल डिजाइन और विकास (Model design and development)

- समस्या डोमेन, डेटा विशेषताओं और वांछित परिणामों के आधार पर उपयुक्त AI एल्गोरिदम और आर्किटेक्चर का चयन कीजिए।
- विभिन्न मॉडल प्रकारों, जैसे मशीन लर्निंग (ML), डीप लर्निंग (DL), या प्राकृतिक भाषा प्रसंस्करण (NLP) मॉडल आदि का पता लगाएँ और उनमें से चुनिए।
- AI मॉडल को डिजाइन करें और बनाएँ, संभावित रूप से पूर्व-मौजूदा मॉडल का पता लगाएँ या एक को शुरू से प्रशिक्षित कीजिए।
- मॉडल डिजाइन के दौरान कम्प्यूटेशनल संसाधनों, मॉडल व्याख्या और स्केलेबिलिटी जैसे कारकों पर विचार कीजिए।

2.3.5. मॉडल प्रशिक्षण और परीक्षण (Model training and testing)

- तैयार किए गए प्रशिक्षण डेटा का उपयोग करके मॉडल को प्रशिक्षित कीजिए।

- प्रदर्शन को अनुकूलित करने के लिए मॉडल मापदंडों को समायोजित करें और हाइपरपैरामीटर्स को ठीक कीजिए।
- अलग-अलग सत्यापन और परीक्षण डेटासेट पर मॉडल के प्रदर्शन और सटीकता का कड़ाई से परीक्षण कीजिए।
- उपयुक्त मेट्रिक्स (सटीकता, परिशुद्धता, पुनर्स्मरण, F1-स्कोर आदि) का उपयोग करके मॉडल के प्रदर्शन का मूल्यांकन कीजिए।
- अति-अनुकूलन (overfitting) को रोकने के लिए क्रॉस-वैलिडेशन कीजिए।
- मॉडल परिणामों की तुलना बेसलाइन विधियों से कीजिए।

2.3.6. तैनाती और एकीकरण (Deployment and integration)

- क्लाउड होस्टिंग, एज डिप्लॉयमेंट या हाइब्रिड दृष्टिकोण जैसे उपयुक्त तैनाती मंच का चयन कीजिए।
- प्रशिक्षित मॉडल को उत्पादन वातावरण में तैनात कीजिए जहाँ यह पहचानी गई समस्या का समाधान कर सके।
- मॉडल को मौजूदा अनुप्रयोगों, प्लेटफार्मों के साथ एकीकृत करें। मौजूदा अनुप्रयोगों के साथ एकीकरण के लिए API का उपयोग कीजिए।
- तैनाती के दौरान स्केलेबिलिटी, सुरक्षा और अनुपालन पर ध्यान दीजिए।

2.3.7. प्रदर्शन निगरानी और रखरखाव (Performance Monitoring and Maintenance)

- मॉडल सटीकता की निरंतर निगरानी कीजिए और आवश्यकतानुसार अद्यतन (अपडेट) कीजिए।
- मॉडल ड्रिफ्ट का पता लगाने के लिए स्वचालित अलर्ट स्थापित कीजिए।
- नए डेटा के साथ नियमित रूप से मॉडल को पुनः प्रशिक्षित कीजिए।
- पुनरुत्पादन के लिए लॉग और दस्तावेज़ीकरण बनाए रखें।

2.3.8. नैतिक विचार और अनुपालन (Ethical Considerations and Compliance)

- AI मॉडल में निष्पक्षता सुनिश्चित करें और पूर्वाग्रह को समाप्त कीजिए।
- AI शासन नीतियों और नैतिक दिशानिर्देशों का पालन कीजिए।
- निर्णय लेने की प्रक्रियाओं का दस्तावेजीकरण करके पारदर्शिता बनाए रखें।
- सुनिश्चित करें कि AI-संचालित निर्णय सामाजिक और व्यावसायिक नैतिकता के अनुरूप हों।

2.3.9. प्रोजेक्ट दस्तावेज़ीकरण और रिपोर्टिंग (Project Documentation and Reporting)

- कार्यप्रणाली, कोड और परिणामों सहित संपूर्ण दस्तावेज़ीकरण बनाए रखें।
- प्रोजेक्ट की प्रगति पर हितधारकों को समय-समय पर रिपोर्ट प्रदान कीजिए।
- अंतिम उपयोगकर्ताओं के लिए उपयोगकर्ता गाइड और तकनीकी दस्तावेज़ बनाएँ।
- भविष्य के संदर्भ के लिए स्रोत कोड और डेटासेट सुरक्षित रूप से संग्रहीत कीजिए।

2.3.10. सहयोग और टीम समन्वय (Collaboration and Team Coordination)

- टीम के सदस्यों को स्पष्ट भूमिकाएँ और जिम्मेदारियाँ सौंपें।
- Jira, Trello या Asana जैसे प्रोजेक्ट प्रबंधन उपकरणों का उपयोग कीजिए।
- Git के साथ संस्करण नियंत्रण बनाए रखें और नियमित कोड समीक्षाएँ कीजिए।
- डेटा वैज्ञानिकों, इंजीनियरों और व्यावसायिक टीमों के बीच क्रॉस-फंक्शनल सहयोग को बढ़ावा दें।

2.3.11. स्केलेबिलिटी और भविष्य के सुधार (Scalability and Future Improvements)

- भविष्य के संवर्द्धन के लिए स्केलेबिलिटी को ध्यान में रखते हुए मॉडल डिज़ाइन कीजिए।
- सिस्टम अपग्रेड और प्रदर्शन अनुकूलन की योजना बनाएं।
- मैनुअल प्रयासों को कम करने के लिए स्वचालन संभावनाओं का पता लगाएं।
- प्रोजेक्ट में नवाचारों को शामिल करने के लिए उभरते AI रुझानों पर नज़र रखें।

इन दिशानिर्देशों का पालन करके, AI परियोजनाओं को कुशलतापूर्वक निष्पादित किया जा सकता है, जिससे मजबूती, निष्पक्षता और दीर्घकालिक स्थिरता सुनिश्चित हो सकती है।

2.4 प्रोजेक्ट ka प्रारूप (Project Formats)

प्रोजेक्ट प्रारूप उन विभिन्न तरीकों को संदर्भित करते हैं जिनसे प्रोजेक्ट की जानकारी और दस्तावेज़ीकरण को संरचित, प्रस्तुत और संग्रहीत किया जा सकता है। प्रोजेक्ट दस्तावेज़ीकरण का प्रारूप प्रोजेक्ट की प्रकृति, लक्षित समूह और दस्तावेज़ के उद्देश्य पर निर्भर करता है।

AI प्रोजेक्ट के लिए निम्नलिखित नमूना सारांश (synopsis) दिया गया है। आप प्रोजेक्ट कार्य की शुरुआत में सारांश तैयार करने के लिए इसका उपयोग कर सकते हैं।

