Support Vector Machines (SVM)

A Technical Seminar Report
submitted for fulfilment of
the requirements for the
Degree of Bachelor of Technology
Under Biju Pattnaik University of Technology

BY

JYOTIRMAYEE NAIK **Roll No. 201811500**

2018 - 2022

Under the guidance of

Mr. CH Sree Kumar

NATIONAL INSTITUTE OF SCIENCE & TECHNOLOGY

Palur Hills, Berhampur- 761008, Orissa, India

NATIONAL INSTITUTE OF SCIENCE & TECHNOLOGY

PALUR HILLS, BERHAMPUR, ODISHA - 761008, INDIA

BONAFIDE CERTIFICATE

This is to certify that the Seminar entitled “SUPPORT VECTOR MACHINES (SVM)” is a bonafide record by Jyotirmayee Naik (Roll No. B-TECH 201811500) under my supervision and guidance, in partial fulfilment of the requirements for the award of Degree of Bachelor of technology from National Institute of Science and Technology under Biju Pattnaik University of Technology for the year 2022.

Mr.CH Sree Kumar

(Guide)

Asst. Professor

Dept. of Computer Science & Engineering

ABSTRACT

Support Vector Machine (SVM) has been introduced in the late 1990s and successfully applied to many engineering related applications. Here we made to introduce the SVM, its principles, structures, and parameters. The issue of selecting a kernel function and other associated parameters of SVMs was also raised and applications from different petroleum and mining related tasks were brought to show how those parameters can be properly selected. It seems that the cross-validation approach would be the best technique for parameter selections of SVMs but few other concerns such as running time must not be neglected. Support Vector Machine (SVM)—a machine learning method that has become exceedingly popular for neuroimaging analysis in recent years. Because of their relative simplicity and flexibility for addressing a range of classification problems, SVMs distinctively afford balanced predictive performance, even in studies where sample sizes may be limited. In brain disorders research, SVMs are typically employed using multivoxel pattern analysis (MVPA) because their relative simplicity carries a lower risk of overfitting even using high-dimensional imaging data.

ACKNOWLEDGEMENT

I would like to take this opportunity to thank all those individuals whose invaluable contribution in a direct or indirect manner has gone into the making of this project a tremendous learning experience for me.

It is my proud privilege to epitomize my deepest sense of gratitude and indebtedness to my faculty guide, **Mr. CH Sree Kumar** for his valuable guidance, keen and sustained interest, intuitive ideas and persistent endeavour. His guidance and inspirations enabled me to complete my report work successfully.

I give my sincere thanks **to Ms. Deepika Rani Sahu**, **Seminar Coordinator**, for giving me the opportunity and motivating me to complete the project within stipulated period of time and providing a helping environment.

I acknowledge with immense pleasure the sustained interest, encouraging attitude and constant inspiration rendered by **Prof. (Dr) sukant K. Mohapatra** (Chairman), **Dr Priyadarshi (Piyu) Tripathy** (Principal), **Dr. Hemant Kumar Reddy** (HOD, School of Computer Science) N.I.S.T. and selfless inspiration has always helped us to move ahead.

Jyotirmayee Naik

Roll No: 201811500

CONTENTS

1.INTRODUCTION

SVMs are the most popular algorithm for classification in machine learning algorithms. Their mathematical background is quintessential in building the foundational block for the geometrical distinction between the two classes. We will see how Support vector machines work by observing their implementation in Python and finally, we will look at some of the important applications.

