ML-Exp 6: Classification of Credit Card Default Risk using Support Vector Machine.

Student Name: Prewitt Gomes

Roll No.: 22 Batch: 1 Date: 25-02-2026

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score, accuracy_score
```

```
data=pd.read_csv("credit.csv", skiprows=1)
data=data.sample(5000, random_state=42)

print("--- Column Names ---")
print(data.columns)
print("\n--- Data Info (Types & Nulls) ---")
data.info()
print("\n--- Statistical Summary ---")
print(data.describe())
print("\n--- First 5 Rows ---")
print(data.head())
print("\n--- Last 5 Rows ---")
print(data.tail())
print("\n--- Dataset Dimensions (Rows, Cols) ---")
print(data.shape)
```

```
35384       0       0             53198      43181      31473      2000      2000
```

```
25284      0      0  ...      52198      43181      31473      2000      2000
18355     -2     -2  ...        500       2057      23322      1053         0
27684      0      0  ...      53216      49194      48487      2000      2600
4110       0      0  ...       4173       4408       5846      1522      3010

       PAY_AMT3  PAY_AMT4  PAY_AMT5  PAY_AMT6  default payment next month
6778      10009     22000     20000     15000                           0
25284     10052      2000      1136      2000                           0
18355       500      2057     23322      4299                           0
27684      4241      1700      2500      1500                           0
4110       1500      1500      1500      2000                           0

[5 rows x 25 columns]

--- Dataset Dimensions (Rows, Cols) ---
(5000, 25)
```

```
data.drop(columns=['ID'], inplace=True)
print(data.columns)
```

```
Index(['LIMIT_BAL', 'GENDER', 'EDUCATION', 'MARRIAGE', 'AGE', 'PAY_0', 'PAY_2',
       'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6', 'BILL_AMT1', 'BILL_AMT2',
       'BILL_AMT3', 'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1',
       'PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6',
       'default payment next month'],
      dtype='object')
```

```
data.isnull().sum()
```

	0
LIMIT_BAL	0
GENDER	0
EDUCATION	0
MARRIAGE	0
AGE	0
PAY_0	0
PAY_2	0
PAY_3	0
PAY_4	0
PAY_5	0
PAY_6	0
BILL_AMT1	0
BILL_AMT2	0
BILL_AMT3	0
BILL_AMT4	0
BILL_AMT5	0
BILL_AMT6	0
PAY_AMT1	0
PAY_AMT2	0
PAY_AMT3	0
PAY_AMT4	0
PAY_AMT5	0
PAY_AMT6	0
default payment next month	0

dtype: int64

```
print(data.dtypes)
```

```
LIMIT_BAL                     int64
GENDER                        int64
EDUCATION                     int64
```

```
MARRIAGE                      int64
AGE                           int64
PAY_0                         int64
PAY_2                         int64
PAY_3                         int64
PAY_4                         int64
PAY_5                         int64
PAY_6                         int64
BILL_AMT1                     int64
BILL_AMT2                     int64
BILL_AMT3                     int64
BILL_AMT4                     int64
BILL_AMT5                     int64
BILL_AMT6                     int64
PAY_AMT1                      int64
PAY_AMT2                      int64
PAY_AMT3                      int64
PAY_AMT4                      int64
PAY_AMT5                      int64
PAY_AMT6                      int64
default payment next month    int64
dtype: object
```

```
y=data['default payment next month']
x=data[['BILL_AMT1', 'BILL_AMT2']]

print(data.dtypes)
print(data.head())
```

```
LIMIT_BAL                     int64
GENDER                        int64
EDUCATION                     int64
MARRIAGE                      int64
AGE                           int64
PAY_0                         int64
PAY_2                         int64
PAY_3                         int64
PAY_4                         int64
PAY_5                         int64
PAY_6                         int64
BILL_AMT1                     int64
BILL_AMT2                     int64
BILL_AMT3                     int64
BILL_AMT4                     int64
BILL_AMT5                     int64
BILL_AMT6                     int64
PAY_AMT1                      int64
PAY_AMT2                      int64
PAY_AMT3                      int64
PAY_AMT4                      int64
PAY_AMT5                      int64
PAY_AMT6                      int64
default payment next month    int64
dtype: object
       LIMIT_BAL  GENDER  EDUCATION  MARRIAGE  AGE  PAY_0  PAY_2  PAY_3  \
2308       30000       1          2         2   25      0      0      0
22404     150000       2          1         2   26      0      0      0
23397      70000       2          3         1   32      0      0      0
25058     130000       1          3         2   49      0      0      0
2664       50000       2          2         2   36      0      0      0

       PAY_4  PAY_5  ...  BILL_AMT4  BILL_AMT5  BILL_AMT6  PAY_AMT1  PAY_AMT2  \
2308       0      0  ...      12580      13716      14828      1500      2000
22404      0      0  ...     101581      77741      77264      4486      4235
23397      0      0  ...      69753      70111      70212      2431      3112
25058      0      0  ...      16898      11236       6944      1610      1808
2664       0      0  ...      19574      20295      19439      2000      1500

