After prediction

"Based on the patterns and relationships our prediction model has learned from past applications, **this application shares similarities with other declined applications**. This is why we recommend **declining** your application at this time."

【However, it is also essential to provide more specific feedback or reasons behind the decision. Although SVMs do not directly provide feature importance, we can use the **weights of a linear SVM model** or other techniques to gain insights into the most important features. Once you have this information, you can tailor the feedback to the user based on the factors that most influenced the decision. 】

算weights:

"Some of the factors that contributed to this decision include a credit score, debt-to-income ratio, and recent late payments on other credit accounts (weights高的factors). We recommend improving these factors and reapplying in the future when your financial situation has improved."

Weights 部分：

1. obtain the weights (coefficients) associated with each feature from our svm model
2. Identify the top features that influenced the decision. Calculate the absolute values of the weights and sort them in descending order. Pick the top 5 features that have the largest absolute weights.
3. For each of the top 5 features, determine whether the feature value in the user's application is above or below the threshold that separates accepted and declined applications. We use the average of the feature in the accepted applications as a reference.
4. For each of the top 5 features, we prepare an explanation highlighting how the user's application deviates from the reference. Focus on the direction of the deviation (above or below) and its possible impact on the decision.
5. Combine the explanations for each of the top 5 features into a coherent message, and present it to the user.

Code:

pip install scikit-learn

import numpy as np

from sklearn import datasets

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.svm import SVC

# Load a dataset (e.g., the Iris dataset)

data = datasets.load\_iris()

X = data.data

y = data.target

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Standardize the features

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Create a linear SVM model

svm\_linear = SVC(kernel='linear')

# Train the model on the training data

svm\_linear.fit(X\_train, y\_train)

# Get the weights (coefficients) of the linear SVM model

weights = svm\_linear.coef\_

# Print the weights

print("Weights (Coefficients) of the linear SVM model:\n", weights)