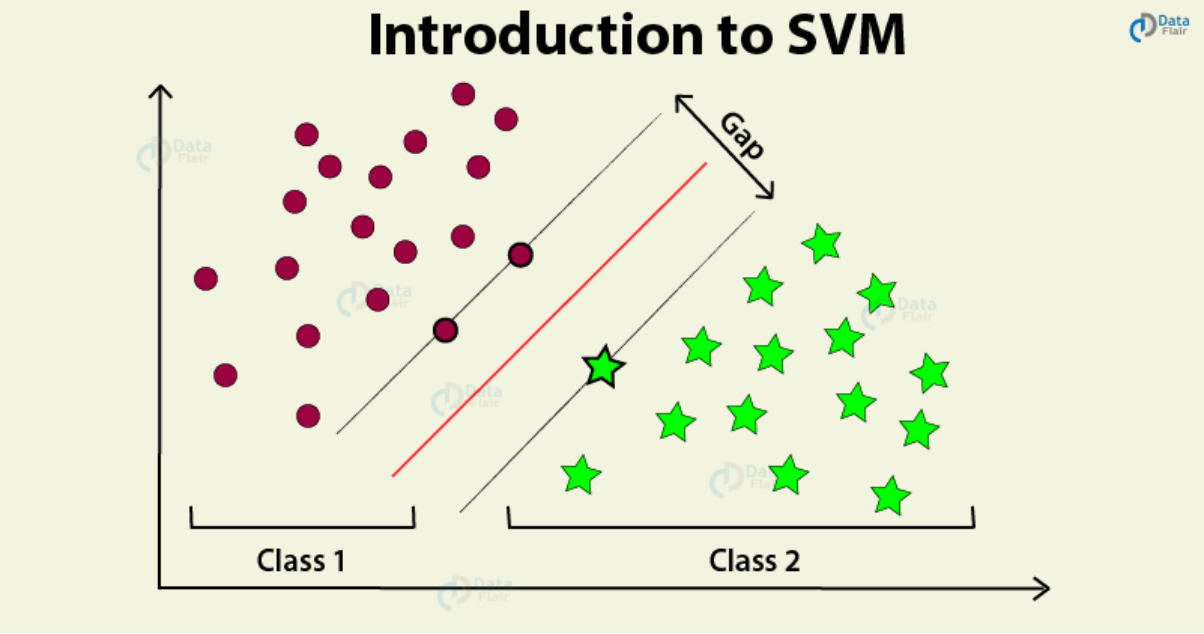


Figure 1: Introduction to SVM

2.What is Support Vector Machine?

"**Support Vector Machine**" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

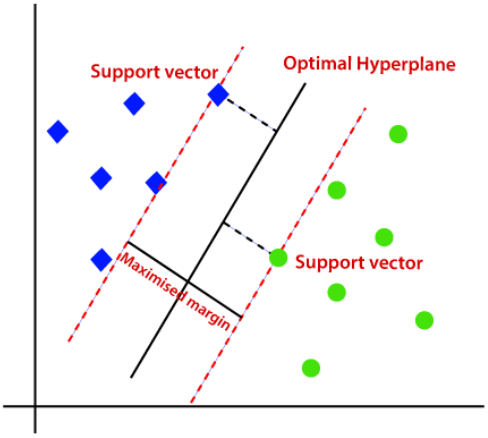


Figure 2: Optimal Hyperplane using the SVM algorithm

2.1. Support-Vectors

Support vectors are the data points that are nearest to the hyper-plane and affect the position and orientation of the hyper-plane. We have to select a hyperplane, for which the margin, i.e., the distance between support vectors and hyper-plane is maximum. Even a little interference in the position of these support vectors can change the hyper-plane.

2.2. Hyper-Plane

A hyperplane is a decision boundary that differentiates the two classes in SVM. A data point falling on either side of the hyperplane can be attributed to different classes. The dimension of the hyperplane depends on the number of input features in the dataset. If we have 2 input features the hyper-plane will be a line. likewise, if the number of features is 3, it will become a two-dimensional plane.

2.3. Maximised margin

SVM's are highly appropriate in finding (or learning) nonlinear class boundaries represented as linear models: The **maximum margin hyperplane** is thus a linear model that represents a nonlinear decision boundary in the original space constructed.

3.Linear and Non-linear SVM

Linear SVM	Non-Linear SVM
It can be easily separated with a linear line.	It cannot be easily separated with a linear line.
Data is classified with the help of hyperplane.	We use Kernels to make non-separable data into separable data.
Data can be easily classified by drawing a straight line.	We map data into high dimensional space to classify.

