

## ASSIGNMENT - 5.4

2303A51060

Batch-10

Task-1

Prompt: generate a Python script that collects user data like name, age, and email, then add comments on how to anonymise the data

Code :

```
# Collecting user data
name = input("Enter your name: ")
age = input("Enter your age: ")
email = input("Enter your email: ")

# Anonymizing the data
print("Anonymized Data:")

def anonymize_data(name, age, email):
    anonymized_name = "User123"
    anonymized_age = "25-34"
    anonymized_email = "user@example.com"

    print(f"Name: {name}")
    print(f"Age: {age}")
    print(f"Email: {email}")

    return anonymized_name, anonymized_age, anonymized_email

anon_name, anon_age, anon_email = anonymize_data(name, age, email)
print("Anonymized Data:")
print(f"Name: {anon_name}")
print(f"Age: {anon_age}")
print(f"Email: {anon_email}")
```

Output :

The screenshot shows a VS Code interface with the following details:

- File Explorer:** Shows files in the AI ASSISTANT folder, including 2303A51355\_ASSIGNMENT, app.log, lab-1.py, lab-2.py, lab-3.py, lab-3.4.py, lab-4.py, lab-4.3.py, lab-5.py, lab-5.4.py, and practice.py.
- Code Editor:** The active file is lab-5.4.py, containing Python code for generating a script to collect user data and anonymize it. The code uses input() to get name, age, and email, then applies a hash function to the name and creates an anonymized age based on the original age. It also handles email addresses by replacing them with masked versions.
 

```

1 #Task 1:
2 #generate a python script that collects user data like name,age and email then add comments on how to anonymize this data.
3 def collect_user_data():
4     name = input("Enter your name: ")
5     age = input("Enter your age: ")
6     email = input("Enter your email: ")
7     anonymized_name = "User" + str(hash(name)) % 1000
8     age = int(age)
9     if age <= 18:
10         anonymized_age = "0-18"
11     elif age <= 35:
12         anonymized_age = "19-35"
13     elif age <= 50:
14         anonymized_age = "36-50"
15     else:
16         anonymized_age = "51+"
17     email_parts = email.split('@')
18     if len(email_parts) == 2:
19         anonymized_email = email_parts[0] + "*****@" + email_parts[1]
20     else:

```
- Terminal:** Shows the command run in the terminal: PS C:\Users\saipr\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\Desktop\AI assistant\lab-5.4.py'. The output shows the program prompting for name, age, and email, and then printing the anonymized user data.

## Code Analysis :

- The program first asks the user to enter personal details like name, age, and email using input().
- These values are stored in variables so they can be processed later.
- The anonymize\_data() function replaces real data with dummy values to protect privacy.
- This shows how personal data can be hidden or masked before sharing or storing it.

## Task-2

Prompt: generate python func on for sentiment analysis than identify and handle potential biases in data used for analysis without using modules

### Code :

```
def simple_sentiment_analysis(text):
    positive_words = ['good', 'happy', 'joy', 'excellent', 'fortunate', 'correct', 'superior']
    negative_words = ['bad', 'sad', 'pain', 'terrible', 'unfortunate', 'wrong', 'inferior']
```

```
# Convert text to lowercase for uniformity
text = text.lower()
```

```
# Initialize counters
pos_count = 0
neg_count = 0
```

```
# Count positive and negative words

for word in positive_words:
    pos_count += text.count(word)

for word in negative_words:
    neg_count += text.count(word)

# Determine sentiment

if pos_count > neg_count:
    return "Positive Sentiment"
elif neg_count > pos_count:
    return "Negative Sentiment"
else:
    return "Neutral Sentiment"

# Example usage
user_input = input("Enter a sentence for sentiment analysis: ")
sentiment = simple_sentence_analysis(user_input)
print(f"The sentiment of the given text is: {sentiment}")
```

Output :

```

File Edit Selection View Go Run ...
lab-3.4.py lab-5.4.py lab-3.4.py
lab-5.4.py > simple_sentiment_analysis
30     print(anonymized_data)"""
31 #~<~~~#
32 #Task--2
33 #generate a python function for sentiment analysis.To identify and handle potential biases in the data.use balancing dataset and remove offensive terms.with
34 def simple_sentiment_analysis(text):
35     positive_words = ['good', 'happy', 'joy', 'excellent', 'fortunate', 'correct', 'superior']
36     negative_words = ['bad', 'sad', 'pain', 'terrible', 'unfortunate', 'wrong', 'inferior']
37     text = text.lower()
38     pos_count = 0
39     neg_count = 0
40     for word in positive_words:
41         pos_count += text.count(word)
42     for word in negative_words:
43         neg_count += text.count(word)
44     if pos_count > neg_count:
45         return "Positive Sentiment"
46     elif neg_count > pos_count:
47         return "Negative Sentiment"
48     else:
49         return "Neutral Sentiment"
50 user_input = input("Enter a sentence for sentiment analysis: ")
51 sentiment = simple_sentiment_analysis(user_input)
52 print(f"The sentiment of the given text is: {sentiment}")
53
54 #~<~~~#
55 #Task-3

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS C:\Users\saipr\OneDrive\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\.vscode\extensions\ms-python.debugpy-2025n32-x64\bundled\libs\debugpy\launcher' '5774' '<-->' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-5.4.py'

Enter a sentence for sentiment analysis: sad

The sentiment of the given text is: Negative Sentiment

### Code Analysis :

- The function checks the text for positive and negative words using predefined lists.
- The input text is converted to lowercase to avoid case-sensitive errors.
- It counts how many positive and negative words are present in the sentence.
- Based on the count, the program decides whether the sentiment is Positive, Negative, or Neutral.

### Task-3

Prompt : Generate python program to recommends products based on user history and follow ethical guidelines to avoid manipulative practices

```

def recommend_products(user_history):

    # Sample product database

    products = {

        'electronics': ['Smartphone', 'Laptop', 'Headphones'],

        'books': ['Fiction Novel', 'Science Textbook', 'Biography'],

        'clothing': ['T-Shirt', 'Jeans', 'Jacket']

    }

    recommendations = []

```

```

# Recommend products based on user history

for category in user_history:

if category in products:

    recommendations.extend(products[category])



# Ethical guideline: Avoid recommending products that are not relevant to user's interests

if not recommendations:

    return "No recommendations available based on your history."



```

```
return recommendations #
```

Example usage

```

user_history_input = ['electronics', 'books']

recommended_items = recommend_products(user_history_input)

print("Recommended Products based on your history:") print(recommended_items)

```

Output :

```

File Edit Selection View Go Run ... 🔍 AI assistant
lab-3.4.py lab-5.4.py ● lab-3.4.py
lab-5.4.py > ⚡ recommend_products
57 def recommend_products(user_history):
58     # Sample product database
59     products = {
60         'electronics': ['Smartphone', 'Laptop', 'Headphones'],
61         'books': ['Fiction Novel', 'Science Textbook', 'Biography'],
62         'clothing': ['T-Shirt', 'Jeans', 'Jacket']
63     }
64
65     recommendations = []
66     for category in user_history:
67         if category in products:
68             recommendations.extend(products[category])
69     if not recommendations:
70         return "No recommendations available based on your history."
71
72     return recommendations
73 user_history_input = ['electronics', 'books']
74 recommended_items = recommend_products(user_history_input)
75
76 print("Recommended Products based on your history:")
77 print(recommended_items)
78

