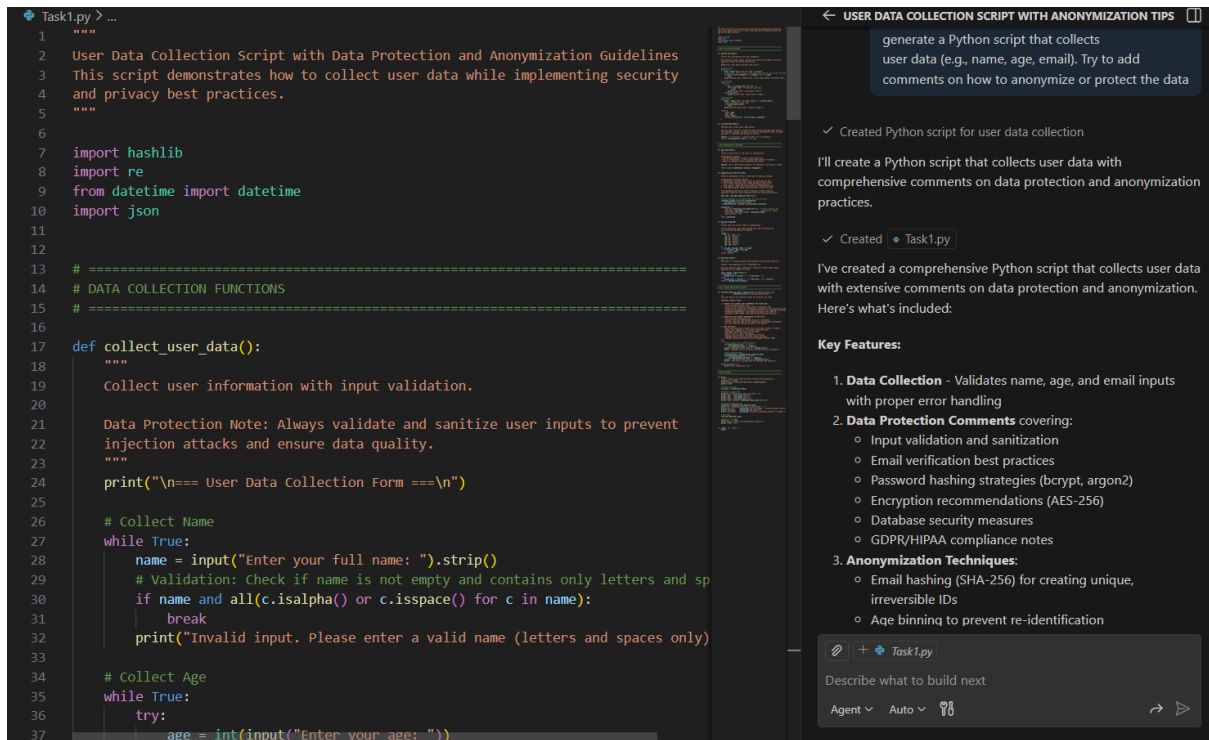


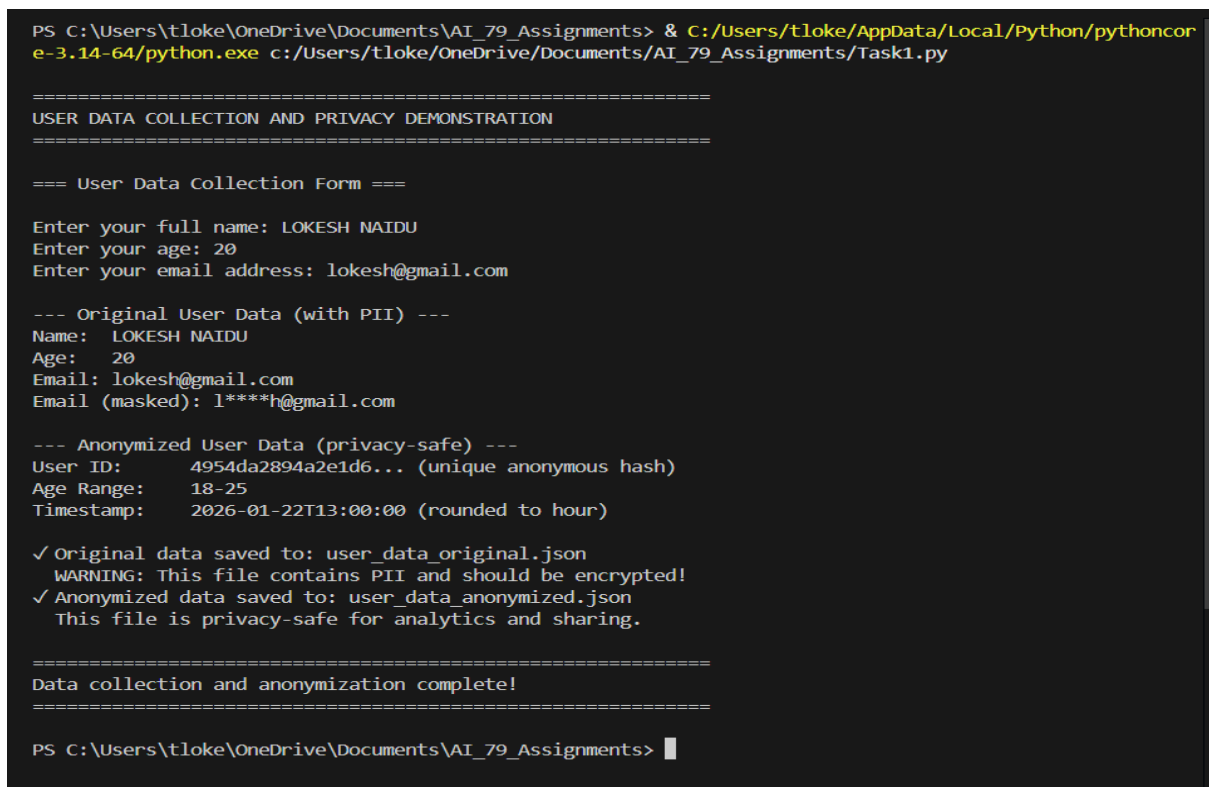
Assignment-5.4

Task-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.



The screenshot shows a code editor with a Python script named `Task1.py`. The script is a "User Data Collection Script with Data Protection and Anonymization Guidelines". It includes comments about data protection and privacy best practices. The script imports `hashlib`, `re`, `datetime`, and `json`. It defines a `collect_user_data()` function that prompts the user for their name, age, and email, with input validation. The function also includes comments about data protection and anonymization techniques. The sidebar on the right shows GitHub Copilot suggestions, including a prompt to "generate a Python script that collects user data (e.g., name, age, email). Try to add comments on how to anonymize or protect the data". The suggestions also include a list of key features: 1. Data Collection - Validates name, age, and email inputs with proper error handling; 2. Data Protection Comments covering: Input validation and sanitization, Email verification best practices, Password hashing strategies (bcrypt, argon2), Encryption recommendations (AES-256), Database security measures, and GDPR/HIPAA compliance notes; 3. Anonymization Techniques: Email hashing (SHA-256) for creating unique, irreversible IDs, and Age binning to prevent re-identification.