प्रोजेक्ट प्रारूप

प्रोजेक्ट का शीर्षक: <प्रोजेक्ट के लिए एक संक्षिप्त और स्पष्ट शीर्षक प्रदान करें।>

परिचय: <परियोजना, इसके उद्देश्य और इसके महत्व का एक संक्षिप्त परिचय।>

उद्देश्य / मौजूदा प्रणाली और प्रणाली की आवश्यकता

- <प्रोजेक्ट के मुख्य उद्देश्यों की सूची बनाएँ, यह निर्दिष्ट करते हुए कि यह क्या हासिल करना चाहती है।>

प्रोजेक्ट का दायरा

- <प्रोजेक्ट की सीमाओं को परिभाषित करें, जिसमें कार्यक्षमताएँ और सीमाएँ शामिल हैं।>

उपयोग की गई प्रौद्योगिकियाँ

- <उपयोग की गई प्रोग्रामिंग भाषाओं, फ्रेमवर्क, डेटाबेस और उपकरणों की रूपरेखा तैयार करें।>

प्रणाली वास्तुकला

- <प्रणाली डिजाइन का एक उच्च-स्तरीय विवरण, यदि लागू हो तो एक ब्लॉक आरेख सहित।>

मॉड्यूल और कार्यक्षमताएँ

- <प्रोजेक्ट को प्रमुख मॉड्यूल में विभाजित करें और उनकी भूमिकाओं और कार्यक्षमताओं का संक्षेप में वर्णन

करें।>

- मॉड्यूल 1: विवरण
- मॉड्यूल 2: विवरण
- मॉड्यूल 3: विवरण

कार्यप्रणाली

- <उपयोग किए गए सॉफ्टवेयर विकास मॉडल का वर्णन करें, जैसे, एजाइल, वॉटरफॉल आदि।>

अपेक्षित परिणाम

- <प्रोजेक्ट के अपेक्षित परिणामों और लाभों की व्याख्या करें।>

. **निष्कर्ष**

- <प्रोजेक्ट के महत्व और प्रभाव का सारांश।>

. **संदर्भ (यदि कोई हो)**

- <प्रोजेक्ट में संदर्भित पुस्तकों, वेबसाइटों या शोध पत्रों की सूची।>

2.5 प्रोजेक्ट की समीक्षा (Project Review)

एक प्रोजेक्ट समीक्षा अपने जीवनचक्र में एक विशिष्ट बिंदु पर किसी प्रोजेक्ट की प्रगति, प्रदर्शन और परिणामों का एक औपचारिक मूल्यांकन है। यह एक ऐसी प्रक्रिया है जो यह निर्धारित करने में मदद करती है कि प्रोजेक्ट अपने लक्ष्यों और उद्देश्यों को प्राप्त करने की राह पर है या नहीं और उन क्षेत्रों की पहचान करती है जिनमें सुधार की आवश्यकता है।

प्रोजेक्ट के मूल्यांकन के लिए निम्नलिखित प्रोजेक्ट समीक्षा चार्ट का उपयोग किया जा सकता है।

प्रोजेक्ट की समीक्षा चार्ट

प्रोजेक्ट चरण	मूल्यांकन मानदंड	रेटिंग (1-5)	टिप्पणियाँ
प्रोजेक्ट परिभाषा	समस्या कथन और उद्देश्यों की स्पष्टता		
	प्रोजेक्ट की व्यवहार्यता और प्रासंगिकता		
अनुसंधान और पृष्ठभूमि	साहित्य समीक्षा की पर्याप्तता		
	मौजूदा समाधानों के साथ तुलना		
डेटा संग्रह	डेटा स्रोतों की पहचान और सत्यापन		
	डेटा प्रीप्रोसेसिंग और सफाई		
कार्यप्रणाली	उपयोग किए गए ML/AI मॉडलों की उपयुक्तता		
	चुने गए एल्गोरिदम के लिए औचित्य		
कार्यान्वयन	मॉडल निष्पादन और प्रशिक्षण की सटीकता		

© PSSCIVE Draft

	समाधान की दक्षता और अनुकूलन		
मूल्यांकन और परीक्षण	प्रदर्शन मेट्रिक्स का आकलन		
	मॉडल सत्यापन और क्रॉस-वैलिडेशन		
तैनाती	तैनाती रणनीति (क्लाउड, एज, स्थानीय)		
	समाधान की स्केलेबिलिटी और उपयोगिता		
परिणाम और विश्लेषण	परिणामों की प्रस्तुति और विज़ुअलाइज़ेशन		
	निष्कर्षों की व्याख्या और चर्चा		
नैतिक विचार	पूर्वाग्रह शमन रणनीतियाँ		
	विनियमों का अनुपालन		
दस्तावेज़ीकरण	प्रोजेक्ट दस्तावेज़ीकरण की स्पष्टता और पूर्णता		
	संदर्भ और उद्धरण शामिल		
समग्र मूल्यांकन	प्रोजेक्ट नवाचार और विशिष्टता		

अंतिम स्कोर (Final Score): _____/-----

यह समीक्षा चार्ट AI प्रोजेक्ट के प्रत्येक चरण का आकलन करने के लिए एक संरचित दृष्टिकोण प्रदान करता है। रेटिंग (1-5) प्रोजेक्ट की गुणवत्ता और प्रभावशीलता का आकलन करने में मदद करती हैं।

सारांश (Summary)

इस सत्र में, विद्यार्थी एक प्रोजेक्ट विकसित करने के लिए अपने ज्ञान का अनुप्रयोग करते हैं। वे सीखते हैं कि किसी समस्या का चयन कैसे करें, डेटा एकत्र करें, एक मॉडल बनाएँ और एक प्रोजेक्ट रिपोर्ट तैयार करें। यह व्यावहारिक कार्यान्वयन और व्यावहारिक सीखने पर केंद्रित है।

अपनी प्रगति जांचें

अ. बहुविकल्पीय प्रश्न

1. प्रोजेक्ट विकास प्रक्रिया में किस चरण में प्रोजेक्ट के लक्ष्यों, दायरे और व्यवहार्यता की पहचान करना शामिल है? (अ) योजना (ब) निष्पादन (स) प्रारंभ (द) समापन
2. निगरानी और नियंत्रण (Monitoring and Controlling) चरण का प्राथमिक उद्देश्य क्या है? (अ) प्रोजेक्ट को बंद करना और दस्तावेज़ों को संग्रहीत करना (ब) प्रोजेक्ट के दायरे और उद्देश्यों को परिभाषित करना (स) प्रोजेक्ट की प्रगति को ट्रैक करना, विचलनों का प्रबंधन करना और गुणवत्ता सुनिश्चित करना (द) प्रोजेक्ट टीम का विकास करना और संचार को सुविधाजनक बनाना