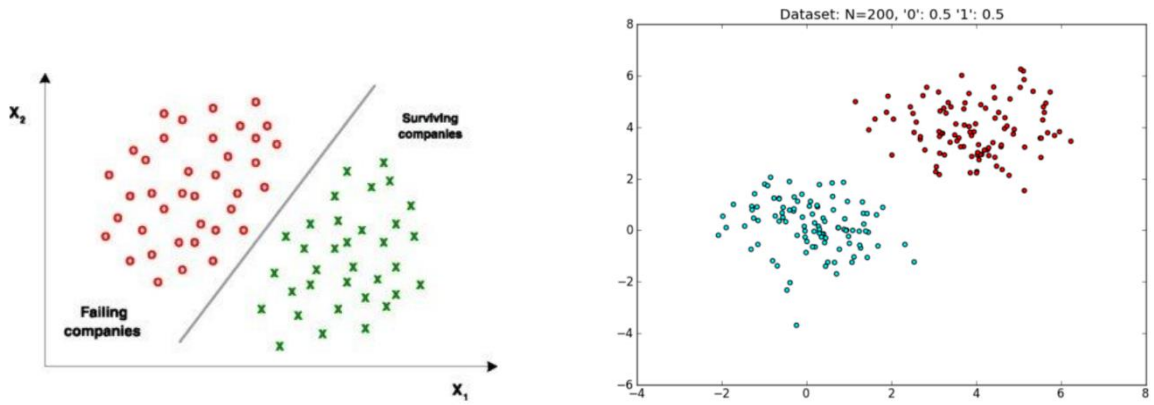


Figure 3: Linear separable data

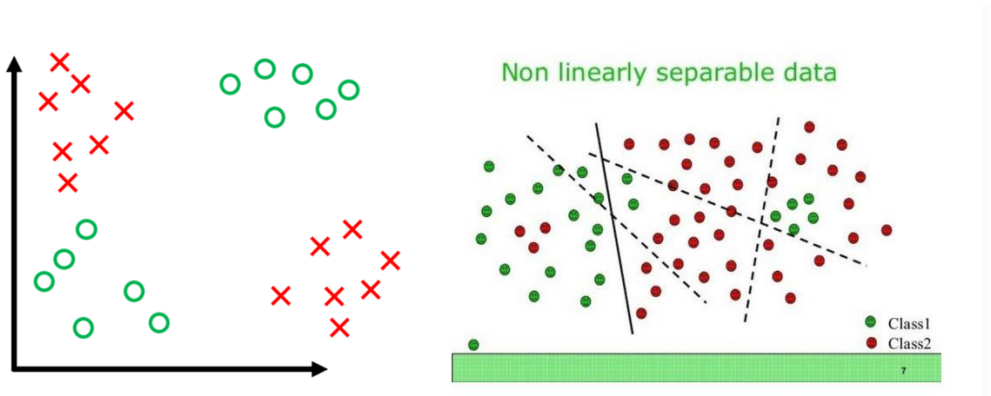
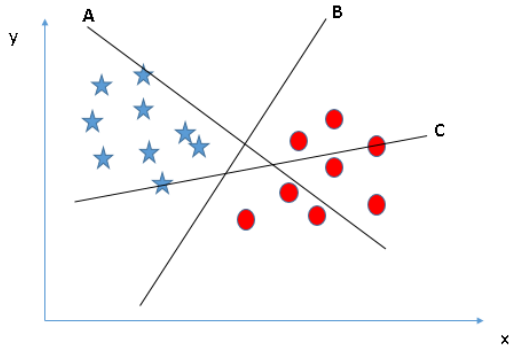


Figure 4: non-linear separable data

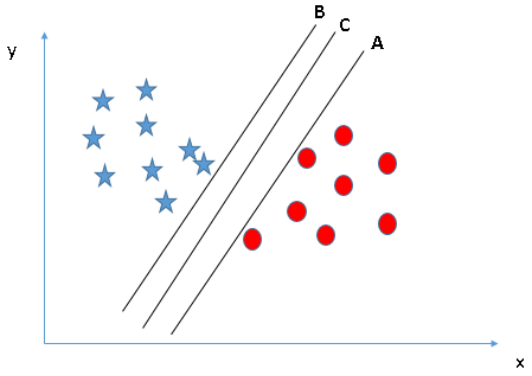
4.How does it work?

- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.

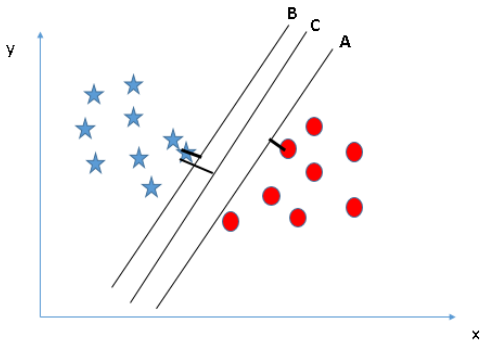


We need to remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.

- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, how can we identify the right hyper-plane?

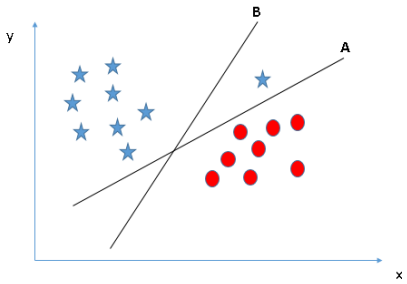


Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**.



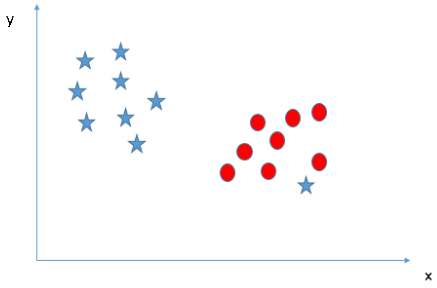
Above, we can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

- **Identify the right hyper-plane (Scenario-3):** Hint: Use the rules as discussed in previous section to identify the right hyper-plane

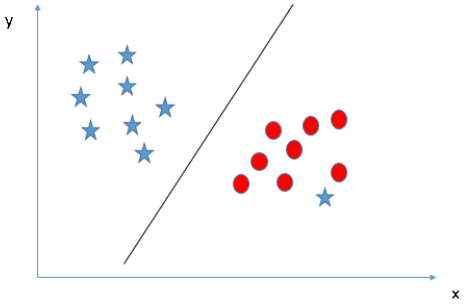


Some of we may have selected the hyper-plane **B** as it has higher margin compared to **A.** But here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

- **Can we classify two classes (Scenario-4)?** Below, I am unable to segregate the two classes using a straight line, as one of the stars lies in the territory of other(circle) class as an outlier.

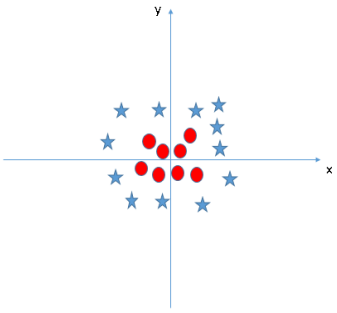


As I have already mentioned, one star at another end is like an outlier for star class. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.