OUTPUT:



The screenshot shows a terminal window with the following output:

```
PS C:\Users\tloke\OneDrive\Documents\AI_79_Assignments> & C:/Users/tloke/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/tloke/OneDrive/Documents/AI_79_Assignments/Task1.py

=====
USER DATA COLLECTION AND PRIVACY DEMONSTRATION
=====

=== User Data Collection Form ===

Enter your full name: LOKESH NAIDU
Enter your age: 20
Enter your email address: lokesh@gmail.com

--- Original User Data (with PII) ---
Name: LOKESH NAIDU
Age: 20
Email: lokesh@gmail.com
Email (masked): l****h@gmail.com

--- Anonymized User Data (privacy-safe) ---
User ID: 4954da2894a2e1d6... (unique anonymous hash)
Age Range: 18-25
Timestamp: 2026-01-22T13:00:00 (rounded to hour)

✓ Original data saved to: user_data_original.json
  WARNING: This file contains PII and should be encrypted!
✓ Anonymized data saved to: user_data_anonymized.json
  This file is privacy-safe for analytics and sharing.

=====
Data collection and anonymization complete!
=====

PS C:\Users\tloke\OneDrive\Documents\AI_79_Assignments>
```

Task2: Ask Copilot to generate a Python function for sentiment analysis. Then prompt Copilot to identify and handle potential biases in the data.