rial Not be Published

3. प्रोजेक्ट प्रबंधन में निम्नलिखित में से किसे एक प्रमुख सर्वोत्तम अभ्यास (best practice) माना जाता है? (अ) संचार को केवल औपचारिक रिपोर्टों तक सीमित रखना (ब) प्रोजेक्ट पूर्ण होने तक हितधारकों की भागीदारी से बचना (स) प्रोजेक्ट से संबंधित हर चीज़ का दस्तावेजीकरण करना (द) प्रोजेक्ट में असफलता की संभावना को अनदेखा करना
4. AI परियोजनाओं में डेटा एकत्र करने की निम्नलिखित में से कौन सी एक विशिष्ट विधि *नहीं* है? (अ) सर्वेक्षण और प्रश्नावली (ब) अवलोकन संबंधी अध्ययन (स) मॉडल प्रशिक्षण (द) वेब स्क्रैपिंग
5. AI प्रोजेक्ट चक्र में डेटा अन्वेषण (data exploration) का मुख्य उद्देश्य क्या है? (अ) समस्या विवरण को परिभाषित करना (ब) AI मॉडल को प्रशिक्षित करना (स) डेटा विशेषताओं को समझना, पैटर्न की पहचान करना और विसंगतियों का पता लगाना (द) AI मॉडल को तैनात करना
6. AI के लिए डेटा संग्रह में यह निम्नलिखित में से कौन सी एक प्रमुख चुनौती है? (अ) प्रक्रिया को स्वचालित करना (ब) डेटा गुणवत्ता सुनिश्चित करना और पूर्वाग्रह (bias) से बचना (स) पर्याप्त डेटा ढूंढना (द) अन्य AI उपकरणों के साथ एकीकृत करना
7. "फीचर इंजीनियरिंग (feature engineering)" का क्या उद्देश्य है? (अ) डेटा संग्रह को स्वचालित करना (ब) मॉडल के प्रदर्शन को बढ़ाने के लिए नई या संशोधित विशेषताएँ बनाना (स) चार्ट का उपयोग करके डेटा पैटर्न को देखना (द) मॉडल की सटीकता का मूल्यांकन करना
8. AI परियोजनाओं में डेटा विज़ुअलाइज़ेशन के लिए आमतौर पर किस उपकरण का उपयोग किया जाता है? (अ) TensorFlow (ब) Tableau (स) Apache Spark (द) Scikit-learn
9. निम्नलिखित में से कौन सा डेटा प्रीप्रोसेसिंग (data preprocessing) का प्राथमिक लक्ष्य है? (अ) बेहतर समझ के लिए डेटा को देखना (ब) सही मशीन लर्निंग एल्गोरिदम का चयन करना (स) कच्चे डेटा को मशीन लर्निंग एल्गोरिदम के लिए उपयुक्त प्रारूप में तैयार करना (द) प्रशिक्षित मॉडल को तैनात करना
10. प्रोजेक्ट समीक्षा (project review) का प्राथमिक उद्देश्य क्या है? (अ) प्रोजेक्ट के लिए संसाधन आवंटित करना (ब) प्रोजेक्ट की सफलता का मूल्यांकन करना (स) प्रोजेक्ट प्रबंधक के प्रदर्शन का निर्धारण करना (द) नई प्रोजेक्ट विचार उत्पन्न करना

ब. रिक्त स्थान भरें

1. _____ वेबसाइटों से डेटा स्वचालित रूप से निकालने की एक तकनीक है।
2. _____ सूचना का चित्रमय प्रतिनिधित्व है, जो चार्ट, ग्राफ और मानचित्रों का उपयोग करता है।
3. _____ ढांचा सुनिश्चित और प्राप्त करने योग्य लक्ष्य बनाने के लिए एक लोकप्रिय दृष्टिकोण है।
4. डेटा में त्रुटियों, असंगतियों और लापता मानों की पहचान करने और उन्हें ठीक करने की प्रक्रिया को _____ कहा जाता है।
5. किसी मॉडल के प्रदर्शन को अनुकूलित करने के लिए उसकी सेटिंग्स को समायोजित करने की प्रक्रिया को _____ कहा जाता है।
6. एक प्रोजेक्ट समीक्षा अपने जीवनचक्र में एक विशिष्ट बिंदु पर प्रोजेक्ट की _____, _____ और _____ का एक औपचारिक मूल्यांकन है।
7. फीचर इंजीनियरिंग _____ बढ़ाती है।
8. प्रदर्शन को अनुकूलित करने के लिए _____ को ठीक करना (fine-tuning)।

9. पहचानी गई समस्या को हल करने के लिए प्रशिक्षित मॉडल को _____ पर तैनात करना।
10. AI मॉडल को _____ का उपयोग करके प्रशिक्षित किया जाता है।

स. सत्य या असत्य बताएँ

1. डेटा संग्रह केवल बड़े पैमाने की AI परियोजनाओं के लिए प्रासंगिक है।
2. डेटा अन्वेषण का एक महत्वपूर्ण हिस्सा डेटा विज़ुअलाइज़ेशन नहीं है।
3. API का उपयोग केवल वास्तविक समय का डेटा एकत्र करने के लिए किया जाता है।
4. प्रीप्रोसेसिंग के बाद भी खराब डेटा गुणवत्ता AI मॉडल के प्रदर्शन पर नकारात्मक प्रभाव डाल सकती है।
5. डेटा प्रीप्रोसेसिंग AI प्रोजेक्ट चक्र में एक वैकल्पिक चरण है।
6. मॉडल पैरामीटर प्रशिक्षण से पहले डेटा वैज्ञानिक द्वारा मैन्युअल रूप से सेट किए जाते हैं, जबकि हाइपरपैरामीटर प्रशिक्षण के दौरान सीखे जाते हैं।
7. प्रोजेक्ट समीक्षा एक प्रक्रिया है जो यह जांचने के लिए है कि लोग काम कर रहे हैं या नहीं?
8. प्रभावी प्रोजेक्ट समीक्षाएँ संगठनों को अपनी प्रोजेक्ट प्रबंधन रणनीतियों और प्रथाओं को परिष्कृत करने में मदद करके निरंतर सुधार में योगदान करती हैं।
9. प्रोजेक्ट दस्तावेज़ीकरण का प्रारूप प्रोजेक्ट की प्रकृति पर निर्भर करता है।
10. प्रोजेक्ट के मूल्यांकन का अंतिम स्कोर प्रोजेक्ट के समग्र प्रभाव पर आधारित होता है।

द. लघु उत्तरीय प्रश्न

1. प्रोजेक्ट कार्य (Project Work) विद्यार्थियों के व्यक्तिगत विकास में कैसे सहायक होता है?

2. AI प्रोजेक्ट के विकास की प्रक्रिया में "Problem Definition and Goal Setting" चरण का क्या महत्व है?

3. डेटा संग्रह और अन्वेषण (Data Collection and Exploration) में किन-किन स्रोतों से डेटा एकत्र किया जा सकता है?

4. मॉडल प्रशिक्षण और परीक्षण (Model Training and Testing) के दौरान किन मेट्रिक्स का उपयोग किया जाता है?

5. प्रोजेक्ट रिव्यू (Project Review) का उद्देश्य क्या होता है और यह क्यों आवश्यक है?