4.1. Projection into Higher Dimension

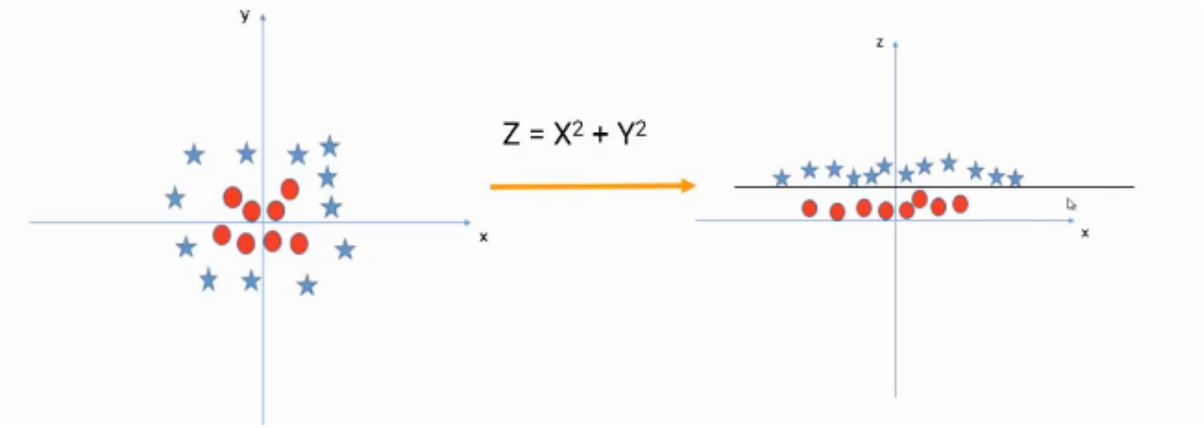
Now let's understand how SVM projects the data into a higher dimension. Take this data, it is a circular non linearly separable dataset.



To project the data in a higher dimension, we are going to create another dimension z, where

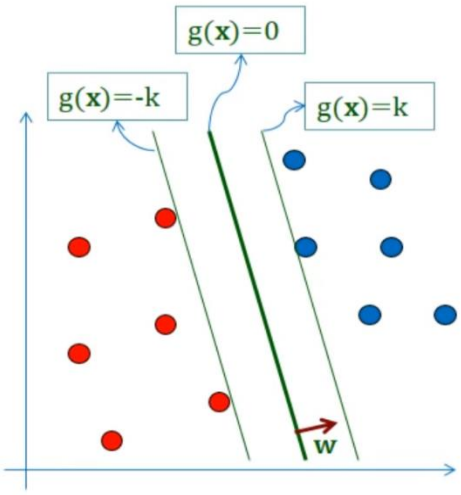
$$Z = X^2 + Y^2$$

Now we will plot this feature Z with respect to x, which will give us linearly separable data that looks like this.



Here, we have created a mapping Z using the base features X and Y, this process is known as **kernel transformation**. Precisely, a kernel takes the features as input and creates the linearly separable data in a higher dimension.

5. Formulation Of SVM



$$g(x) = w^T x + b$$

$$Maximize\ k\ such\ that:$$

$$-w^T x + b \geq k\ for\ d_i == 1$$

$$-w^T x + b \leq k\ for\ d_i == -1$$

$$Value\ of\ g(x)\ depends\ upon\ ||w||:$$

1) $Keep\ ||w|| = 1\ and\ maximize\ g(x)\ or,$

2) $g(x) \geq 1\ and\ minimize\ ||w||$

We use the approach 2 and formulate the problem as:

$$\phi(w) = \frac{1}{2} w^T w\ -\ minimize$$

$$Subject\ to\ d_i(w^T x + b) \geq 1\ \forall\ i$$

Integrating the constants in Lagrangian form we get:

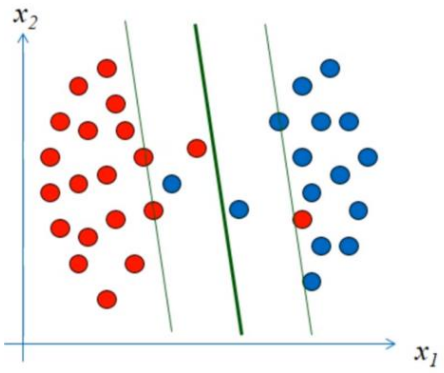
$$Minimize:\ J(w, b, \alpha) = \frac{1}{2} w^T w\ - \sum_{i=1}^{N} \alpha_i d_i (w^T x_i + b)\ + \sum_{i=1}^{N} \alpha_i$$

$$Subject\ to: \alpha_i \geq 0\ \forall\ i$$

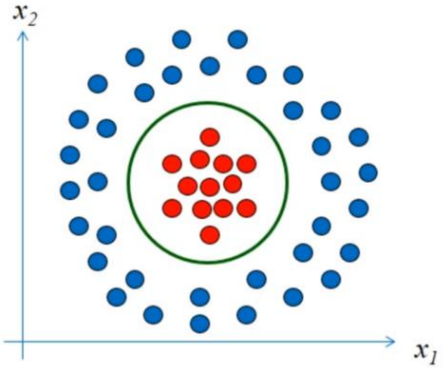
Method of Lagrange multipliers states that J is minimized for w and b as before, but it has to be **maximized** for **α.** The point that J represents is called a saddle point.