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS C:\Users\saipr\OneDrive\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\.vscode\extensions\ms-vscode\debugpy\launcher' '56991' '--' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-5.4.py'
Recommended Products based on your history:
['Smartphone', 'Laptop', 'Headphones', 'Fiction Novel', 'Science Textbook', 'Biography']

```

Code Analysis :

- The program stores products in a dictionary based on categories like electronics and books.
- It checks the user's past interests (user\_history) to suggest related products.
- Only relevant items are recommended, avoiding unnecessary or misleading suggestions.
- This follows ethical guidelines by respecting user preferences and avoiding manipulation.

#### Task-4

Prompt: Generate python program that logging functionality in python web application and logs do not record sensitive information

Code :

```
import logging #  
  
Configure logging  
  
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s -  
%(message)s')  
def log_user_action(  
    acion,  
    user_id=None): # Avoid logging sensitive information like user_id  
    logging.info(f"User performed action: {action}")  
  
# Example usage  
log_user_action("Login")  
log_user_action("Viewed Product Page")  
log_user_action("Logout")
```

Output :

```

80 #!/usr/bin/env python
81 #generate logging functionality in a python web application.do not record sensitive information like passwords or personal data.
82 import logging
83 def setup_logging():
84     # configure logging
85     logging.basicConfig(
86         level=logging.INFO,
87         format='%(asctime)s - %(levelname)s - %(message)s',
88         handlers=[
89             logging.FileHandler("app.log"),
90             logging.StreamHandler()
91         ]
92     )
93 def log_user_action(user_id, action):
94     # Log user actions without sensitive information
95     logging.info(f"User {user_id} performed action: {action}")
96 # Example usage
97 if __name__ == "__main__":
98     setup_logging()
99     log_user_action("User123", "Logged in")
100    log_user_action("User123", "Viewed product page")
101    log_user_action("User123", "Logged out")
102 #~~~~~#

```

PS C:\Users\saipr\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\.vscode\extensions\ms-python.python.lib\debug\launcher' '65380' '--' 'c:\Users\saipr\Desktop\AI assistant\lab-5.4.py'

2026-02-05 23:36:44,334 - INFO - User User123 performed action: Logged in  
2026-02-05 23:36:44,335 - INFO - User User123 performed action: Viewed product page  
2026-02-05 23:36:44,336 - INFO - User User123 performed action: Logged out

## Code Analysis :

- The program uses Python's logging feature to record user actions.
- It logs only general actions like login or logout, not private data.
- Sensitive details such as user ID or passwords are intentionally avoided.
- This improves system monitoring while maintaining user privacy and security.

## Task 5

Prompt : Generate python program that machine learning model than add documenta on on how to use the model like explainability ,accuracy limkits .

code :

```

def simple_ml_model(data):
    # A simple placeholder function for a machine learning model
    # In a real scenario, this would involve training a model on the provided data
    model_accuracy = 0.85 # Example accuracy

    return model_accuracy

# Documenta on:
"""

This function represents a simple machine learning model.

```

It takes input data and returns an accuracy score.

Explainability: The model is a placeholder and does not provide detailed explanations.

Accuracy Limitations: The accuracy is hardcoded for demonstration purposes.

\*\*\*\*\*

```
# Example usage

input_data = [1, 2, 3, 4, 5]

accuracy = simple_ml_model(input_data)

print(f"The model accuracy is: {accuracy * 100}%")
```

Output :

The screenshot shows the VS Code interface with the code editor displaying `lab-5.4.py`. The terminal at the bottom shows the execution of the script and its output:

```
PS C:\Users\saipr\OneDrive\Desktop\AI assistant> & 'c:\Users\saipr\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\saipr\.vscode\extensions\ms-python.debug.libs\debugpy\launcher' '56157' '--' 'c:\Users\saipr\OneDrive\Desktop\AI assistant\lab-5.4.py'
The model accuracy is: 99.0%
PS C:\Users\saipr\OneDrive\Desktop\AI assistant>
```

Code Analysis :

- The function represents a basic machine learning model using a placeholder.
- It returns a fixed accuracy value for demonstration purposes.
- Comments explain that the model does not show real predictions or explanations.
- Documentation clearly mentions limitations in accuracy and explainability.