Machine Learning
Experiment 7

SAP ID: 60004200107 Name: Kartik Jolapara
Division: B Batch: B1

AIM
To implement SVM.

THEORY
Support Vector Machines (SVMs) are a type of supervised learning algorithm used for classification, regression, and outlier detection. SVMs are based on the idea of finding a hyperplane that separates the data points in a high-dimensional space with the maximum margin, where the margin is defined as the distance between the hyperplane and the closest data points from each class. SVMs are particularly effective for solving binary classification problems, where the goal is to separate the data into two classes, such as "spam" and "not spam".
The basic idea of SVMs is to transform the input data into a higher dimensional space, where it becomes more separable by a hyperplane. This transformation is done using a kernel function, which maps the input data to a higher dimensional feature space. The kernel function can be chosen based on the type of data and the problem at hand. Some common types of kernel functions include linear, polynomial, and radial basis function (RBF) kernels.
Once the data has been transformed, the SVM algorithm finds the hyperplane that separates the data points with the maximum margin. The hyperplane is defined as the set of all points x in the feature space that satisfy the equation - $w^T x + b = 0$
where w is a vector perpendicular to the hyperplane and b is the intercept. The distance between the hyperplane and the closest data points from each class is given by the margin, which is proportional to the length of the vector w.

The SVM algorithm aims to find the values of w and b that maximize the margin, subject to the constraint that all data points are classified correctly. This is done by solving a quadratic optimization problem, which involves finding the Lagrange multipliers that maximize the margin.
One of the main advantages of SVMs is their ability to handle non-linearly separable data by using kernel functions to transform the input data into a higher dimensional space. SVMs are also robust to overfitting and can generalize well to new data. However, SVMs can be sensitive to the choice of kernel function and its parameters, and the optimization problem can become computationally expensive for large datasets.

CODE AND OUTPUT

```
from sklearn.datasets import load_breast_cancer from
sklearn.model_selection import train_test_split from sklearn.svm
import SVC from sklearn.metrics import
```

```
confusion_matrix,classification_report import seaborn as sns
import matplotlib.pyplot as plt import pandas as pd

data =
pd.read_csv("UniversalBank.csv") X =
data.iloc[:, :-1] y = data.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
svm_model = SVC(kernel='linear', C=1.0, random_state=0) svm_model.fit(X_train,
y_train)
```

```
SVC
SVC(kernel='linear', random_state=0)
```

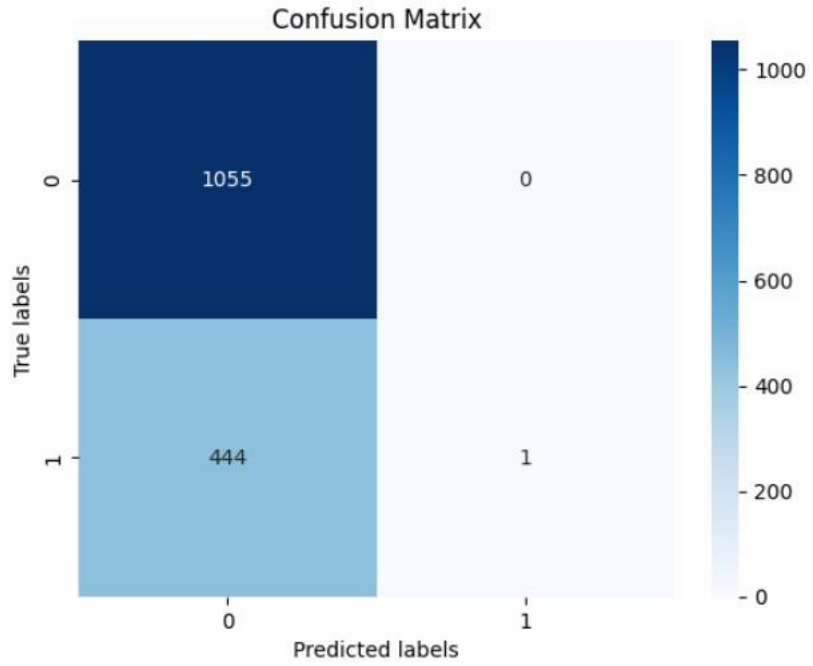
```
from sklearn.metrics import accuracy_score
y_pred = svm_model.predict(X_test) accuracy
= accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)
```

```
Accuracy: 0.704
```

```
cm = confusion_matrix(y_test, y_pred)
print(f"Confusion Matrix : ") print(cm)
```

```
Confusion Matrix :
[[1055    0]
 [ 444    1]]
```

```
sns.heatmap(cm, annot=True, cmap='Blues', fmt='g')
plt.xlabel('Predicted labels') plt.ylabel('True labels')
plt.title('Confusion Matrix')
plt.show()
```



from sklearn.metrics import confusion_matrix,classification_report print("Classification report")
print(classification_report(y_test,y_pred))

```
Classification report
              precision    recall  f1-score   support

           0       0.70      1.00      0.83      1055
           1       1.00      0.00      0.00       445

    accuracy                           0.70      1500
   macro avg       0.85      0.50      0.42      1500
weighted avg       0.79      0.70      0.58      1500
```

CONCLUSION

Hence, we have successfully implemented SVM.