Non-Linear Separable Data

1: Noisy Data/Bad Features



2: Non-linear Boundary



Noisy Data

For noisy data, we introduce a training error parameter in our estimation/optimization. We introduce a slack variable and add the extra condition as

$$d_i(w^T x_i \ + \ b) \geq 1 \ - \ \epsilon_i$$

Going through the same process again with Lagrange coefficients we get

$$Minimize : \phi(w) = \frac{1}{2} w^T w \ + \ C \sum_{i=1}^{N} \epsilon_i$$

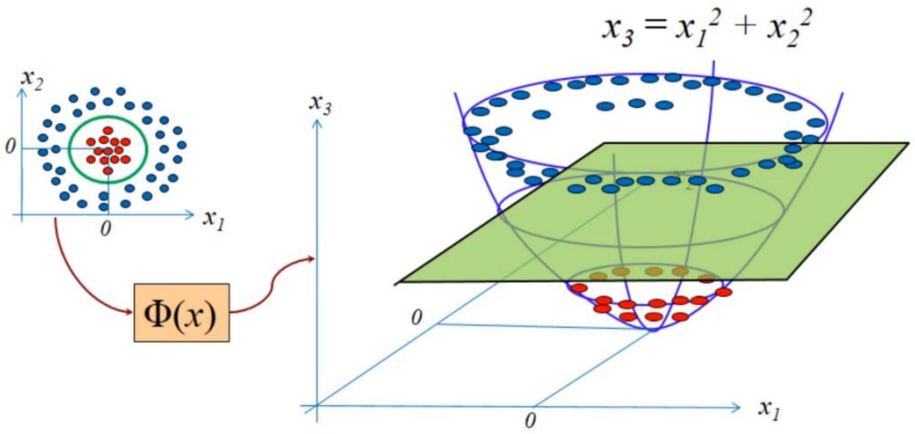
$$\Rightarrow Q(\alpha) = \sum_{i=1}^{N} \alpha_i \ - \ \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T x_j$$

$$where \ 0 \leq \alpha_i \leq C \ \forall \ i$$

The only difference is that there is a limit on the Lagrange coefficients now. The parameter C controls the relative weight between training error and VC dimension.

Non-Linear Boundary

Any data set which has a non-linear boundary would be theoretically linearly separable if projected to higher dimensions.



Hence Q(α) can we written as:

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i \; - \; \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \; \phi(x_i) \cdot \phi(x_j)$$

$$where \; 0 \leq \alpha_i \leq C \; \forall \; i$$

6.How to tune Parameters of SVM?

Tuning the parameters' values for machine learning algorithms effectively improves model performance. Let's look at the list of parameters available with SVM.

1. Margin
2. Regularization
3. Gamma
4. Kernel

6.1. Margin:

- Margin is the perpendicular distance between the closest data points and the Hyperplane (on both sides)
- The best optimized line (hyperplane) with maximum margin is termed as Margin Maximal Hyperplane.
- The closest points where the margin distance is calculated are considered as the support vectors.

6.2. Kernel:

Kernel in the SVM is responsible for transforming the input data into the required format. Some of the kernels used in SVM are linear, polynomial and radial basis function (RBF). For creating a non-linear hyperplane, we use RBF and Polynomial function. For complex applications, one should use more advanced kernels to separate classes that are nonlinear in nature. With this transformation, one can obtain accurate classifiers.

Various kernels available:

1. Linear kernel
2. Non - linear kernel
3. Radial basis function (RBF)
4. Sigmoid
5. Polynomial
6. Exponential

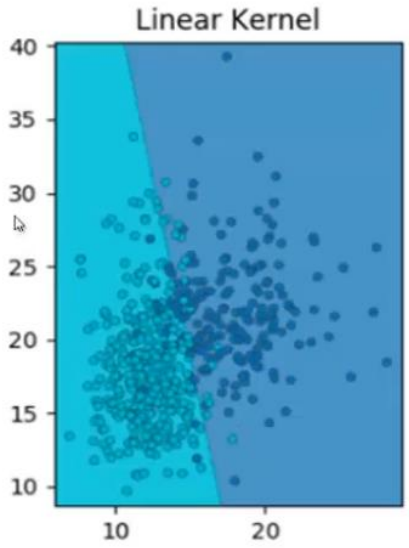
Example: **K(x, y) = <f(x), f(y)>** Kernel function dot product of n- dimensional inputs

6.2.1. Linear Kernel

To start with, in the linear kernel, the decision boundary is a straight line. Unfortunately, most of the real-world data is not linearly separable, this is the reason the linear kernel is not widely used in SVM.