       PAY_AMT3  PAY_AMT4  PAY_AMT5  PAY_AMT6  default payment next month
2308       1500      1500      1500      2000                           0
22404      3161      2647      2669      2669                           0
23397      3000      2438      2500      2554                           0
25058      7014        27      7011      4408                           0
2664       1000      1800         0      1000                           1

[5 rows x 24 columns]
```

```
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.3, random_state=42)
```

```
scaler=StandardScaler()
x_train=scaler.fit_transform(x_train)
```

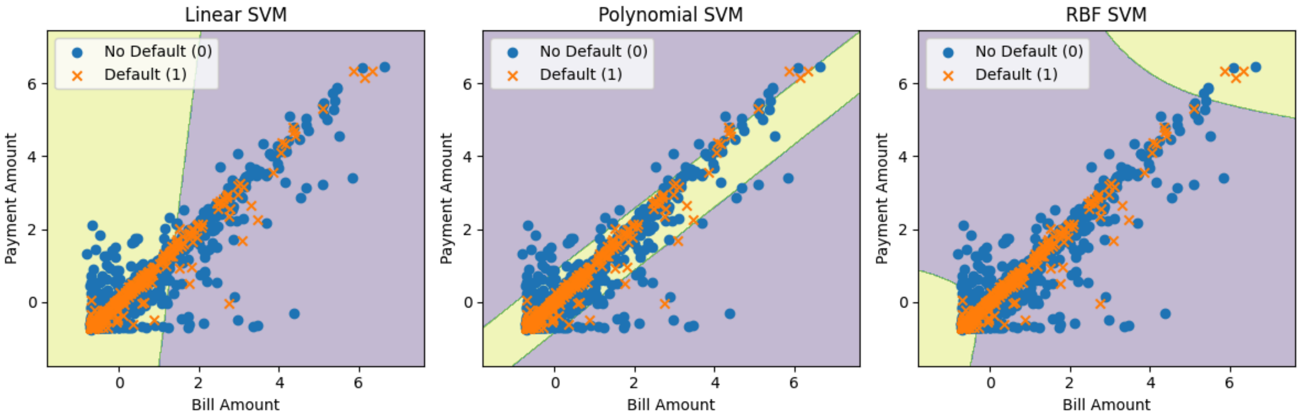
```
x_test=scaler.transform(x_test)
```

```
models = {
    "Linear SVM": SVC(kernel='linear', C=1, class_weight='balanced'),
    "Polynomial SVM": SVC(kernel='poly', degree=2, C=1, gamma='scale', class_weight='balanced'),
    "RBF SVM": SVC(kernel='rbf', C=5, gamma=0.1, class_weight='balanced')
}

for name, model in models.items():
    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)

    print("\n", name)
    print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
    print("Precision:", precision_score(y_test, y_pred, zero_division=0))
    print("Recall:", recall_score(y_test, y_pred))
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("F1 Score:", f1_score(y_test, y_pred))
```

```
 Linear SVM
Confusion Matrix:
 [[ 103 1056]
 [  34  307]]
Precision: 0.2252384446074835
Recall: 0.9002932551319648
Accuracy: 0.2733333333333333
F1 Score: 0.36032863849765256

 Polynomial SVM
Confusion Matrix:
 [[  35 1124]
 [   8  333]]
Precision: 0.22855181880576528
Recall: 0.9765395894428153
Accuracy: 0.24533333333333332
F1 Score: 0.3704115684093437

 RBF SVM
Confusion Matrix:
 [[560 599]
 [166 175]]
Precision: 0.22609819121447028
Recall: 0.5131964809384164
Accuracy: 0.49
F1 Score: 0.31390134529147984
```

```
def plot_boundary(model, title):
    h = 0.02
    x_min, x_max = x_train[:, 0].min() - 1, x_train[:, 0].max() + 1
    y_min, y_max = x_train[:, 1].min() - 1, x_train[:, 1].max() + 1

    xx, yy = np.meshgrid(
        np.arange(x_min, x_max, h),
        np.arange(y_min, y_max, h)
    )

    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, alpha=0.3)

    plt.scatter(x_train[y_train == 0, 0],
                x_train[y_train == 0, 1],
                label='No Default (0)',
                marker='o')

    plt.scatter(x_train[y_train == 1, 0],
                x_train[y_train == 1, 1],
                label='Default (1)',
                marker='x')

    plt.legend()
    plt.title(title)
    plt.xlabel("Bill Amount")
    plt.ylabel("Payment Amount")

plt.figure(figsize=(12, 4))
```

```
for i, (name, model) in enumerate(models.items()):
    plt.subplot(1,3,i+1)
    model.fit(x_train, y_train)
    plot_boundary(model, name)

plt.tight_layout()
plt.show()
```


Linear SVM
Polynomial SVM
RBF SVM
● No Default (0)
× Default (1)
Payment Amount
Bill Amount