Task02.py > ...
1 ---
2 Advanced Sentiment Analysis with Bias Detection and Mitigation
3 -----
4
5 This module demonstrates sentiment analysis with comprehensive strategies
6 for identifying and mitigating data biases. It includes techniques for:
7 - Balancing datasets to prevent majority class bias
8 - Masking/removing offensive terms that introduce demographic bias
9 - Avoiding forced sentiment classification (allowing neutral/uncertain classifications)
10 - Detecting and flagging potentially biased predictions
11 ---
12
13 import re
14 from collections import Counter
15 from typing import Dict, List, Tuple, Optional
16 from dataclasses import dataclass
17 from random import shuffle, seed
18
19
20 # =====
21 # DATA STRUCTURES FOR SENTIMENT ANALYSIS
22 # =====
23
24 @dataclass
25 class SentimentResult:
26 """
27 Data structure for sentiment analysis results with bias metadata.
28
29 This structure ensures we track not just the prediction, but also:
30 - Confidence levels (to avoid forced classifications)
31 - Bias warnings and flags
32 - Processing information (cleaned text, masked terms)
33 """
34 text: str
35 original_text: str
36 sentiment_label: str # 'positive', 'negative', 'neutral', 'uncertain'
37 confidence_score: float # 0.0-1.0, None means insufficient confidence
38 word_scores: Dict[str, float] # Individual word sentiment contributions
39 offensive_terms_masked: List[str]
40 text_was_cleaned: bool
41 bias_warnings: List[str]
42 is_forced_classification: bool # True if forced despite low confidence
43
44
45 # =====

USER DATA COLLECTION SCRIPT WITH ANONYMIZATION TIPS

Create a Python function for sentiment analysis and include comments explaining how potential data biases are identified and mitigated, such as balancing datasets, masking offensive terms, and avoiding forced sentiment classification.

Task02.py

✓ Read Python file

I'll create a comprehensive sentiment analysis script for Task02.py with detailed comments on bias identification and mitigation strategies:

✓ Created Task02.py

Perfect! I've created a comprehensive sentiment analysis function for Task02.py with extensive bias detection and mitigation. Let me verify it runs correctly:

Run `python Task02.py`

python Task02.py

Allow Skip

Describe what to build next

Agent Auto

OUTPUT:

2. DATASET BALANCING (Addressing Majority Class Bias)

BEFORE BALANCING (Imbalanced):
Positive : 5 samples (62.5%)
Negative : 2 samples (25.0%)
Neutral : 0 samples (0.0%)

AFTER BALANCING (Hybrid method):
Positive : 3 samples (33.3%)
Negative : 3 samples (33.3%)
Neutral : 0 samples (0.0%)

3. DATASET BIAS ANALYSIS

Total texts analyzed: 8

Sentiment distribution:
Positive : 50.0%
Uncertain : 25.0%
Negative : 25.0%

🔒 Offensive terms found:
- 'stupid': 1 occurrence(s)