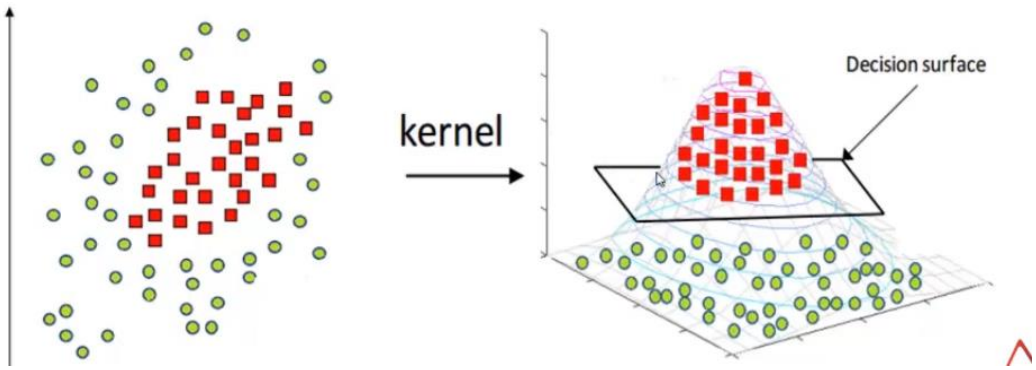
It is defined simply as

$$K(\boldsymbol{x}, \boldsymbol{z}) = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{z}) = \boldsymbol{x} \cdot \boldsymbol{z}$$



6.2.2. Gaussian / RBF kernel

It is the most commonly used kernel. It projects the data into a Gaussian distribution, where the red points become the peak of the Gaussian surface and the green data points become the base of the surface, making the data linearly separable.



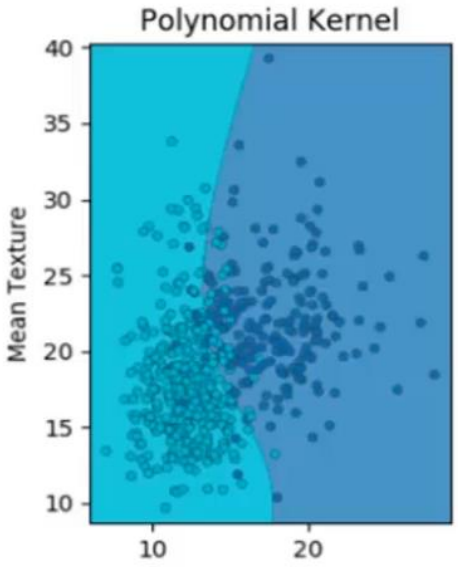
But this kernel is prone to overfitting and it captures the noise.

The gaussian kernel is probably the mostly used kernel function and is defined as

$$K(\boldsymbol{x}, \boldsymbol{z}) = exp\left\{-\frac{\|\boldsymbol{x} - \boldsymbol{z}\|^2}{2\sigma^2}\right\}$$

6.2.3. Polynomial kernel

At last, we have a polynomial kernel, which is non-uniform in nature due to the polynomial combination of the base features. It often gives good results.



But the problem with the polynomial kernel is, the number of higher dimension features increases exponentially. As a result, this is computationally more expensive than RBF or linear kernel.

The homogeneous polynomial kernel of degree d(d>_2) is defined as

$$K(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x} \cdot \boldsymbol{z})^d$$

6.3. Regularization:

Also, the ‘C ‘parameter in Python’s SkLearn Library

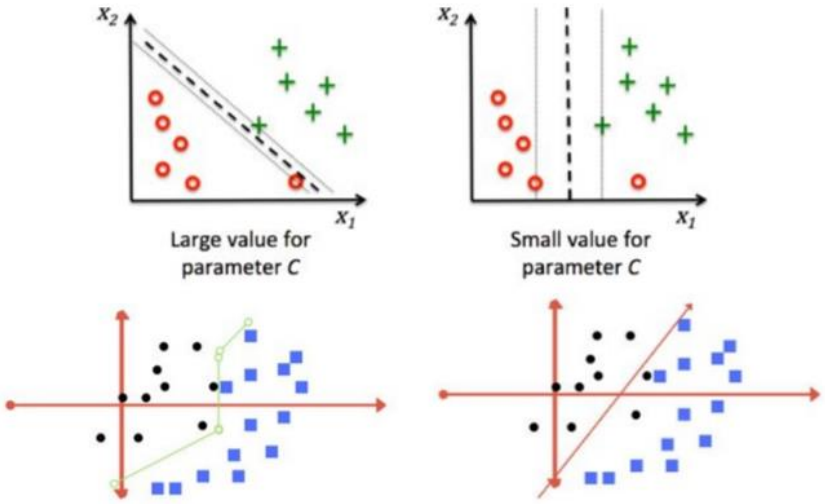
● Optimises SVM classifier to avoid misclassifying the data.

