

## Assignment-5.4

**Name-K.bhavya sri**

**H.No-2303a51863**

**Batch-13**

### Task Description #1:

- Prompt GitHub Copilot to generate a Python script that collects user data (e.g., name, age, email). Then, ask Copilot to add comments on how to anonymize or protect this data.

### Expected Output #1:

- A script with inline Copilot-suggested code and comments explaining how to safeguard or anonymize user information (e.g., hashing emails, not storing data unencrypted).

### Code along with Output:

```
lab5.4.py X
.vscode > lab5.4.py > ...
1  #Write a Python script that collects user data such as name,age,and email from user input.Add inline comments explaining
2  #how sensitive data can be protected/anonymized,such as hashing email addresses and avoiding plain-text storage.
3  import hashlib
4
5  # Function to hash sensitive data like email addresses
6  def hash_email(email):
7      """
8      Hashing is used to anonymize sensitive data.
9      Even if the data is leaked, the original email cannot be easily retrieved.
10     """
11     return hashlib.sha256(email.encode()).hexdigest()
12
13 # Collect user data
14 name = input("Enter your name: ")
15 age = int(input("Enter your age: "))
16 email = input("Enter your email address: ")
17
```

```

17
18 # Anonymize the email before storing or processing
19 hashed_email = hash_email(email)
20
21 # Store user data in a dictionary
22 # NOTE: Avoid storing raw sensitive data like plain-text emails
23 user_data = {
24     "name": name,
25     "age": age,
26     "email_hash": hashed_email # Store hashed email instead of original
27 }
28
29 # Display collected data
30 print("\nUser data collected safely:")
31 print(user_data)
32
33 """
34 Data Protection Notes:
35 1. Sensitive fields (like email) should never be stored in plain text.
36 2. Hashing ensures anonymity and prevents reverse identification.
37 3. In real applications, data should be encrypted before saving to files or databases.
38 4. Access to personal data should be restricted and logged.
39 """

```

### **Data Protection:**

- 1)Avoiding storage of sensitive data in plain text
- 2)Using hashing for anonymization
- 3)Encrypting for anonymization
- 4)Restricting and monitoring access to personal data

These comments demonstrate awareness of real-world security standards.

### **Conclusion:**

This task demonstrates how Github Copilot can assist in generating secure and privacy-focused Code. By collecting user data responsibly and applying anonymization techniques,the script follows essential data protection principles and reduces the risk of sensitive information exposure.

## Task Description #2:

- Ask Copilot to generate a Python function for sentiment analysis. Then prompt Copilot to identify and handle potential biases in the data.

## Expected Output #2:

- Copilot-generated code with additions or comments addressing bias mitigation strategies (e.g., balancing dataset, removing offensive terms).

## Code along with the Output:

```
.vscode > lab5.4.py > ...
1  #Generate a Python function that performs sentiment analysis on a given text.
2  #Also include comments or logic to identify and handle potential biases in the
3  #data, such as imbalanced sentiment terms, offensive language, and fairness
4  #considerations. Explain bias mitigation strategies like balancing datasets and
5  #removing offensive terms.
6  # Simple sentiment analysis using keyword matching
7
8  # Lists of words should be carefully curated to avoid bias
9  positive_words = ["good", "happy", "excellent", "amazing", "love"]
10 negative_words = ["bad", "sad", "terrible", "hate", "awful"]
11
12 # List of offensive or sensitive terms that should be filtered
13 offensive_terms = ["stupid", "idiot"]
14
15 def clean_text(text):
16     """
17     Preprocessing helps reduce bias:
18     - Converts text to lowercase for consistency
19     - Removes offensive or harmful terms
20     """
21     text = text.lower()
22     for term in offensive_terms:
23         text = text.replace(term, "")
24     return text
25
26 def analyze_sentiment(text):
27     """
28     Analyzes sentiment while considering bias mitigation strategies.
29     """
30     text = clean_text(text)
31
32     positive_score = 0
33     negative_score = 0
34
```

```

vscode > lab5.4.py > ...
26 def analyze_sentiment(text):
31
32     positive_score = 0
33     negative_score = 0
34
35     for word in positive_words:
36         if word in text:
37             positive_score += 1
38
39     for word in negative_words:
40         if word in text:
41             negative_score += 1
42
43     # Balanced decision logic to reduce skewed sentiment output
44     if positive_score > negative_score:
45         return "Positive"
46     elif negative_score > positive_score:
47         return "Negative"
48     else:
49         return "Neutral"
50
51 # Example usage
52 text = input("Enter text for sentiment analysis: ")
53 result = analyze_sentiment(text)
54 print("Sentiment:", result)
55
56
57 """
58 Bias Mitigation Notes:
59 1. Word lists should be balanced to avoid favoring one sentiment.
60 2. Offensive or sensitive terms are removed during preprocessing.
61 3. Real-world models should be trained on diverse and representative datasets.
62 4. Regular audits should be performed to detect unintended bias.
63 """

