Practical 3

Problem Statement:

Implementation of Support Vector Machines (SVM) for classifying images of hand written digits into their respective numerical classes (0 to 9)

Objective:

1. Understand working of SVM algorithm
2. Evaluate the SVM models based on accuracy

Outcome:

Support Vector machine for prediction of test data should be understood and accuracy is evaluated.

Theory:

Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other.

Types of Support Vector Machine Algorithms

1. Linear SVM

When the data is perfectly linearly separable only then we can use Linear SVM. Perfectly linearly separable means that the data points can be classified into 2 classes by using a single straight line(if 2D).

2. Non-Linear SVM

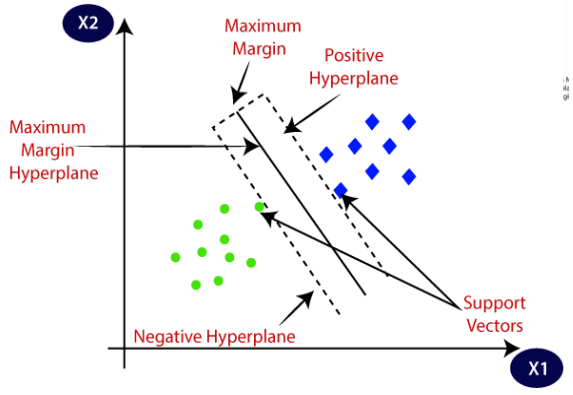
When the data is not linearly separable then we can use Non-Linear SVM, which means when the data points cannot be separated into 2 classes by using a straight line (if 2D) then we use some advanced techniques like kernel tricks to classify them. In most real-world applications we do not find linearly separable datapoints hence we use kernel trick to solve them.

Important Terms

Two main terms in svm:

Support Vectors: These are the points that are closest to the hyperplane. A separating line will be defined with the help of these data points.

Margin: it is the distance between the hyperplane and the observations closest to the hyperplane (support vectors). In SVM large margin is considered a good margin. There are two types of margins hard margin and soft margin. I will talk more about these two in the later section.



Requirements:
1) Get the dataset of images called Digits Dataset

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

2) Importing Libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

Numpy: Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

import numpy as nm

Here we have used nm, which is a short name for Numpy, and it will be used in the whole program.

Matplotlib: The second library is matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code. It will be imported as below:

import matplotlib.pyplot as mpt

Here we have used mpt as a short name for this library.

Pandas: The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:

Here, we have used pd as a short name for this library. Consider the below image

```
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

```

3) Importing the Datasets

Now we need to import the datasets which we have collected for our machine learning project.

```
from sklearn import datasets   ## importing datasets from sklearn
digits = datasets.load_digits()  ### loading data from scikit_learn library
```

4) Data understanding and exploration

Here we understand and visualize missing values, if necessary, remove outliers, and converting data types as needed.

a) Display the data types of each column in the DataFrame

```
numbers.info()
```

b) Describe the distribution of your data

```
numbers.describe(percentiles = [0.05,0.10,0.25,0.50,0.75,0.90,0.99])
```

c) Checking for null values

```
round(100*(numbers.isnull().sum()/(len(numbers.index))),2).sort_values(ascending =
False)
```

d) Check unique entries of label column

```
np.unique(numbers['label'])
numbers['label'].value_counts()
```

e) Visualising the column - label

```
sns.countplot(numbers['label'],palette = 'icefire')
```

5) Data Preparation

a) Set average feature values

```
pd.set_option('display.max_rows', 999)
round(numbers.drop('label', axis=1).mean(), 2).sort_values(ascending = False)
```

b) Splitting into x and y

```
X = numbers.drop("label", axis = 1)
y = numbers['label']
```

c) scaling the features

```
X_scaled = scale(X)
```

d) Splitting out data to train and test using sklearn

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
train_size=0.2,test_size = 0.8, random_sta
```

6) Building SVM model

kernel is the hyperparameter that we want to tune, Linear Kernel is used when the data is Linearly separable, that is, it can be separated using a single Line. It is one of the most common kernels to be used. It is mostly used when there are many Features in a particular Data Set.

```
model_linear = SVC(kernel='linear')
model_linear.fit(X_train, y_train)
```

7) Predicting test data

y_pred = model_linear.predict(X_test)

8) Model Evaluation

To evaluate the performance of the model, we calculate the accuracy of the model using the accuracy score method from the scikit-learn metrics module. The accuracy score measures the percentage of correctly classified data points out of all the data points. The accuracy score is calculated by comparing the predicted labels with the actual labels and then dividing the number of correct predictions by the total number of data points.

a) Calculating the accuracy of the model

```
accuracy = accuracy_score(y_pred, y_test)
```

Algorithm:

Support vector machine

1. Import the dataset
2. Explore the data to figure out what they look like
3. Pre-process the data
4. Split the data into attributes and labels
5. Divide the data into training and testing sets
6. Train the SVM algorithm
7. Make some predictions
8. Evaluate the results of the algorithm

Conclusion:

SVM is a supervised learning algorithm which separates the data into different classes using a hyperplane. The chosen hyperplane is one with the greatest margin between the hyperplane and all points, this yields the greatest likelihood of accurate classification.

Oral Questions:

1. What is a SVM algorithm?
2. Why is SVM the best algorithm?
3. What are the steps of SVM algorithm?
4. What does SVM do in machine learning?