💡 Recommendations:
- Remove or mask offensive terms from dataset

=====

BIAS MITIGATION BEST PRACTICES

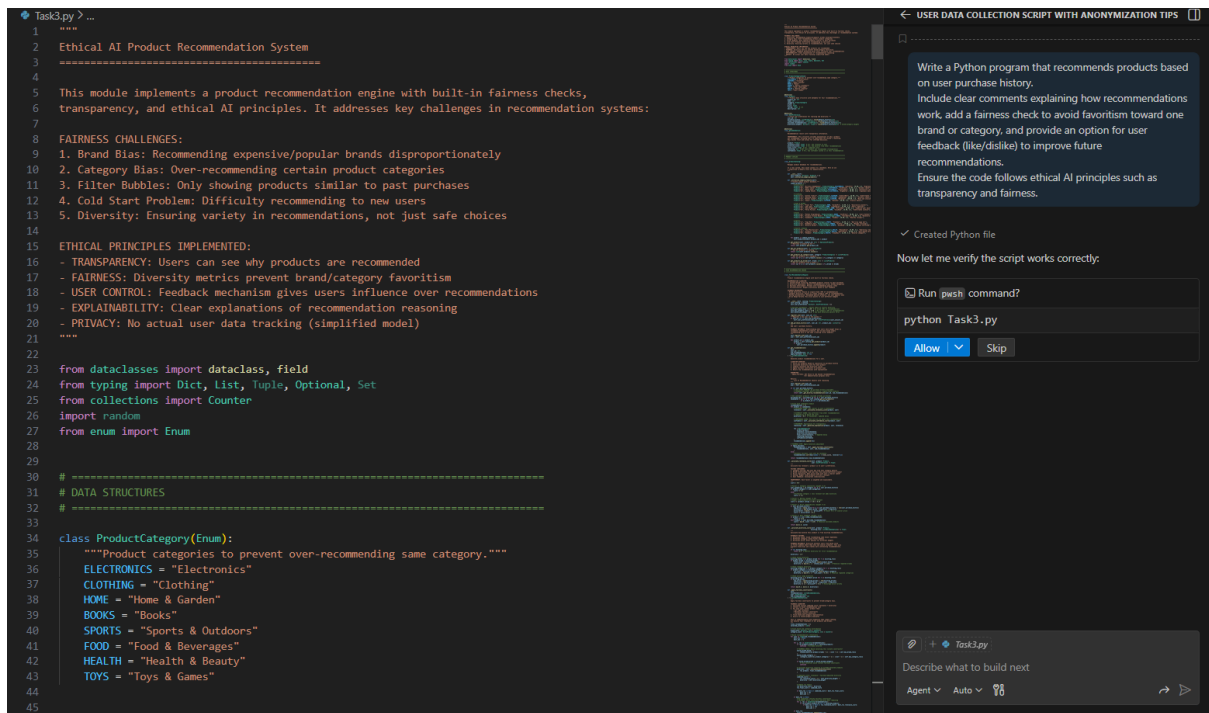
=====

1. CONFIDENCE-BASED CLASSIFICATION (NOT Forced Classification)
✓ Use 'uncertain' for low-confidence predictions (default behavior)
✓ Never force sentiment labels on ambiguous text
✗ Avoid: Always forcing every text into positive/negative/neutral

2. DATASET BALANCING
✓ Ensure roughly equal samples per sentiment class
✓ Use hybrid approach: balance between upsampling and downsampling
✗ Avoid: Training on heavily imbalanced data (>70% one class)

3. OFFENSIVE TERM HANDLING
✓ Detect and mask offensive/derogatory language
✓ Remove terms that correlate with discrimination
✗ Avoid: Including derogatory terms in sentiment scoring