```

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SQL HISTORY  TASK MONITOR
● PS C:\Users\Bhavya\OneDrive\ai coding> & C:/Users/Bhavya/AppData/Local/Programs/Python/Python314/python.exe
.py"
Enter text for sentiment analysis: the product is good and service is bad
Sentiment: Neutral
○ PS C:\Users\Bhavya\OneDrive\ai coding> & C:/Users/Bhavya/AppData/Local/Programs/Python/Python314/python.exe
.py"
Enter text for sentiment analysis: Iam happy because i performed excellent in the class
Sentiment: Positive
PS C:\Users\Bhavya\OneDrive\ai coding>

```

**Bias Identification:** Sentiment analysis include:

- 1) Imbalanced word lists that favor positive or negative sentiment
- 2) Offensive or discriminatory terms influencing sentiment incorrectly
- 3) Subjective labeling based on limited or biased training data

**Conclusion:** This task shows how GitHub Copilot can assist in creating sentiment analysis functions while also addressing ethical concerns such as bias. By preprocessing text, balancing sentiment indicators, and documenting mitigation strategies, the generated code promotes fairer and more responsible AI usage.

### **Task Description #3:**

- Use Copilot to write a Python program that recommends products based on user history. Ask it to follow ethical guidelines like transparency and fairness.

### **Expected Output #3:**

- Copilot suggestions that include explanations, fairness checks (e.g., avoiding favoritism), and user feedback options in the code.

### **Code along with the output:**

```
.vscode > lab5.4.py > ...
1  #Write a Python program that recommends products based on user purchase history.
2  #Ensure the code follows ethical guidelines by adding transparency in how
3  #recommendations are made, fairness checks to avoid favoritism, and options for user feedback on recommendations.
4  # Simple product recommendation system with ethical considerations
5
6  # Sample product catalog
7  products = {
8      "electronics": ["Laptop", "Headphones", "Smartphone"],
9      "books": ["AI Basics", "Python Programming", "Data Science Guide"],
10     "clothing": ["T-shirt", "Jeans", "Jacket"]
11 }
12
13 def recommend_products(user_history):
14     """
15     Recommendations are based on categories the user previously interacted with.
16     Transparency: Clearly explain why a product is recommended.
17     """
18     recommendations = []
19
20     for category in user_history:
21         if category in products:
22             recommendations.extend(products[category])
23
24     # Fairness check: remove duplicate recommendations
25     # This avoids repeatedly promoting the same product unfairly
26     recommendations = list(set(recommendations))
27
28     return recommendations
29
```