सत्र 3. सैम्पल प्रोजेक्ट

(Sample Project)

AI प्रोजेक्ट की एक नमूना प्रोजेक्ट रिपोर्ट नीचे प्रस्तुत की गई है।

1. शीर्षक पृष्ठ (Title Page)

प्रोजेक्ट का शीर्षक: ग्राहक समीक्षाओं पर भावना विश्लेषण (Sentiment Analysis on Customer Reviews)

लेखक / टीम के सदस्य: X, Y

संस्थान / संगठन: XYZ विश्वविद्यालय

प्रस्तुतिकरण की तिथि: जून 28, 2025

2. सारांश (Abstract)

इस प्रोजेक्ट का उद्देश्य ग्राहक समीक्षाओं को सकारात्मक, नकारात्मक या तटस्थ के रूप में वर्गीकृत करने के लिए एक भावना विश्लेषण मॉडल विकसित करना है। प्राकृतिक भाषा प्रसंस्करण (NLP) तकनीकों और मशीन लर्निंग का उपयोग करके, हम सार्थक अंतर्दृष्टि निकालने के लिए पाठ डेटा को पूर्व-प्रक्रिया और विश्लेषण करते हैं। प्रोजेक्ट भावना वर्गीकरण के लिए एक लॉजिस्टिक रिग्रेशन मॉडल और डीप लर्निंग तकनीकों को नियोजित करती है, जो 85% की सटीकता प्राप्त करती है। परिणाम व्यवसायों को ग्राहकों की राय समझने और अपनी सेवाओं में सुधार करने में मदद कर सकते हैं।

3. परिचय (Introduction)

पृष्ठभूमि जानकारी (Background Information)

ग्राहक समीक्षाएँ व्यवसायों के लिए प्रतिक्रिया का एक मूल्यवान स्रोत हैं। इन समीक्षाओं का मैन्युअल रूप से विश्लेषण करना समय लेने वाला और अक्षम है। भावना विश्लेषण, एक NLP तकनीक, राय को भावना वर्गों में वर्गीकृत करके इस प्रक्रिया को स्वचालित करती है।

समस्या विवरण (Problem Statement)

मौजूदा भावना विश्लेषण प्रणालियाँ अक्सर प्रासंगिक समझ और व्यंग्य के साथ संघर्ष करती हैं, जिससे गलत वर्गीकरण होता है। हमारी प्रोजेक्ट का उद्देश्य मशीन लर्निंग का उपयोग करके भावना वर्गीकरण सटीकता में सुधार करना है।

उद्देश्य (Objectives)

- भावना वर्गीकरण के लिए एक मशीन लर्निंग मॉडल विकसित करना।
- प्रीप्रोसेसिंग और मॉडल अनुकूलन के माध्यम से सटीकता में सुधार करना।
- भावना विश्लेषण के लिए एक उपयोगकर्ता-अनुकूल इंटरफ़ेस प्रदान करना।

4. साहित्य समीक्षा (Literature Review)

मौजूदा शोध (Existing Research)

भावना विश्लेषण के लिए कई दृष्टिकोण मौजूद हैं, जिनमें लेक्सिकॉन-आधारित विधियाँ और नाइव बेयस, सपोर्ट वेक्टर मशीन और डीप लर्निंग मॉडल जैसे मशीन लर्निंग मॉडल शामिल हैं।

दृष्टिकोणों की तुलना (Comparison of Approaches)

- **लेक्सिकॉन-आधारित विधियाँ**: पूर्वनिर्धारित शब्द सूचियों पर निर्भर करती हैं लेकिन उनमें संदर्भ समझ का अभाव होता है।
- **मशीन लर्निंग मॉडल**: लेबल किए गए डेटा की आवश्यकता होती है लेकिन अच्छी तरह से सामान्यीकरण करते हैं।
- **डीप लर्निंग मॉडल**: जटिल पैटर्न को पकड़ते हैं लेकिन बड़े डेटासेट की आवश्यकता होती है।

औचित्य (Justification)

हमने बेसलाइन प्रदर्शन के लिए लॉजिस्टिक रिग्रेशन को चुना और बेहतर सटीकता के लिए एक डीप लर्निंग मॉडल को चुना, क्योंकि इसकी जटिल भाषाई पैटर्न को समझने की क्षमता है।

5. कार्यप्रणाली (Methodology)

डेटा संग्रह (Data Collection)

- डेटासेट: IMDB और अमेज़न ग्राहक समीक्षाएँ।
- प्रीप्रोसेसिंग: टोकनाइज़ेशन, स्टॉपवर्ड हटाना, स्टेमिंग और TF-IDF का उपयोग करके वेक्टराइज़ेशन।

मशीन लर्निंग मॉडल (Machine Learning Models)

- बेसलाइन के लिए लॉजिस्टिक रिग्रेशन (Logistic Regression)।
- डीप लर्निंग के लिए LSTM (लॉन्ग शॉर्ट-टर्म मेमोरी) न्यूरल नेटवर्क।

उपकरण और फ्रेमवर्क (Tools and Frameworks)

- पायथन, Scikit-learn, TensorFlow, NLTK, Pandas, Matplotlib।

प्रशिक्षण और मूल्यांकन (Training and Evaluation)

- डेटासेट को प्रशिक्षण (80%) और परीक्षण (20%) में विभाजित करें।
- मूल्यांकन मेट्रिक्स: सटीकता, परिशुद्धता, पुनर्स्मरण, F1-स्कोर।

6. कार्यान्वयन चरण (Implementation Steps)

1. **डेटा प्रीप्रोसेसिंग:** पाठ डेटा को साफ करना और बदलना।
2. **फीचर निष्कर्षण:** पाठ को संख्यात्मक प्रतिनिधित्व में बदलना।
3. **मॉडल प्रशिक्षण:** लॉजिस्टिक रिग्रेशन और LSTM दोनों मॉडलों को प्रशिक्षित करना।
4. **मॉडल मूल्यांकन:** मेट्रिक्स का उपयोग करके प्रदर्शन को मापना।

प्रणाली वास्तुकला (System Architecture)

- **इनपुट:** ग्राहक समीक्षाएँ।
- **प्रसंस्करण:** NLP प्रीप्रोसेसिंग और मॉडल भविष्यवाणी।
- **आउटपुट:** भावना वर्गीकरण (सकारात्मक/नकारात्मक/तटस्थ)।

कोड कार्यान्वयन (Code Implementation)

```
import pandas as pd
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, SpatialDropout1D
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

# Load dataset
df = pd.read_csv('customer_reviews.csv')

# Preprocessing
def clean_text(text):
    text = re.sub(r'[^a-zA-Z]', ' ', text)
    text = text.lower()
    text = text.split()
    text = [word for word in text if word not in stopwords.words('english')]
    return ' '.join(text)
df['cleaned_reviews'] = df['review'].apply(clean_text)

# TF-IDF Vectorization
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(df['cleaned_reviews']).toarray()
y = df['sentiment']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Logistic Regression Model
lr_model = LogisticRegression()
lr_model.fit(X_train, y_train)

# Evaluate Logistic Regression
```