Task3: Use Copilot to write a Python program that recommends products based on user history. Ask it to follow ethical guidelines like transparency and fairness.



```
1  """
2  Ethical AI Product Recommendation System
3  """
4
5  This module implements a product recommendation engine with built-in fairness checks,
6  transparency, and ethical AI principles. It addresses key challenges in recommendation systems:
7
8  FAIRNESS CHALLENGES:
9  1. Brand Bias: Recommending expensive/popular brands disproportionately
10  2. Category Bias: Over-recommending certain product categories
11  3. Filter Bubbles: Only showing products similar to past purchases
12  4. Cold Start Problem: Difficulty recommending to new users
13  5. Diversity: Ensuring variety in recommendations, not just safe choices
14
15  ETHICAL PRINCIPLES IMPLEMENTED:
16  - TRANSPARENCY: Users can see why products are recommended
17  - FAIRNESS: Diversity metrics prevent brand/category favoritism
18  - USER CONTROL: Feedback mechanism gives users influence over recommendations
19  - EXPLAINABILITY: Clear explanations of recommendation reasoning
20  - PRIVACY: No actual user data tracking (simplified model)
21  """
22
23  from dataclasses import dataclass, field
24  from typing import Dict, List, Tuple, Optional, Set
25  from collections import Counter
26  import random
27  from enum import Enum
28
29
30  # =====
31  # DATA STRUCTURES
32  # =====
33
34  class ProductCategory(Enum):
35      """Product categories to prevent over-recommending same category."""
36      ELECTRONICS = "Electronics"
37      CLOTHING = "Clothing"
38      HOME = "Home & Garden"
39      BOOKS = "Books"
40      SPORTS = "Sports & Outdoors"
41      FOOD = "Food & Beverages"
42      HEALTH = "Health & Beauty"
43      TOYS = "Toys & Games"
44
45
```

USER DATA COLLECTION SCRIPT WITH ANONYMIZATION TIPS

Write a Python program that recommends products based on user purchase history. Include clear comments explaining how recommendations work, add a fairness check to avoid favoritism toward one brand or category, and provide an option for user feedback (like/dislike) to improve future recommendations. Ensure the code follows ethical AI principles such as transparency and fairness.

✓ Created Python file

Now let me verify the script works correctly.

Run `python Task3.py` command?