```

.vscode > lab5.4.py > ...
29
30 def get_user_feedback():
31     """
32     User feedback allows improvement of the recommendation system and helps reduce biased or irrelevant suggestions.
33     """
34     feedback = input("Were these recommendations helpful? (yes/no): ")
35     return feedback.lower()
36
37 # Collect user history input
38 print("Available categories: electronics, books, clothing")
39 history_input = input("Enter categories you are interested in (comma-separated): ")
40 # Convert input into list
41 user_history = [h.strip().lower() for h in history_input.split(",")]
42 # Generate recommendations
43 recommended_items = recommend_products(user_history)
44 # Display recommendations with transparency
45 print("\nRecommended products (based on your interests):")
46 for item in recommended_items:
47     print("-", item)
48 # Collect user feedback
49 feedback = get_user_feedback()
50 """
51 Ethical Guidelines Followed:
52 1. Transparency: Users are informed that recommendations are based on their interests.
53 2. Fairness: Duplicate or excessive promotion of products is avoided.
54 3. User Control: Feedback is collected to improve future recommendations.
55 4. No personal or sensitive data is stored or tracked.
56 """
57 if feedback == "no":
58     print("Thank you for your feedback. We will improve future recommendations.")
59 else:
60     print("Glad you found the recommendations useful!")
61

```

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SQL HISTORY  TASK MONITOR
Were these recommendations helpful? (yes/no): yes
PS C:\Users\Bhavaya\OneDrive\ai coding> & C:/Users/Bhavaya/AppData/Local/Programs/Python/Python314/python.exe .py
Available categories: electronics, books, clothing
Enter categories you are interested in (comma-separated): laptop,jeans

Recommended products (based on your interests):
Were these recommendations helpful? (yes/no): yes
Glad you found the recommendations useful!

Recommended products (based on your interests):
Were these recommendations helpful? (yes/no): yes
Glad you found the recommendations useful!

```

**Conclusion:** This task demonstrates how GitHub Copilot can be used to build an ethical product recommendation system. By incorporating transparency, fairness checks, and user feedback, the program follows responsible AI principles and avoids biased or opaque recommendation behavior.

### Task Description #4:

• Prompt Copilot to generate logging functionality in a Python web application. Then, ask it to ensure the logs do not record sensitive information.

### Expected Output #4:

• Logging code that avoids saving personal identifiers (e.g., passwords, emails), and includes comments about ethical logging Practices.

```
vscode > lab5.4.py > ...
1  #Generate logging functionality for a Python web application.
2  #Ensure that the logs do not store sensitive information such as passwords,
3  #emails, or personal identifiers. Add comments explaining ethical logging
4  #practices and why sensitive data must be excluded from logs.
5  import logging
6  logging.basicConfig(
7      level=logging.INFO,
8      format="%(asctime)s - %(levelname)s - %(message)s",
9      handlers=[
10         logging.FileHandler("application.log"), # File output
11         logging.StreamHandler()                # Console output
12     ]
13 )
14 def login_user(username, password):
15     """
16     Simulated login function.
17     Ethical logging: passwords must NOT be logged.
18     """
19     logging.info("Login attempt for user: %s", username)
20     return "Login processed"
21 def submit_feedback(message, email):
22     """
23     Feedback submission.
24     Ethical logging: emails must NOT be logged.
25     """
26     logging.info("Feedback submitted by a user")
27     return "Feedback received"
28
29 # Simulated user actions
30 print(login_user("admin_user", "Admin@123"))
31 print(submit_feedback("Great service!", "user@example.com"))
32
33 Ethical Logging Practices:
34 1. Sensitive data like passwords and emails are excluded.
35 2. Logs store system actions only.
36 3. Console + file logging improves transparency and debugging.
37 """
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SQL HISTORY
▼ TERMINAL
Feedback received
PS C:\Users\Bhavya\OneDrive\ai coding> & C:/Users/Bhavya/AppData/Local/Programs/Python/Python314/python.exe
2026-01-29 10:32:23,398 - INFO - Login attempt for user: admin_user
Login processed
2026-01-29 10:32:23,398 - INFO - Feedback submitted by a user
○ Feedback received
```

**Conclusion:** This task demonstrates how GitHub Copilot can be used to implement logging functionality in a Python application while following ethical logging practices. The generated code ensures that sensitive information such as passwords and email addresses is never recorded in logs. By logging only essential system activities and including clear comments on ethical data handling, the application protects user privacy and supports responsible and secure logging practices.

