Code 14

November 30, 2023

[ ]:

```
pip install scikit-learn matplotlib

import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Read data from Week14 File
file_path = 'Week14Assignment.txt'
try:
    with open(file_path, 'r') as file:
        lines = file.readlines()
except FileNotFoundError:
    print(f"Error: File '{file_path}' not found.")
    exit()

# Initializing lists for storing data
patient_ids = []
readmissions = []
satisfaction_scores = {'Staff': [], 'Cleanliness': [], 'Food': [], 'Comfort':␣
 ↪[], 'Communication': []}

# Parsing data from the lines
for line in lines:
    data = line.strip().split(',')
    patient_ids.append(int(data[0]))
    readmissions.append(int(data[1]))
    for i, category in enumerate(['Staff', 'Cleanliness', 'Food', 'Comfort',␣
 ↪'Communication']):
        satisfaction_scores[category].append(int(data[i + 2]))

# Calculating and displaying statistics
num_readmitted = sum(readmissions)
avg_satisfaction_scores = {category: np.mean(scores) for category, scores in␣
 ↪satisfaction_scores.items()}
```

```python
print(f"Number of Patients Readmitted: {num_readmitted}\n")
for category, score in avg_satisfaction_scores.items():
    print(f"Average {category} Satisfaction: {score:.2f}")

# Data for logistic regression
X = np.array(list(zip(*[satisfaction_scores[category] for category in satisfaction_scores])))
y = np.array(readmissions)

# Splitting data for training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Training the logistic regression
model = LogisticRegression()
model.fit(X_train, y_train)

# Predictions for the test
y_pred = model.predict(X_test)

# Calculating the accuracy
accuracy = accuracy_score(y_test, y_pred)

# Displaying logistic regression results
print("\nLogistic Regression Results:\n" + "-" * 28)
print(f"Accuracy of the Logistic Regression Model: {accuracy:.2%}")
correlation_strength = "Weak correlation" if accuracy < 0.7 else "Strong correlation"
print(f"Correlation between Overall Satisfaction Scores and Readmission: {correlation_strength}")

# Plotting the data points with the logistic regression curve
plt.scatter(X[:, 0], y, color='blue', label='Original Data')
plt.scatter(X_test[:, 0], y_pred, color='red', marker='o', label='Predicted Data')
plt.xlabel('Staff Satisfaction')
plt.ylabel('Readmission')
plt.legend()
plt.show()
```