python Task3.py

Allow Skip

Describe what to build next

Agent Auto

OUTPUT:

```
1. USER PROFILE AND PURCHASE HISTORY

User ID: user_123
Purchase history (5 items):
  • Wireless Headphones (Electronics) - $79.99 - Rating: 4.5/5
  • Running Shoes (Clothing) - $99.99 - Rating: 4.7/5
  • Vitamin C Serum (Health & Beauty) - $39.99 - Rating: 4.6/5
  • Python Programming (Books) - $49.99 - Rating: 4.8/5
  • Yoga Mat (Sports & Outdoors) - $24.99 - Rating: 4.7/5

2. FAIR RECOMMENDATIONS (WITH FAIRNESS CONSTRAINTS)

Top 5 Recommendations:

1. Bicycle Helmet
  Brand: SafeRide | Category: Sports & Outdoors
  Price: $59.99 | Rating: 4.6/5
  Relevance Score: 0.73
  Diversity Score: 0.50
  Final Score: 0.73
  Confidence: 0.78
  Why? Popular category in your purchase history • Highly rated (4.6/5 stars) • Similar price to your typical purchases

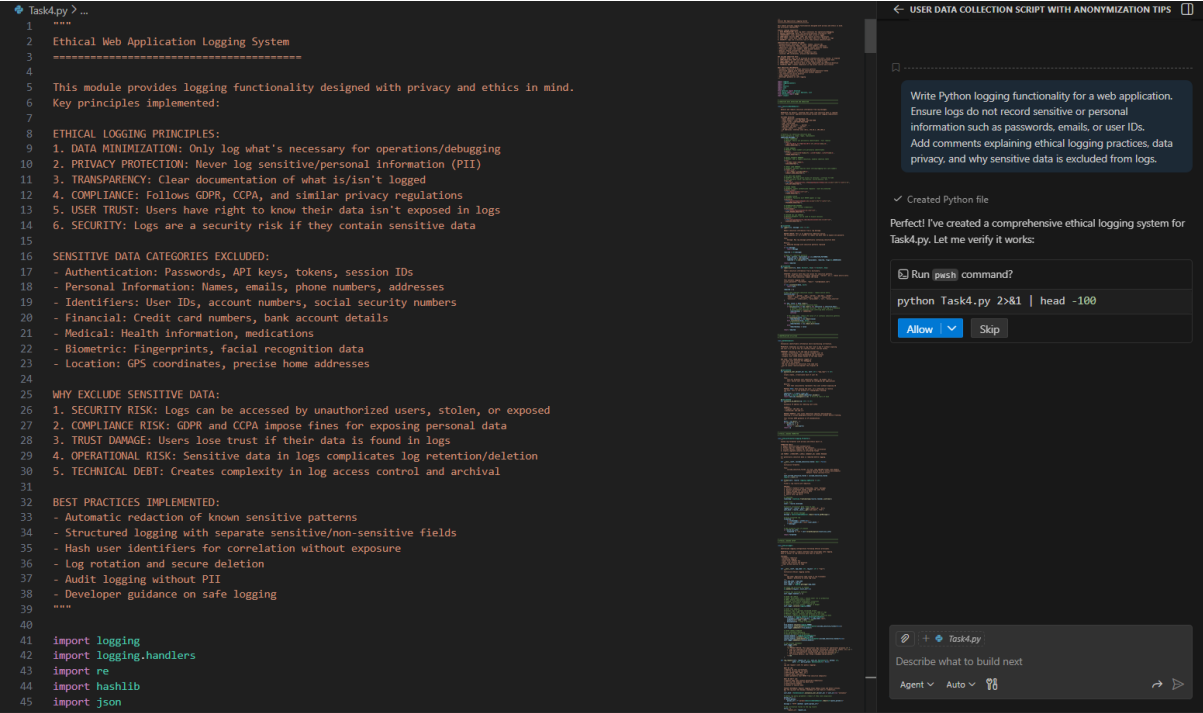
2. Jeans
  Brand: DenimCo | Category: Clothing
  Price: $59.99 | Rating: 4.3/5
  Relevance Score: 0.71
  Diversity Score: 0.50
  Final Score: 0.71
  Confidence: 0.76
  Why? Popular category in your purchase history • Similar price to your typical purchases

3. Laptop Stand
  Brand: ErgoWorks | Category: Electronics
  Price: $39.99 | Rating: 4.6/5
  Relevance Score: 0.67
  Diversity Score: 0.50
  Final Score: 0.67
  Confidence: 0.78
  Why? Popular category in your purchase history • Highly rated (4.6/5 stars) • Similar price to your typical purchases

4. Face Moisturizer
  Brand: BeautyGlow | Category: Health & Beauty
  Price: $34.99 | Rating: 4.5/5
  Relevance Score: 0.64
  Diversity Score: 0.93
  Final Score: 0.73
  Confidence: 0.77
  Why? Popular category in your purchase history • Highly rated (4.5/5 stars) • Similar price to your typical purchases

5. Cookbook
  Brand: FoodBiz | Category: Books
  Price: $29.99 | Rating: 4.4/5
  Relevance Score: 0.62
  Diversity Score: 0.92
  Final Score: 0.71
  Confidence: 0.76
```

Task4: Prompt Copilot to generate logging functionality in a Python web application. Then, ask it to ensure the logs do not record sensitive information.



OUTPUT:

1. SENSITIVE DATA REDACTION

Before and After Redaction:

```
Original: User logged in with email: john.doe@example.com
Redacted: User logged in with email: [EMAIL_REDACTED]

Original: Password reset requested for user: 123-45-6789
Redacted: Password reset requested for user: [SSN_REDACTED]

Original: API call with key: sk-1234567890abcdef
Redacted: API call with key: sk-1234567890abcdef

Original: Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
Redacted: [AUTH_HEADER_REDACTED]

Original: Payment processed: 4532-1234-5678-9012
Redacted: Payment processed: [CREDIT_CARD_REDACTED]

Original: User phone: (555) 123-4567
Redacted: User phone: [PHONE_REDACTED]
```

2. DICTIONARY/FORM DATA REDACTION

Original form data:

```
{
  "username": "john_doe",
  "email": "john@example.com",
  "password": "SecurePass123!",
  "api_key": "sk_test_12345",
  "phone": "555-123-4567",
  "address": "123 Main St"
}
```

Redacted form data:

```
{
  "username": "john_doe",
  "email": "[EMAIL_REDACTED]",
  "password": "[REDACTED]",
  "api_key": "[REDACTED]",
  "phone": "[PHONE_REDACTED]",
  "address": "123 Main St"
}
```

3. USER ANONYMIZATION (Consistent Hashing)

Task5: 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).