© PSSCIVE Draft Study Material Not be Published

```
y_pred = lr_model.predict(X_test)
print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))

# Deep Learning Model with LSTM
max_words = 5000
max_len = 200
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(df['cleaned_reviews'])
X_seq = tokenizer.texts_to_sequences(df['cleaned_reviews'])
X_pad = pad_sequences(X_seq, maxlen=max_len)

X_train_dl, X_test_dl, y_train_dl, y_test_dl = train_test_split(X_pad, y, test_size=0.2,
random_state=42)

model = Sequential()
model.add(Embedding(max_words, 128, input_length=max_len))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(3, activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])

model.fit(X_train_dl, y_train_dl, epochs=5, batch_size=64, validation_data=(X_test_dl,
y_test_dl))
```

7. परिणाम और विश्लेषण (Results and Analysis)

प्रदर्शन मेट्रिक्स (Performance Metrics)

मॉडल (Model)	**सटीकता (Accuracy)**	**परिशुद्धता (Precision)**	**पुनर्स्मरण (Recall)**	**F1-स्कोर (F1-score)**
लॉजिस्टिक रिग्रेशन	78%	76%	74%	75%

PSSCIVE Draft Study Material Not be Published

LSTM	85%	83%	82%	83%

विज़ुअलाइज़ेशन (Visualizations)

- दोनों मॉडलों के लिए भ्रम मैट्रिक्स (Confusion matrices)।
- LSTM मॉडल के लिए सटीकता और हानि वक्र।

8. चर्चा (Discussion)

परिणामों की व्याख्या (Interpretation of Results)

- LSTM मॉडल ने लॉजिस्टिक रिग्रेशन से बेहतर प्रदर्शन किया।
- चुनौतियों में व्यंग्य और अस्पष्ट भावनाओं को संभालना शामिल था।

संभावित सुधार (Potential Improvements)

- बेहतर प्रासंगिक समझ के लिए BERT जैसे ट्रांसफॉर्मर-आधारित मॉडल को शामिल करना।
- कई डोमेन को शामिल करने के लिए डेटासेट का विस्तार करना।

9. निष्कर्ष (Conclusion)

इस प्रोजेक्ट ने मशीन लर्निंग और डीप लर्निंग का उपयोग करके भावना विश्लेषण को सफलतापूर्वक लागू किया। LSTM मॉडल ने 85% की सटीकता हासिल की, जो ग्राहक प्रतिक्रिया विश्लेषण में वास्तविक दुनिया के अनुप्रयोगों के लिए इसकी क्षमता को प्रदर्शित करता है।

10. संदर्भ (References)

- Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing.
- Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis.
- भावना विश्लेषण तकनीकों पर शोध पत्र।

11. परिशिष्ट (Appendices)

- डेटा प्रीप्रोसेसिंग और मॉडल प्रशिक्षण के लिए नमूना कोड।
- अतिरिक्त ग्राफ और विज़ुअलाइज़ेशन।
- डीप लर्निंग मॉडल के लिए हाइपरपैरामीटर ट्यूनिंग विवरण।

शब्दावली

- **एग्रीगेट फ़ंक्शन्स (Aggregate Functions):** ऐसे फ़ंक्शन जो एरे के तत्वों के समूह से एक ही परिणाम निकालते हैं (जैसे sum, mean)।
- **एरे (Array):** एक ही प्रकार के तत्वों का संग्रह, जो मेमोरी में एक साथ संग्रहीत होते हैं और सूचकांक (इंडेक्स) द्वारा पहुँचे जा सकते हैं।
- **अक्ष (Axis):** एक एरे का एक आयाम। 2D एरे में, पंक्तियाँ अक्ष 0 हैं और स्तंभ अक्ष 1 हैं।
- **प्रसारण (Broadcasting):** NumPy की सुविधा, जिससे अलग-अलग आकार के एरे एक साथ ऑपरेशन कर सकते हैं।
- **संयोजन (Concatenation):** दो या अधिक एरे को जोड़कर एक बनाना।
- **प्रतिलिपि (Copy):** एक नया, स्वतंत्र एरे जिसमें डेटा की अलग प्रति होती है।
- **डेटा प्रकार (dtype):** एरे में संग्रहीत डेटा का प्रकार (जैसे, int32, float64)।
- **आयाम (Dimension):** एक एरे में अक्षों की संख्या (1D, 2D, 3D)।
- **तत्व-वार संक्रिया (Element-wise Operation):** एरे के प्रत्येक संबंधित तत्वों पर किया जाने वाला ऑपरेशन।
- **समतलन (Flattening):** एक बहु-आयामी एरे को एक-आयामी एरे में बदलना।
- **अनुक्रमण या इंडेक्सिंग (Indexing):** किसी एरे तत्व को उसकी स्थिति संख्या द्वारा एक्सेस करना।
- **ndarray:** NumPy में मूल n-आयामी एरे ऑब्जेक्ट।
- **पुनः आकार देना (Reshape):** डेटा को बदले बिना किसी एरे का आकार बदलना।
- **आकार बदलना (Resize):** किसी एरे का आकार बदलना, जिससे तत्व जोड़े या हटाए जा सकते हैं।
- **स्लाइसिंग (Slicing):** सूचकांकों की एक श्रृंखला का उपयोग करके एरे का एक भाग निकालना।
- **विभाजन (Splitting):** एक एरे को कई उप-एरे में विभाजित करना।
- **ट्रांसपोज़ (Transpose):** 2D एरे को पलटना, पंक्तियों को स्तंभों में और स्तंभों को पंक्तियों में बदलना।
- **वेक्टराइजेशन (Vectorization):** धीमे पायथन लूप का उपयोग किए बिना, पूरे एरे पर एक साथ संक्रियाएँ करना।
- **दृश्य (View):** एक नया एरे ऑब्जेक्ट जो मूल एरे के साथ डेटा साझा करता है।
- **एग्रीगेशन (Aggregation):** एक से अधिक मानों को एक में संयोजित करना, जैसे किसी कक्षा के औसत अंक ज्ञात करना।
- **गुण (Attribute):** किसी वस्तु का गुण या विशेषता, जैसे .shape डेटाफ्रेम का आकार बताता है।
- **बॉक्स प्लॉट (Box Plot):** एक चार्ट जो न्यूनतम, चतुर्थक, माध्यिका और अधिकतम मानों का उपयोग करके डेटा के प्रसार को दर्शाता है।
- **CSV (अल्पविराम से अलग किए गए मान):** एक सरल फ़ाइल प्रारूप जहाँ डेटा को अल्पविराम द्वारा अलग किए गए पाठ के रूप में संग्रहीत किया जाता है, जैसे बिना फ़ॉर्मेटिंग के एक्सेल शीट।
- **डेटा सफाई (Data Cleaning):** गलत, अधूरे या डुप्लिकेट डेटा को सही करने या हटाने की प्रक्रिया।
- **डेटाफ्रेम (DataFrame):** पंक्तियों और स्तंभों वाली पांडस में 2D तालिका, जैसे एक स्प्रेडशीट।
- **फ़िल्टरिंग (Filtering):** केवल उन डेटा की पंक्तियों या स्तंभों का चयन करना जो कुछ शर्तों को पूरा करते हैं।

