Name: Mariam Badure

Date: 25/02/2026

Subject: ML

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

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [3]:
```
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_sco
```

In [4]:
```
data = pd.read_csv("credit.csv",skiprows = 1)
data = data.sample(5000, random_state = 42)

print(data.columns)

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

```
Index(['ID', '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')
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')
```

In [5]:
```
data.tail()
```

Out[5]:

	LIMIT_BAL	GENDER	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4
6778	180000	1	1	2	27	0	0	0	0
25284	50000	2	2	2	22	0	0	0	0
18355	200000	2	1	2	24	-2	-2	-2	-2
27684	120000	2	2	2	24	0	0	0	0
4110	90000	2	2	1	29	0	0	0	0

5 rows × 24 columns

In [6]:
```
data.head()
```

Out[6]:

	LIMIT_BAL	GENDER	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4
2308	30000	1	2	2	25	0	0	0	0
22404	150000	2	1	2	26	0	0	0	0
23397	70000	2	3	1	32	0	0	0	0
25058	130000	1	3	2	49	0	0	0	0
2664	50000	2	2	2	36	0	0	0	0

5 rows × 24 columns

In [7]:
```
data.isnull()
```

Out[7]:

	LIMIT_BAL	GENDER	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4
2308	False	False	False	False	False	False	False	False	False
22404	False	False	False	False	False	False	False	False	False
23397	False	False	False	False	False	False	False	False	False
25058	False	False	False	False	False	False	False	False	False
2664	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
6778	False	False	False	False	False	False	False	False	False
25284	False	False	False	False	False	False	False	False	False
18355	False	False	False	False	False	False	False	False	False
27684	False	False	False	False	False	False	False	False	False
4110	False	False	False	False	False	False	False	False	False

5000 rows × 24 columns

In [8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5000 entries, 2308 to 4110
Data columns (total 24 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   LIMIT_BAL                   5000 non-null   int64
 1   GENDER                      5000 non-null   int64
 2   EDUCATION                   5000 non-null   int64
 3   MARRIAGE                    5000 non-null   int64
 4   AGE                         5000 non-null   int64
 5   PAY_0                       5000 non-null   int64
 6   PAY_2                       5000 non-null   int64
 7   PAY_3                       5000 non-null   int64
 8   PAY_4                       5000 non-null   int64
 9   PAY_5                       5000 non-null   int64
 10  PAY_6                       5000 non-null   int64
 11  BILL_AMT1                   5000 non-null   int64
 12  BILL_AMT2                   5000 non-null   int64
 13  BILL_AMT3                   5000 non-null   int64
 14  BILL_AMT4                   5000 non-null   int64
 15  BILL_AMT5                   5000 non-null   int64
 16  BILL_AMT6                   5000 non-null   int64
 17  PAY_AMT1                    5000 non-null   int64
 18  PAY_AMT2                    5000 non-null   int64
 19  PAY_AMT3                    5000 non-null   int64
 20  PAY_AMT4                    5000 non-null   int64
 21  PAY_AMT5                    5000 non-null   int64
 22  PAY_AMT6                    5000 non-null   int64
 23  default payment next month  5000 non-null   int64
dtypes: int64(24)
memory usage: 976.6 KB
```

In [9]:
```
y = data['default payment next month']
x = data[['BILL_AMT1', 'BILL_AMT2']]
print(x.dtypes)
print(x.head())
```

```
BILL_AMT1    int64
BILL_AMT2    int64
dtype: object
       BILL_AMT1  BILL_AMT2
2308        8864      10062
22404     136736     125651
23397      70122      69080
25058      20678      18956
2664       94228      47635
```

In [10]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_s
```

In [11]:
```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

In [12]:
```
models = {
    "Linear SVM" : SVC(kernel = 'linear', C = 1, class_weight = 'balanced'),
    "Polynomial SVM" : SVC(kernel = 'poly', degree = 2, C = 1, gamma = 'scale', cla
```

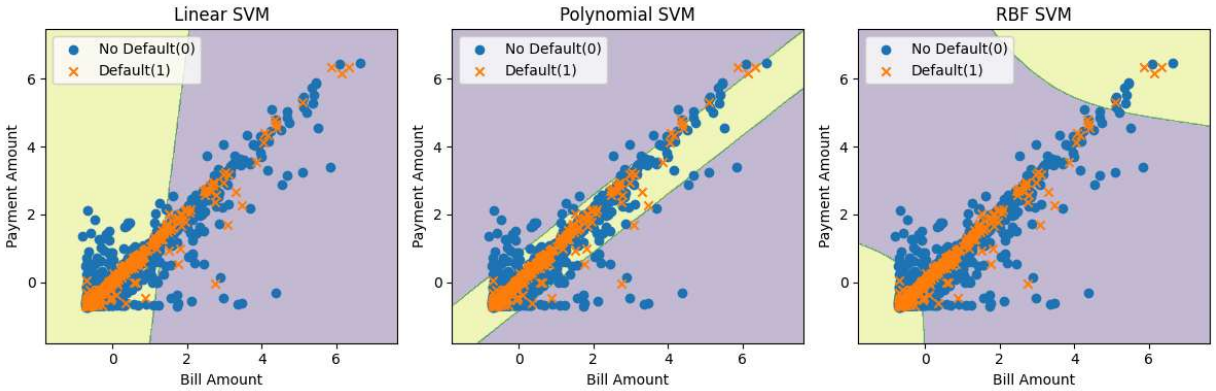
```
    "RBF SVM" : SVC(kernel = 'rbf', C = 1, gamma = 0.1, class_weight = 'balanced')
}
```

In [13]:
```
for name, model in models.items():
    model.fit(x_train, y_train)
    y_pred_test = model.predict(x_test)
    y_pred_train = model.predict(x_train)

    print('\n', name)
    print("Confusion Matrix:\n", confusion_matrix(y_test,y_pred_test))
    print("Precision:", precision_score(y_test,y_pred_test, zero_division = 0))
    print("Recall:", recall_score(y_test,y_pred_test))
    print("F-1:", f1_score(y_test,y_pred_test))
    print("Accuracy Score Test:", accuracy_score(y_test, y_pred_test))
    print("Accuracy Score Train:", accuracy_score(y_pred_train,y_train))
```

```
 Linear SVM
Confusion Matrix:
 [[ 103 1056]
 [  34  307]]
Precision: 0.2252384446074835
Recall: 0.9002932551319648
F-1: 0.36032863849765256
Accuracy Score Test: 0.2733333333333333
Accuracy Score Train: 0.2797142857142857

 Polynomial SVM
Confusion Matrix:
 [[  35 1124]
 [   8  333]]
Precision: 0.22855181880576528
Recall: 0.9765395894428153
F-1: 0.3704115684093437
Accuracy Score Test: 0.24533333333333332
Accuracy Score Train: 0.24114285714285713

 RBF SVM
Confusion Matrix:
 [[499 660]
 [138 203]]
Precision: 0.23522595596755505
Recall: 0.5953079178885631
F-1: 0.3372093023255814
Accuracy Score Test: 0.468
Accuracy Score Train: 0.464
```

In [14]:
```
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[:, 0].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 Defa
    plt.scatter(x_train[y_train == 1, 0], x_train[y_train== 1, 1], label = 'Default
    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()
```



In [ ]: