詢問 GPT  
以下資料  
1.請他比較 logistic regression 還有 SVM

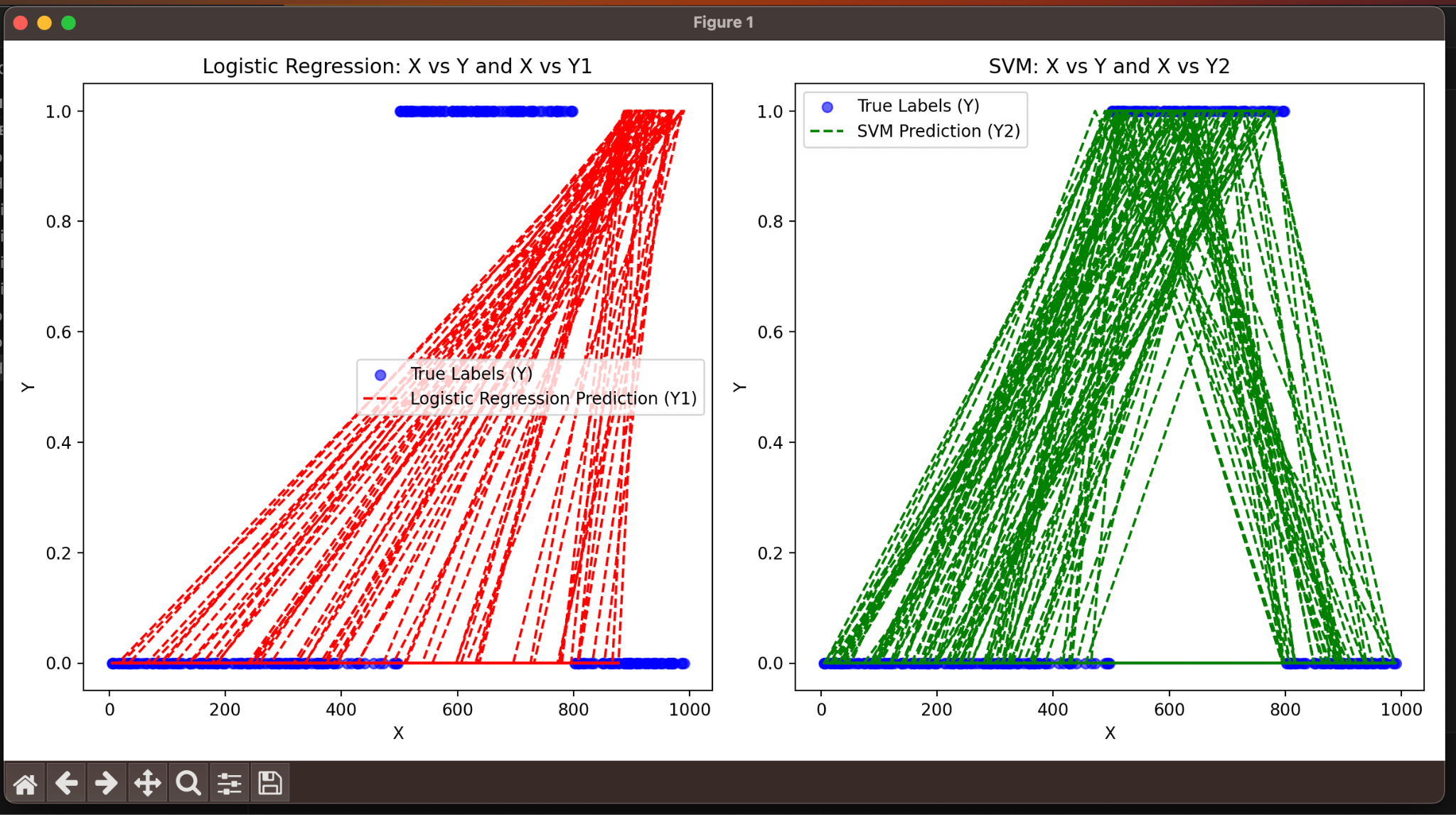
1. Genrate 300 random variable X(i) in 0 to 1000

2. Y(i)=1 IF 500< X(i)< 800, ELSE Y(I)=0

3. Write Logistic regression and output PREDICT RESULT y1, SVM output predict result y2

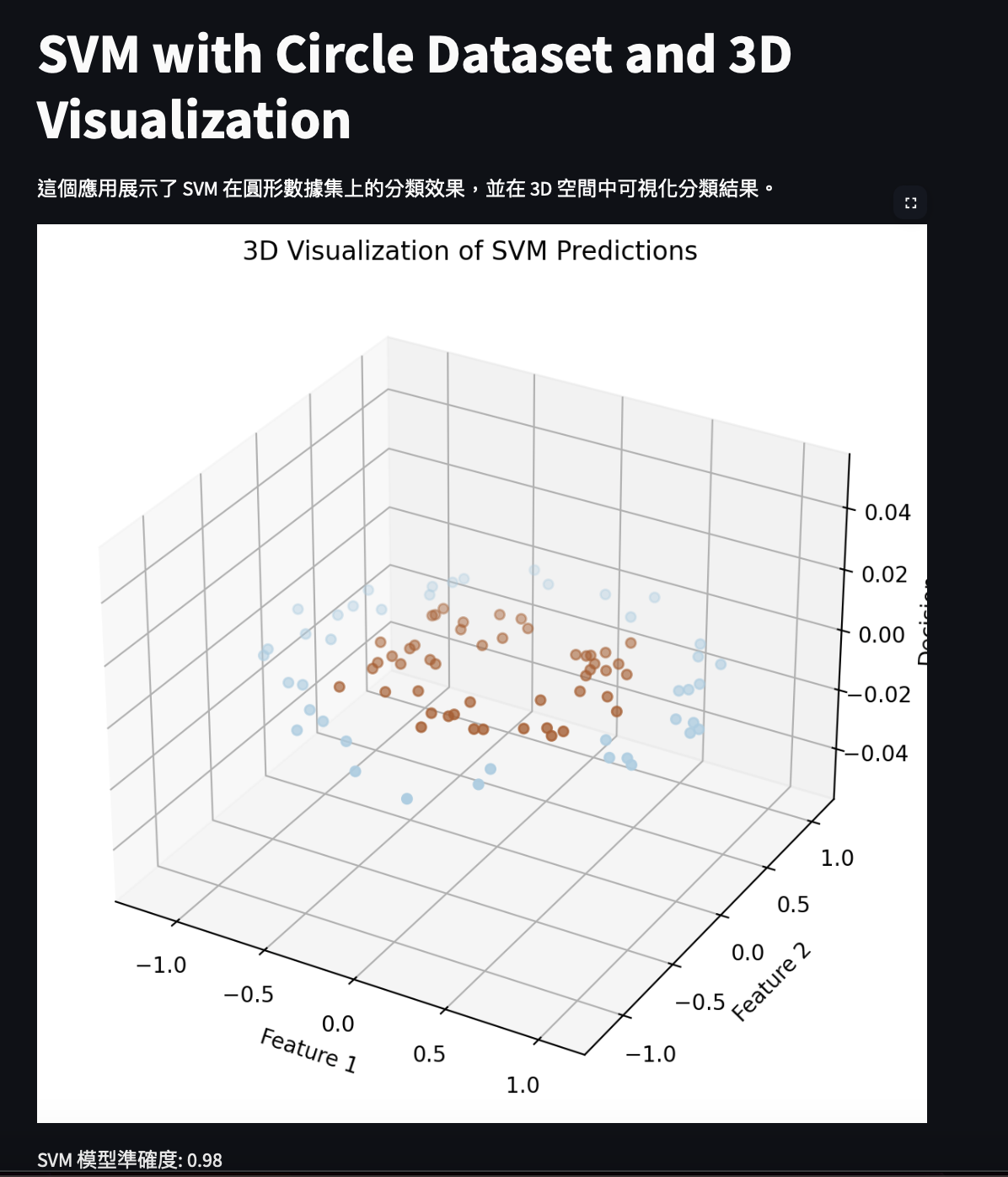
4. draw X, Y and X,Y1 in one figure , and draw X, Y and X,Y2 in another

| import numpy as np  import matplotlib.pyplot as plt  from sklearn.linear\_model import LogisticRegression  from sklearn.svm import SVC  from sklearn.model\_selection import train\_test\_split  # Step 1: Generate 300 random values for X(i) in the range [0, 1000]  np.random.seed(42) # For reproducibility  X = np.random.uniform(0, 1000, 300).reshape(-1, 1)  # Step 2: Define Y(i) based on the condition 500 < X(i) < 800  Y = np.where((X > 500) & (X < 800), 1, 0).reshape(-1)  # Step 3: Train Logistic Regression and SVM models  # Split data into training and testing sets  X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=42)  # Logistic Regression  log\_reg = LogisticRegression()  log\_reg.fit(X\_train, Y\_train)  y1 = log\_reg.predict(X)  # Support Vector Machine  svm = SVC()  svm.fit(X\_train, Y\_train)  y2 = svm.predict(X)  # Step 4: Plotting the results  # Plot 1: X vs Y and X vs Y1 (Logistic Regression)  plt.figure(figsize=(12, 6))  # Plot X vs Y (True labels)  plt.subplot(1, 2, 1)  plt.scatter(X, Y, color='blue', label='True Labels (Y)', alpha=0.6)  plt.plot(X, y1, color='red', label='Logistic Regression Prediction (Y1)', linestyle='--')  plt.xlabel('X')  plt.ylabel('Y')  plt.title('Logistic Regression: X vs Y and X vs Y1')  plt.legend()  # Plot 2: X vs Y and X vs Y2 (SVM)  plt.subplot(1, 2, 2)  plt.scatter(X, Y, color='blue', label='True Labels (Y)', alpha=0.6)  plt.plot(X, y2, color='green', label='SVM Prediction (Y2)', linestyle='--')  plt.xlabel('X')  plt.ylabel('Y')  plt.title('SVM: X vs Y and X vs Y2')  plt.legend()  plt.tight\_layout()  plt.show() |
| --- |