- **ग्रुपबाय (GroupBy):** एक पांडस फ़ंक्शन जो डेटा की पंक्तियों को किसी स्तंभ के आधार पर समूहीकृत करता है, जैसे क्षेत्र के अनुसार बिक्री।
- **हिस्टोग्राम (Histogram):** एक ग्राफ जो निश्चित श्रेणियों (बिन्स) के भीतर डेटा की आवृत्ति को दर्शाता है।
- **सूचकांक (Index):** पांडस में सीरीज या डेटाफ्रेम में पंक्तियों या स्तंभों की पहचान करने के लिए उपयोग किए जाने वाले लेबल या स्थितियाँ।
- **JSON (जावास्क्रिप्ट ऑब्जेक्ट नोटेशन):** एक फ़ाइल प्रारूप जिसका उपयोग डेटा को कुंजी-मान युग्मों में संग्रहीत करने और विनिमय करने के लिए किया जाता है, जिसका उपयोग अक्सर वेब अनुप्रयोगों में किया जाता है।
- **लेजेंड (Legend):** एक चार्ट में एक बॉक्स जो विभिन्न रंगों, प्रतीकों या रेखाओं के अर्थ को समझाता है।
- **Matplotlib:** एक पायथन लाइब्रेरी जिसका उपयोग ग्राफ और विज़ुअलाइज़ेशन बनाने के लिए किया जाता है।
- **लापता मान (Missing Values) (NaN):** डेटा जो किसी डेटासेट में उपलब्ध नहीं है या खाली है।
- **NumPy:** एरे का उपयोग करके तीव्र संख्यात्मक संक्रियाओं के लिए एक पायथन लाइब्रेरी।
- **बाहरी मान (Outlier):** एक डेटा बिंदु जो अन्य मानों से बहुत भिन्न होता है, जैसे एक विद्यार्थी का 100 अंक प्राप्त करना जब अधिकांश ने 40-60 अंक प्राप्त किए हों।
- **पैनल डेटा (Panel Data):** बहुआयामी संरचित डेटा जिसका उपयोग अक्सर अर्थशास्त्र और सांख्यिकी में किया जाता है।
- **पांडस (Pandas):** तालिकाओं (सीरीज और डेटाफ्रेम) में डेटा को संभालने और विश्लेषण करने के लिए एक पायथन लाइब्रेरी।
- **पाई चार्ट (Pie Chart):** एक वृत्ताकार चार्ट जो अनुपात दिखाने के लिए खंडों में विभाजित होता है।
- **पिवट टेबल (Pivot Table):** डेटा को शीघ्रता से सारांशित और पुनर्व्यवस्थित करने का एक उपकरण, जैसे प्रति उत्पाद श्रेणी कुल बिक्री।
- **प्लॉट (Plot):** डेटा को दृष्टिगत रूप से प्रदर्शित करने के लिए उपयोग किया जाने वाला एक ग्राफ या चार्ट।
- **पाइप्लॉट (Pyplot) (plt):** प्लॉटिंग ग्राफ के लिए Matplotlib का एक उप-मॉड्यूल।
- **चतुर्थक (Quartile):** डेटा को चार बराबर भागों में विभाजित करना; Q1 (25%), Q2/माध्यिका (50%), Q3 (75%)।
- **स्कैटर प्लॉट (Scatter Plot):** संबंधों का अध्ययन करने के लिए X और Y अक्ष पर डेटा बिंदु दिखाने वाला एक ग्राफ।
- **सीरीज (Series):** पांडस में एक-आयामी लेबल वाला एरे, जैसे अंकों का एक स्तंभ।
- **SQL (Structured Query Language) (संरचित क्वेरी भाषा):** एक भाषा जिसका उपयोग डेटाबेस में संग्रहीत डेटा को प्रबंधित और क्वेरी करने के लिए किया जाता है।
- **सांख्यिकीय फ़ंक्शन (Statistical Functions):** डेटा का विश्लेषण करने के लिए उपयोग किए जाने वाले माध्य, माध्यिका, योग और मानक विचलन जैसे फ़ंक्शन।
- **सारणीबद्ध डेटा (Tabular Data):** पंक्तियों और स्तंभों में व्यवस्थित डेटा, जैसे Excel में।
- **विज़ुअलाइज़ेशन (Visualization):** आसान समझ के लिए डेटा को चित्रमय रूप (चार्ट/ग्राफ) में प्रदर्शित करना।

उत्तर कुंजी

मॉड्यूल 1. पायथन प्रोग्रामिंग

सत्र 1. कंट्रोल स्ट्रक्चर्स

अ. बहुविकल्पीय प्रश्न

1. (द), 2. (अ), 3. (अ), 4. (अ), 5. (ब), 6. (अ), 7. (अ), 8. (अ), 9. (अ), 10. (अ), 11. (ब), 12. (स), 13. (ब), 14. (अ), 15. (द)

ब. रिक्त स्थान भरें

1. कंट्रोल फ्लो (Control Flow), 2. इंडेंटेशन (Indentation), 3. IndentationError, 4. Else, 5. समाप्त (terminate), 6. छोड़ (skip), 7. नेस्टेड (nested), 8. संख्यात्मक (numeric) और स्टार (star), 9. बहु-आयामी (multidimensional), 10. a == b

स. सत्य या असत्य

1. सत्य 2. असत्य, 3. सत्य, 4. सत्य, 5. सत्य, 6. सत्य, 7. सत्य, 8. सत्य, 9. असत्य, 10. सत्य

सत्र 2. पायथन में फ़ंक्शन

अ. बहुविकल्पीय प्रश्न

1. (द), 2. (स), 3. (ब), 4. (अ), 5. (अ), 6. (अ), 7. (अ), 8. (द), 9. (अ), 10. (द)

ब. रिक्त स्थान भरें

1. मॉड्यूलर, 2. यूज़र-डिफाइंड, 3. आर्गुमेंट, 4. return, 5. लाइब्रेरी, 6. .py, 7. import, 8. डॉट (.), 9. random, 10. from

स. सत्य या असत्य

1. सत्य, 2. असत्य, 3. असत्य, 4. सत्य, 5. सत्य, 6. असत्य, 7. सत्य, 8. असत्य, 9. सत्य, 10. असत्य

मॉड्यूल 2. डेटा विज्ञान

सत्र 1. NumPy का परिचय

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (अ), 3. (ब), 4. (ब), 5. (ब), 6. (अ), 7. (स), 8. (ब)

ब. रिक्त स्थान भरें

1. सन्निहित (contiguous/continuous), 2. ndim, 3. reshape(), 4. np.ones(), 5. dtype