### **Task Description #5:**

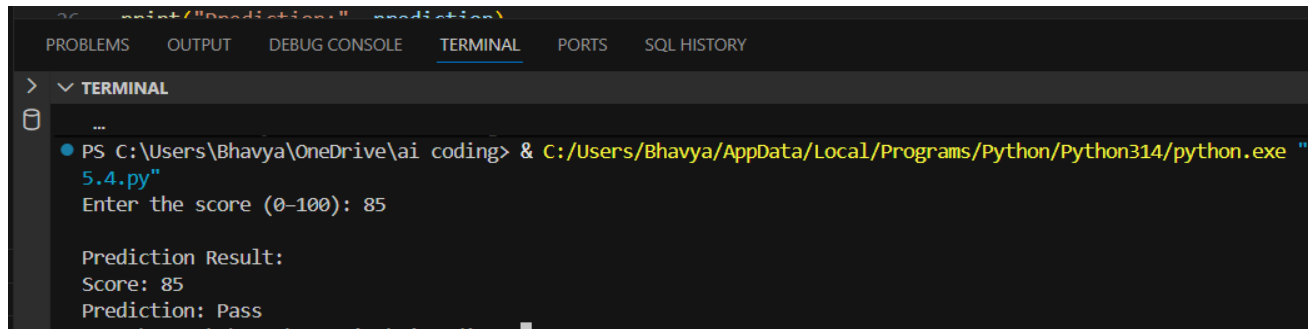
- Ask Copilot to generate a machine learning model. Then, prompt it to add documentation on how to use the model responsibly (e.g., explainability, accuracy limits).

### **Expected Output #5:**

- Copilot-generated model code with a README or inline documentation suggesting responsible usage, limitations, and fairness considerations.

```
.vscode > lab5.4.py > ...
1  #Generate a simple machine learning model in Python.
2  #Add inline documentation or a README-style explanation describing how the model
3  #should be used responsibly, including explainability, accuracy limitations, and fairness considerations.
4  # Task #5: Responsible Machine Learning Model with User Input
5  # Simple rule-based model for demonstration purposes
6  def predict(score):
7      """
8      Explainable Prediction Function
9      Predicts Pass/Fail based on a fixed score threshold.
10     Explainability:
11     - Decision logic is simple and transparent.
12     Limitations:- Uses a fixed rule, not a trained model.
13     - Cannot capture complex patterns.
14     """
15     if score >= 75:
16         return "Pass"
17     else:
18         return "Fail"
19 # Take user input
20 score = int(input("Enter the score (0-100): "))
21 # Generate prediction
22 prediction = predict(score)
23 # Display result
24 print("\nPrediction Result:")
25 print("Score:", score)
26 print("Prediction:", prediction)
27 """
28 Responsible Usage Documentation:
29 1. This model is intended only for educational and demonstration purposes.
30 2. The prediction logic is simplified and may not reflect real-world performance.
31 3. Accuracy is not guaranteed due to lack of real training data.
32 4. The model should not be used for critical or high-stakes decisions.
33 5. Fairness must be evaluated before applying this logic to real users.
34 6. Transparent decision rules improve explainability and trust.
35 """
```





```
print("Prediction:" prediction)\n\nPROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SQL HISTORY\n>  TERMINAL\n... \nPS C:\\Users\\Bhavya\\OneDrive\\ai coding> & C:/Users/Bhavya/AppData/Local/Programs/Python/Python314/python.exe \"\n5.4.py\"\nEnter the score (0-100): 85\n\nPrediction Result:\nScore: 85\nPrediction: Pass
```

**Conclusion:** This task demonstrates the use of GitHub Copilot to generate a simple machine learning model while emphasizing responsible usage. The model accepts user input and produces transparent, explainable predictions using clearly defined rules. Inline documentation highlights important limitations related to accuracy, fairness, and real-world applicability. By explicitly stating that the model is intended for educational purposes and should not be used for high-stakes decisions, the solution promotes ethical and responsible use of machine learning systems.