2.使用 SVC 來處理 2D 數據集，並在 Streamlit 中展示分類結果

| import streamlit as st  import numpy as np  import matplotlib.pyplot as plt  from sklearn.svm import SVC  from sklearn.datasets import make\_circles  from sklearn.model\_selection import train\_test\_split  from mpl\_toolkits.mplot3d import Axes3D  # 生成圓形分佈的數據  X, y = make\_circles(n\_samples=300, factor=0.5, noise=0.1)  # 分割數據集  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)  # 使用 SVM 進行分類  svm = SVC(kernel='rbf') # 使用RBF核函數，對於圓形分佈的數據來說，這是一個合適的選擇  svm.fit(X\_train, y\_train)  # 預測  y\_pred = svm.predict(X\_test)  # Streamlit UI  st.title("SVM with Circle Dataset and 3D Visualization")  st.write("這個應用展示了 SVM 在圓形數據集上的分類效果，並在 3D 空間中可視化分類結果。")  # 顯示原始數據與預測結果  fig = plt.figure(figsize=(10, 7))  ax = fig.add\_subplot(111, projection='3d')  # 為了方便視覺化，我們將所有數據點的 z 軸設置為0，這樣就可以在 3D 中顯示它們  z = np.zeros(X\_test.shape[0]) # 設置 z 為零，所有資料點都在同一個平面  # 繪製 SVM 預測結果的 3D 散點圖  ax.scatter(X\_test[:, 0], X\_test[:, 1], z, c=y\_pred, cmap=plt.cm.Paired)  # 設定 3D 圖形的標題和坐標軸標籤  ax.set\_title("3D Visualization of SVM Predictions")  ax.set\_xlabel("Feature 1")  ax.set\_ylabel("Feature 2")  ax.set\_zlabel("Decision")  # 在 Streamlit 中顯示圖像  st.pyplot(fig)  # 顯示 SVM 模型的準確度  accuracy = svm.score(X\_test, y\_test)  st.write(f"SVM 模型準確度: {accuracy:.2f}") |
| --- |



HW3-3. 2D dataset 分布在feature plane上非圓形

| import streamlit as st  import numpy as np  import matplotlib.pyplot as plt  from sklearn.svm import SVC  from sklearn.datasets import make\_moons # 用於生成月亮形狀的數據  from sklearn.model\_selection import train\_test\_split  # 生成月亮形狀的數據  X, y = make\_moons(n\_samples=300, noise=0.1)  # 分割數據集  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)  # 使用 SVM 進行分類  svm = SVC(kernel='rbf') # 使用RBF核函數，這對於非線性數據有很好的分類效果  svm.fit(X\_train, y\_train)  # 預測  y\_pred = svm.predict(X\_test)  # Streamlit UI  st.title("SVM with Moon Dataset and 2D Visualization")  st.write("這個應用展示了 SVM 在月亮形狀數據集上的分類效果，並在 2D 平面中可視化分類結果。")  # 顯示原始數據與預測結果的 2D 圖  fig, ax = plt.subplots(figsize=(10, 7))  # 繪製 SVM 預測結果的 2D 散點圖  scatter = ax.scatter(X\_test[:, 0], X\_test[:, 1], c=y\_pred, cmap=plt.cm.Paired)  # 設定 2D 圖形的標題和坐標軸標籤  ax.set\_title("2D Visualization of SVM Predictions")  ax.set\_xlabel("Feature 1")  ax.set\_ylabel("Feature 2")  # 在 Streamlit 中顯示圖像  st.pyplot(fig)  # 顯示 SVM 模型的準確度  accuracy = svm.score(X\_test, y\_test)  st.write(f"SVM 模型準確度: {accuracy:.2f}") |
| --- |