स. सत्य या असत्य

1. असत्य, 2. सत्य, 3. सत्य, 4. सत्य, 5. असत्य

सत्र 2. NumPy का उपयोग करके एरे का संचालन

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (अ), 3. (ब), 4. (ब), 5. (अ), 6. (ब), 7. (ब), 8. (अ)

ब. रिक्त स्थान भरें

1. संयोजित (combining), 2. np.vstack(), 3. np.hstack(), 4. np.reshape(), 5. np.squeeze()

स. सत्य या असत्य

1. सत्य, 2. असत्य, 3. सत्य, 4. असत्य, 5. सत्य

सत्र 3. NumPy का उपयोग करके एरे संगणना

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (ब), 3. (ब), 4. (स), 5. (अ), 6. (ब), 7. (ब), 8. (ब)

ब. रिक्त स्थान भरें

1. तत्व-वार (element-wise), 2. np.mean(), 3. %, 4. np.cumsum(), 5. np.dot()

स. सत्य या असत्य

1. असत्य, 2. सत्य, 3. सत्य, 4. सत्य, 5. सत्य

मॉड्यूल 3. डेटा विश्लेषण

सत्र 1: पांडस का परिचय

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (स), 3. (ब), 4. (ब), 5. (ब), 6. (स), 7. (स), 8. (स), 9. (स), 10. (ब)

ब. रिक्त स्थान भरें

1. प्रबंधन, DataFrame, एक, दो; 2. दो; 3. pd; 4. NumPy; 5. head; 6. Not; 7. 0; 8. तालिकात्मक

स. सत्य या असत्य

1. (असत्य), 2. (सत्य), 3. (असत्य), 4. (सत्य), 5. (सत्य), 6. (असत्य), 7. (असत्य), 8. (सत्य), 9. (असत्य), 10. (सत्य)

सत्र 2: पांडस सीरीज

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (d), 3. (स), 4. (d), 5. (स), 6. (d), 7. (अ), 8. (ब), 9. (स), 10. (ब), 11. (ब), 12. (ब)

ब. रिक्त स्थान भरें

1. एक (One), 2. सूचकांक (Index), 3. पूर्णांक (Integer), 4. कुंजियाँ (Keys), 5. सूचकांक (Index), 6. values, 7. dtype, 8. NaN, 9. size, 10. तत्व (Element), 11. hasnans

स. सत्य या असत्य

1. (सत्य), 2. (असत्य), 3. (सत्य), 4. (असत्य), 5. (असत्य), 6. (सत्य), 7. (सत्य), 8. (असत्य), 9. (सत्य), 10. (सत्य), 11. (सत्य)

सत्र 3: पांडस डेटाफ्रेम

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (d), 3. (ब), 4. (स), 5. (स), 6. (ब), 7. (स), 8. (स), 9. (ब), 10. (स)

ब. रिक्त स्थान भरें

1. स्तंभ (Columns), 2. pd, 3. स्तंभ (Column), 4. shape, 5. head(), 6. 'Price', 7. columns, 8. सूचकांक (index), सूचकांक (index), 9. index, 10. dtypes

स. सत्य या असत्य

1. (असत्य), 2. (सत्य), 3. (असत्य), 4. (सत्य), 5. (सत्य), 6. (असत्य), 7. (असत्य), 8. (सत्य), 9. (सत्य), 10. (असत्य)

सत्र 4: Matplotlib का उपयोग करके डेटा विज़ुअलाइज़ेशन

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (स), 3. (ब), 4. (अ), 5. (स), 6. (d), 7. (d), 8. (स), 9. (d), 10. (स)

ब. रिक्त स्थान भरें

1. इंटरैक्टिव (Interactive), 2. pyplot, 3. plot(), 4. show(), 5. title(), 6. ylabel(), 7. savefig(), 8. legend(), 9. scatter(), 10. grid()

स. सत्य या असत्य

1. (असत्य), 2. (सत्य), 3. (असत्य), 4. (सत्य), 5. (असत्य), 6. (सत्य), 7. (असत्य), 8. (सत्य), 9. (असत्य), 10. (सत्य)

मॉड्यूल 4. न्यूरल नेटवर्क

सत्र 1: न्यूरल नेटवर्क का परिचय

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (स), 3. (ब), 4. (ब), 5. (स)

ब. रिक्त स्थान भरें

1. मानव मस्तिष्क (human brain), 2. इनपुट (Input), 3. भार (Weights), 4. बैकप्रोपगेशन (Backpropagation), 5. आउटपुट (output)

स. सत्य या असत्य

1. (असत्य), 2. (सत्य), 3. (सत्य), 4. (असत्य), 5. (सत्य)

सत्र 2: न्यूरॉन के अंदर: सक्रियण फ़ंक्शन

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (ब), 3. (स), 4. (ब), 5. (स)

ब. रिक्त स्थान भरें

1. छिपी हुई परत (hidden layer), 2. ReLU, 3. सिग्मॉइड (Sigmoid), 4. कनवोल्यूशनल (Convolutional), 5. बायस (bias)

स. सत्य या असत्य

1. (सत्य), 2. (असत्य), 3. (सत्य), 4. (असत्य), 5. (असत्य)

सत्र 3: क्रियाशील न्यूरल नेटवर्क: वास्तविक दुनिया के अनुप्रयोग

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (d), 3. (स), 4. (ब), 5. (ब)

ब. रिक्त स्थान भरें

1. वस्तु पहचान (object detection), 2. छवि विभाजन (Image segmentation), 3. मशीन अनुवाद (machine translation), 4. भाषण-से-पाठ (speech-to-text), 5. वर्गीकरण (classification)

स. सत्य या असत्य

1. (असत्य), 2. (सत्य), 3. (सत्य), 4. (असत्य), 5. (असत्य)

सत्र 4: पायथन लाइब्रेरी के साथ न्यूरल नेटवर्क का निर्माण

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (ब), 3. (ब), 4. (d), 5. (स)

ब. रिक्त स्थान भरें

1. NumPy, 2. अनुक्रमिक (Sequential), 3. सामान्यीकरण (Normalization), 4. सटीकता (Accuracy), 5. ऑप्टिमाइज़र (optimizer)

स. सत्य या असत्य

1. (असत्य), 2. (असत्य), 3. (असत्य), 4. (सत्य), 5. (असत्य)

सत्र 5: डेटा तैयार करना और अपने मॉडल को प्रशिक्षित करना

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (स), 3. (ब), 4. (स), 5. (ब)

ब. रिक्त स्थान भरें

1. सामान्यीकरण (normalization), 2. प्रशिक्षण (training), परीक्षण (testing), 3. भ्रम मैट्रिक्स (confusion matrix), 4. हानि (Loss), 5. ReLU

स. सत्य या असत्य

1. (असत्य), 2. (असत्य), 3. (सत्य), 4. (असत्य), 5. (सत्य)

सत्र 6: वर्गीकरण मॉडल का निर्माण

अ. बहुविकल्पीय प्रश्न

1. (ब), 2. (ब), 3. (ब), 4. (स), 5. (ब)