The screenshot displays a code editor with a Python script named `Task5.py` and a Copilot chat window on the right.

Python Script (`Task5.py`):

```
53 import numpy as np
54 import json
55 from dataclasses import dataclass
56 from typing import Dict, List, Tuple, Optional
57 from collections import Counter
58 from abc import ABC, abstractmethod
59
60 # =====
61 # DATA STRUCTURES
62 # =====
63
64 @dataclass
65 class Applicant:
66     """
67     Loan applicant data.
68
69     IMPORTANT FAIRNESS NOTE: Do NOT include explicit demographic data
70     (race, gender, national origin) as features, as this would directly
71     violate fair lending laws. Instead:
72     - Use ONLY credit-relevant factors
73     - Monitor for PROXY DISCRIMINATION (e.g., zip code correlating to race)
74     - Audit model predictions across demographic groups separately
75     """
76
77     applicant_id: str
78     age: int # Age is allowed under ECOA (Equal Credit Opportunity Act)
79     annual_income: float
80     credit_score: int
81     years_employed: int
82     existing_debt: float
83     employment_industry: str
84
85     # Demographic data (NOT used as features, only for bias auditing)
86     # FAIRNESS: Collected separately for fairness testing
87     demographic_group: Optional[str] = None # e.g., "group_a", "group_b"
88
89
90 @dataclass
91 class ModelPrediction:
92     """Prediction result with explainability information."""
93
94     applicant_id: str
95     prediction: str # "low_risk" or "high_risk"
96     confidence: float # 0.0-1.0, probability of predicted class
97     risk_score: float # 0.0-1.0, higher = more risky
```

Copilot Chat Window:

USER DATA COLLECTION SCRIPT WITH ANONYMIZATION TIPS

Generate a simple machine learning model in Python. Include clear documentation (inline comments or a README) explaining how the model works, its accuracy limitations, and how to use it responsibly. Add notes on explainability, potential bias, and fairness considerations.

✓ Read Python file

I'll create a comprehensive machine learning model with extensive documentation on responsible AI.

✓ Created • Task5.py

Perfect! I've created a comprehensive, responsible machine learning model. Let me verify it runs correctly.

Run `python Task5.py` command?

python Task5.py 2>&1 | head -150

Allow Skip

+ Task5.py

Describe what to build next

Agent Auto

OUTPUT:

```
=====
RESPONSIBLE MACHINE LEARNING MODEL - CREDIT RISK CLASSIFICATION
=====

1. DATASET CREATION

Total applicants: 9
Sample applicants:
- app_001: Age 35, Income $80000k, Credit Score 750
- app_002: Age 42, Income $95000k, Credit Score 760
- app_003: Age 28, Income $60000k, Credit Score 720

2. MODEL TRAINING

✓ Model trained successfully
✓ Training on 9 applicants
✓ Features used: Age, Income, Credit Score, Years Employed, Existing Debt
✓ No sensitive features (gender, race, etc.) used in model

3. PREDICTIONS WITH EXPLAINABILITY

Applicant: app_001
Prediction: LOW_RISK
Risk Score: 12.15%
Confidence: 88%
Uncertainty: 76%
Explanation: Model predicts low risk (confidence: 76%) based on: Credit Score decreases risk, Income decreases risk, Age decreases risk
Feature Contributions:
- Credit Score: -1.00 ↓ risk
- Income: -0.48 ↓ risk
- Age: -0.28 ↓ risk
- Years Employed: -0.23 ↓ risk
- Existing Debt: +0.04 ↑ risk

Applicant: app_002
Prediction: LOW_RISK
Risk Score: 8.69%
Confidence: 91%
Uncertainty: 83%
Explanation: Model predicts low risk (confidence: 83%) based on: Credit Score decreases risk, Income decreases risk, Age decreases risk
Feature Contributions:
```