

AI ASSISTED CODING LAB TEST - 03

NAME: Mohammed Anas Siddiqui

ROLL NUMBER: 2403A52088

BATCH: 04

Q1)

Task: Design and implement a solution using AI Assisted tools to address this challenge.

Prompt: In the domain of environmental monitoring, a company is facing a challenge related to algorithms with AI Assistance.

Code:

```
1 import numpy as np
2 from datetime import datetime
3
4 class Detector:
5     def __init__(self, threshold=2.5):
6         self.threshold, self.mean, self.std = threshold, None, None
7     def fit(self, data):
8         self.mean, self.std = np.mean(data), np.std(data)
9     def detect(self, data):
10        return np.where(np.abs((data - self.mean) / (self.std + 1e-8)) > self.threshold)[0]
11
12 class Predictor:
13     def __init__(self, lookback=10):
14         self.lookback, self.w, self.b = lookback, None, None
15     def train(self, data):
16         X = np.array([data[i:i+self.lookback] for i in range(len(data)-self.lookback)])
17         y, X_flat = data[self.lookback:], X.reshape(X.shape[0], -1)
18         self.w, self.b = np.zeros(X_flat.shape[1]), 0
19         for _ in range(100):
20             err = X_flat @ self.w + self.b - y
21             self.w -= 0.01 * X_flat.T @ err / len(err)
22             self.b -= 0.01 * np.sum(err) / len(err)
23     def predict_next(self, last_vals):
24         return float(last_vals.flatten() @ self.w + self.b) if self.w is not None else np.mean(last_vals)
25
26 class Monitor:
27     def __init__(self, num_sensors=5):
28         self.detector, self.predictor = Detector(), Predictor()
29         self.data, self.num_sensors = np.random.normal(25, 3, (100, num_sensors)), num_sensors
30
31     def report(self):
32         print("-" * 60)
33         print("ENVIRONMENTAL MONITORING REPORT")
34         print("-" * 60)
35
36         variances = np.var(self.data, axis=0)
37         importance = variances / np.sum(variances)
38         allocation = importance * 100
39         total_anomalies = 0
```

```
41 for i in range(self.num_sensors):
42     sensor_data = self.data[:, i]
43     self.detector.fit(sensor_data[:70])
44     anomalies = len(self.detector.detect(sensor_data))
45     total_anomalies += anomalies
46
47     self.predictor.train(sensor_data)
48     forecast = [self.predictor.predict_next(sensor_data[-self.predictor.lookback: ]) for _ in range(3)]
49
50     print(f"Sensor {i}: Val={sensor_data[-1]:.1f}°C | Mean={np.mean(sensor_data):.1f}°C | Anomalies={anomalies} | Forecast=[{f'({v:.0f})' for v in forecast}]")
51
52 critical = np.argsort(importance)[-3].tolist()
53 print(f"\nPower (W): {[f'(p:{if_})' for p in allocation]}")
54 print(f"Critical: {critical} | Total Anomalies: {total_anomalies}")
55 print(f"Status: ('OPTIMAL' if total_anomalies == 0 else 'WARNING')")
56 print("-" * 60)
57
58 if __name__ == "__main__":
59     Monitor(num_sensors=5).report()
```

Output:

Observation: When tasked AI with a task to generate a solution for the challenges related to algorithms with AI assistance in the domain of Environmental Monitoring, it swiftly generated the code for the given problems.

Q2)

Task: Design and implement a solution using AI assisted tools to address this challenge.

Prompt: In the domain of Education, a company is facing a challenge related to algorithms with AI assistance. Generate a code to address this challenge and code a solution for this.

Code:

```
# question_2.py > ...
1  import random
2
3  class Student:
4      def __init__(self, name, scores):
5          self.name = name
6          self.scores = scores
7
8      def average_score(self):
9          return sum(self.scores) / len(self.scores)
10
11 class AIAlgorithm:
12     def __init__(self, students):
13         self.students = students
14
15     def recommend_resources(self):
16         recommendations = {}
17         for student in self.students:
18             avg_score = student.average_score()
19             if avg_score < 60:
20                 recommendations[student.name] = "Basic resources"
21             elif avg_score < 80:
22                 recommendations[student.name] = "Intermediate resources"
23             else:
24                 recommendations[student.name] = "Advanced resources"
25         return recommendations
26
27 class EnhancedAIAlgorithm(AIAlgorithm):
28     def personalized_learning_plan(self):
29         plans = {}
30         for student in self.students:
31             avg_score = student.average_score()
32             if avg_score < 60:
33                 plans[student.name] = "Focus on basics and practice tests"
34             elif avg_score < 80:
35                 plans[student.name] = "Review intermediate topics and take quizzes"
36             else:
37                 plans[student.name] = "Explore advanced topics and engage in projects"
38
39         return plans
```

```

40     # Sample data
41     students = [
42         Student("Alice", [55, 60, 65]),
43         Student("Bob", [75, 80, 85]),
44         Student("Charlie", [90, 95, 100])
45     ]
46
47     # Using the AI algorithm
48     ai_algorithm = EnhancedAIAlgorithm(students)
49     resource_recommendations = ai_algorithm.recommend_resources()
50     learning_plans = ai_algorithm.personalized_learning_plan()
51
52     # Output results
53     print("Resource Recommendations:")
54     for student, resource in resource_recommendations.items():
55         print(f"{student}: {resource}")
56
57     print("\nPersonalized Learning Plans:")
58     for student, plan in learning_plans.items():
59         print(f"{student}: {plan}")
60

```

Output:

PS D:\Anas\2nd Year\AIAC\Lab Test\3> & "C:/Users/Anas Siddiqui/Desktop/3/question_2.py"

Resource Recommendations:

Alice: Intermediate resources

Bob: Advanced resources

Charlie: Advanced resources

Personalized Learning Plans:

Alice: Review intermediate topics and take quizzes

Bob: Explore advanced topics and engage in projects

Charlie: Explore advanced topics and engage in projects

PS D:\Anas\2nd Year\AIAC\Lab Test\3>

Observation: When tasked github copilot with a task to address the challenge related to algorithms with AI assistance, it quickly generated the problem and solution code for it.