ब. रिक्त स्थान भरें

1. द्विआधारी (binary), 2. पुनर्स्मरण (Recall), 3. ड्रॉपआउट (Dropout), 4. अति-अनुकूलित (Overfitting), 5. binary_crossentropy

स. सत्य या असत्य

1. (असत्य), 2. (सत्य), 3. (असत्य), 4. (असत्य), 5. (सत्य)

मॉड्यूल 5. एआई प्रोजेक्ट

सत्र 1. AI प्रोजेक्ट चक्र

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (स), 3. (ब), 4. (स), 5. (d), 6. (स), 7. (ब), 8. (स), 9. (ब), 10. (स)

ब. रिक्त स्थान भरें

1. समस्या क्षेत्र निर्धारण (Problem Scoping), 2. समस्या क्षेत्र निर्धारण (Problem Scoping), 3. 4Ws समस्या कैनवास (4Ws Problem Canvas), 4. डेटा अधिग्रहण (Data Acquisition), 5. डेटा सफाई (Data Cleaning), 6. डेटा अन्वेषण (Data Exploration), 7. मॉडल प्रशिक्षण (Model Training), 8. मॉडल मूल्यांकन (Model Evaluation), 9. तैनाती (Deployment), 10. निगरानी (Monitoring), 11. पुनरावृत्तीय (iterative)

स. सत्य या असत्य

1. असत्य, 2. सत्य, 3. असत्य, 4. असत्य, 5. असत्य, 6. असत्य, 7. सत्य, 8. असत्य, 9. असत्य, 10. असत्य

सत्र 2. प्रोजेक्ट कार्य

अ. बहुविकल्पीय प्रश्न

1. (स), 2. (स), 3. (स), 4. (स), 5. (स), 6. (ब), 7. (ब), 8. (ब), 9. (स), 10. (ब)

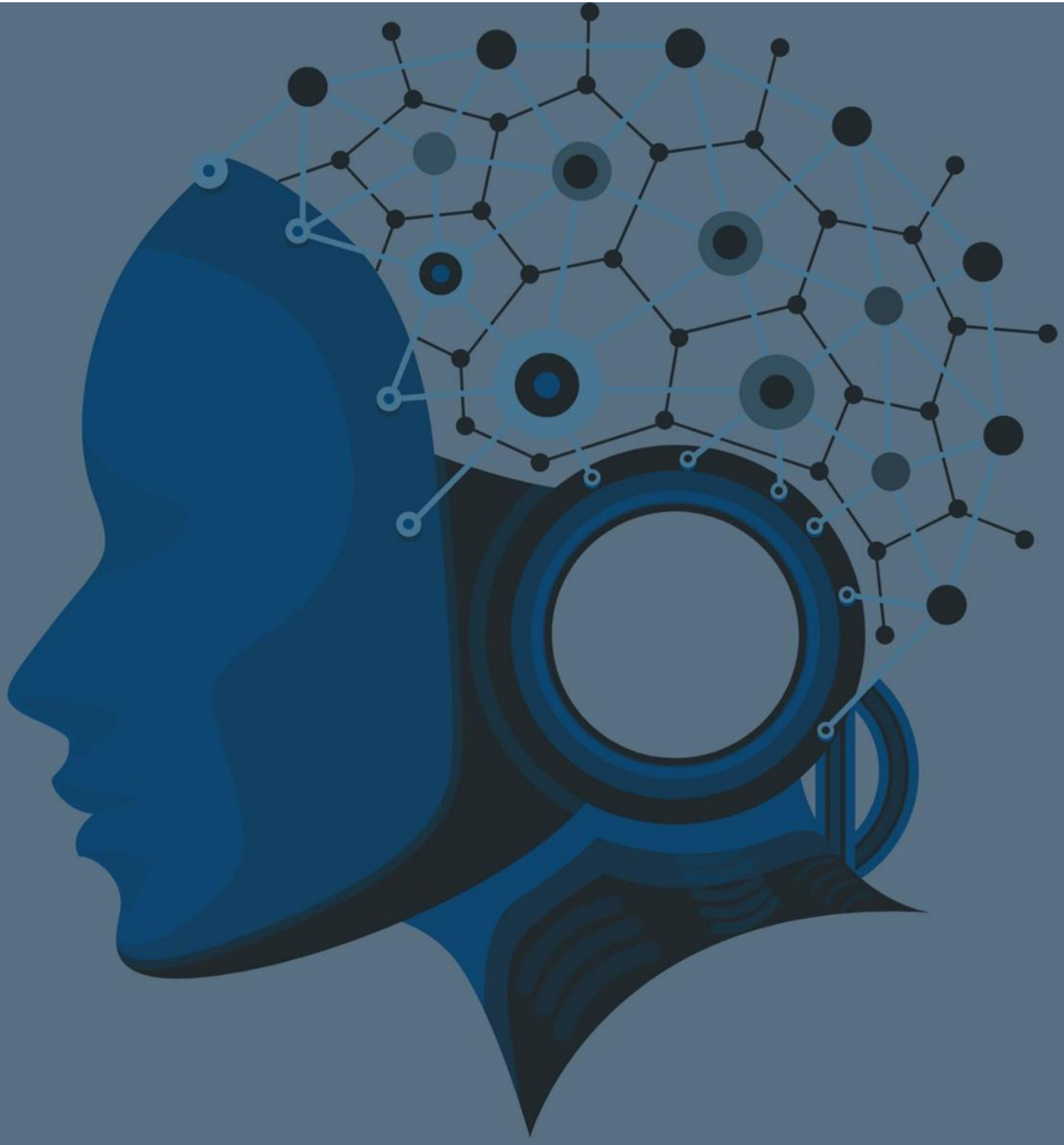
ब. रिक्त स्थान भरें

1. वेब स्क्रैपिंग (Web scraping), 2. डेटा विज़ुअलाइज़ेशन (Data visualization), 3. SMART, 4. डेटा सफाई (Data Cleaning), 5. हाइपरपैरामीटर ट्यूनिंग (Hyperparameter Tuning), 6. प्रगति (progress), प्रदर्शन (performance), परिणामों (outcomes), 7. मॉडल प्रदर्शन (model performance), 8. हाइपरपैरामीटर्स (hyperparameters), 9. उत्पादन वातावरण (production environment), 10. प्रशिक्षण डेटा (training data)

स. सत्य या असत्य

1. असत्य, 2. असत्य, 3. असत्य, 4. सत्य, 5. असत्य, 6. असत्य, 7. असत्य, 8. सत्य, 9. असत्य, 10. असत्य

© PSSCIVE Draft Study Material Not be Published



पंडित सुंदरलाल शर्मा केंद्रीय व्यावसायिक शिक्षा संस्थान
(भारत सरकार के शिक्षा मंत्रालय के अधीन रा.शै.अ.प्र.प. की घटक इकाई)
श्यामला हिल्स, भोपाल– 462002, मध्य प्रदेश, भारत
http://www.psscive.ac.in