● C → large Margin of hyperplane → small

● C → small Margin of hyperplane → large

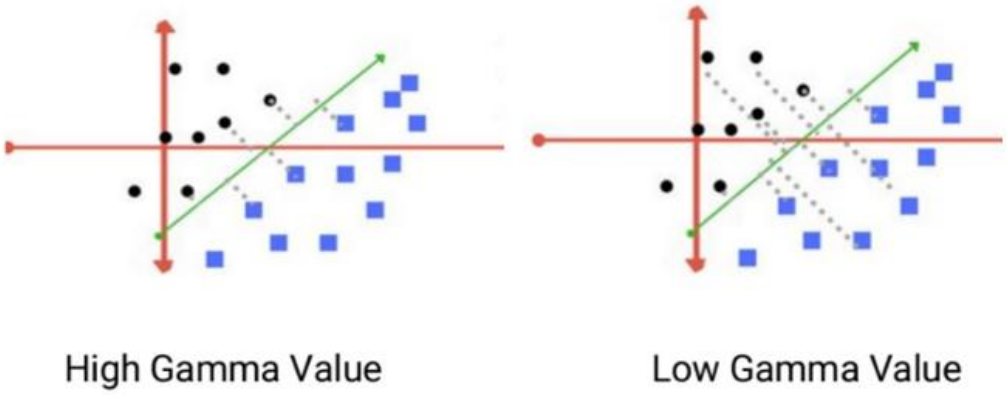
● misclassification(possible)

1. C ---> large, chance of overfit
2. 2. C ---> small, chance of underfitting



6.4. Gamma:

With a lower value of Gamma will create a loose fit of the training dataset. On the contrary, a high value of gamma will allow the model to get fit more appropriately. A low value of gamma only provides consideration to the nearby points for the calculation of a separate plane whereas the high value of gamma will consider all the data-points to calculate the final separation line.



7.How to implement SVM in Python?

In Python, scikit-learn is a widely used library for implementing machine learning algorithms. SVM is also available in the scikit-learn library and we follow the same structure for using it (Import library, object creation, fitting model and prediction).

Support Vector Machine (SVM) code in Python

In the first step, we will import the important libraries that we will be using in the implementation of SVM.

```
import pandas as pd
import numpy as np                                    #DataFlair
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler



%pylab inline
```

Figure 5: Importing libraries

In the second step of implementation of SVM in Python, we will use the iris dataset that is available with the load_iris() method. We will only make use of the petal length and width in this analysis.

```
pylab.rcParams['figure.figsize'] = (10, 6)

iris_data = datasets.load_iris()

# We'll use the petal length and width only for this analysis
X = iris_data.data[:, [2, 3]]
y = iris_data.target

# Place the iris data into a pandas dataframe
iris_dataframe = pd.DataFrame(iris_data.data[:, [2, 3]],
                              columns=iris_data.feature_names[2:])

# View the first 5 rows of the data
print(iris_dataframe.head())

# Print the unique labels of the dataset
print('\n' + 'Unique Labels contained in this data are '
      + str(np.unique(y)))
```

Figure 6: Importing the data

In the next step, we will split our data into training and test set using the train_test_split() function as follows –

```
[ ]  X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3, random_state=0)

     print('The training set contains {} samples and the test set contains {} samples'
           .format(X_train.shape[0], X_test.shape[0]))
```

```
The training set contains 105 samples and the test set contains 45 samples
```

Figure 7: Train-Test-Split

Let us now visualize our data. We observe that one of the classes is linearly separable.

```
markers = ('x', 's', 'o')
colors = ('red', 'blue', 'green')
cmap = ListedColormap(colors[:len(np.unique(y_test))])
for idx, cl in enumerate(np.unique(y)):
    plt.scatter(x=X[y == cl, 0], y=X[y == cl, 1],
                c=cmap(idx), marker=markers[idx], label=cl)
```

Figure 8: Plotting the data

Output:

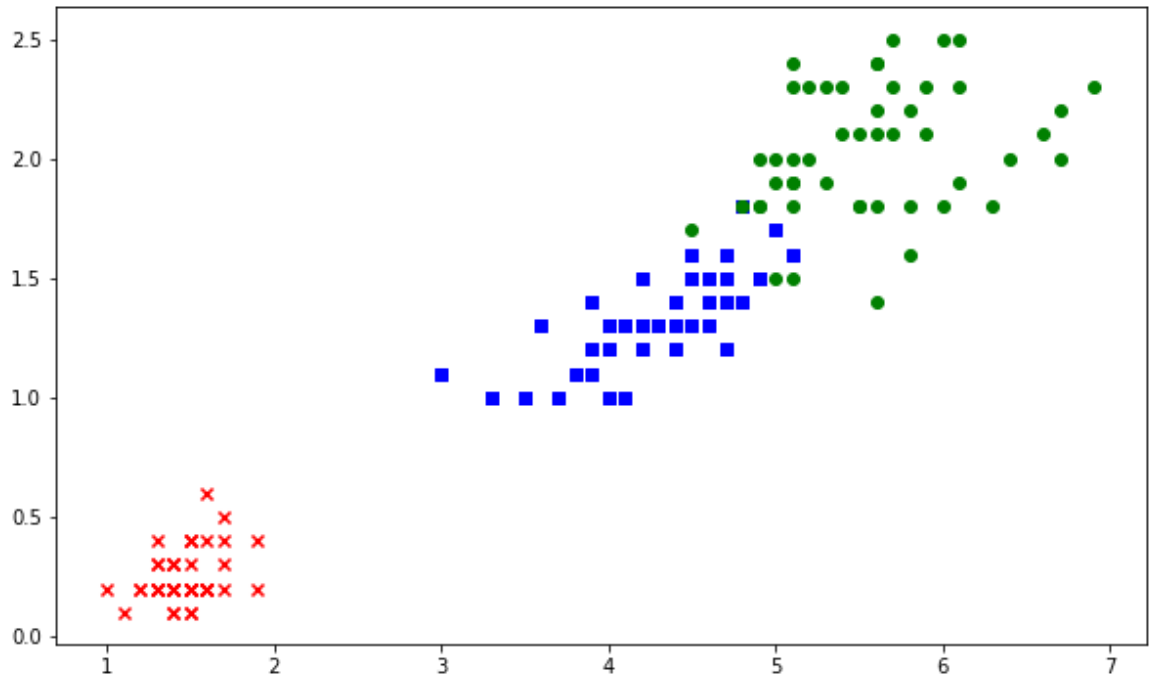


Figure 9: Visualizing the data

Then, we will perform scaling on our data. Scaling will ensure that all of our data-values lie on a common range such that there are no extreme values.

```
standard_scaler = StandardScaler()
#DataFlair
standard_scaler.fit(X_train)

X_train_standard = standard_scaler.transform(X_train)
X_test_standard = standard_scaler.transform(X_test)

print('The first five rows after standardisation look like this:\n')
print(pd.DataFrame(X_train_standard, columns=iris_dataframe.columns).head())
```

```
The first five rows after standardisation look like this:

   petal length (cm)  petal width (cm)
0          -0.182950         -0.293181
1           0.930661          0.737246
2           1.042022          1.638870
3           0.652258          0.350836
```

Figure 10: Scaling data in SVM

After we have pre-processed our data, the next step is the implementation of the SVM model as follows. We will make use of the SVC function provided to us by the sklearn library. In this instance, we will select our kernel as ‘rbf’.

```
[ ]  #DataFlair
     SVM =  SVC(kernel='rbf', random_state=0, gamma=.10, C=1.0)
     SVM.fit(X_train_standard, y_train)

     print('Accuracy of our SVM model on the training data is {:.2f} out of 1'
           .format(SVM.score(X_train_standard, y_train)))

     print('Accuracy of our SVM model on the test data is {:.2f} out of 1'
           .format(SVM.score(X_test_standard, y_test)))
```

Figure 11: Implementation of SVM model

After we have achieved our accuracy, the best course of action would be to visualize our SVM model. We can do this by creating a function called decision_plot() and passing values to it as follows –

```
import warnings


def versiontuple(version):
    return tuple(map(int, (version.split("."))))


def decision_plot(X, y, classifier, test_idx=None, resolution=0.02):

    # setup marker generator and color map
    markers = ('s', 'x', 'o', '^', 'v')
    colors = ('red', 'blue', 'green', 'gray', 'cyan')
    cmap = ListedColormap(colors[:len(np.unique(y))])

    # plot the decision surface
    x1min, x1max = X[:, 0].min() - 1, X[:, 0].max() + 1
    x2min, x2max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx1, xx2 = np.meshgrid(np.arange(x1min, x1max, resolution),
                           np.arange(x2min, x2max, resolution))
    Z = classifier.predict(np.array([xx1.ravel(), xx2.ravel()]).T)
    Z = Z.reshape(xx1.shape)
    plt.contourf(xx1, xx2, Z, alpha=0.4, cmap=cmap)
    plt.xlim(xx1.min(), xx1.max())
    plt.ylim(xx2.min(), xx2.max())

    for idx, cl in enumerate(np.unique(y)):
        plt.scatter(x=X[y == cl, 0], y=X[y == cl, 1],
                    alpha=0.8, c=cmap(idx),
                    marker=markers[idx], label=cl)
```

Figure 12: Decision plot function in SVM

```
decision_plot(X_test_standard, y_test, SVM)
```

Figure 13: Decision plot call

Output:

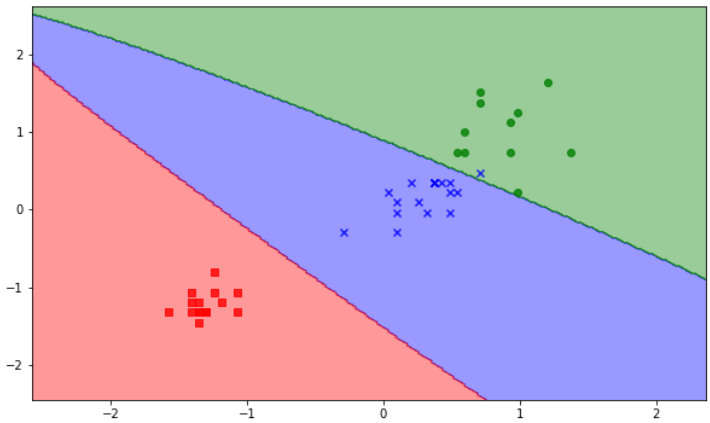


Figure 14: SVM decision plot

8.Advantages and Disadvantages of Support Vector Machine

- ## Advantages:

    - It works really well with a clear margin of separation.
    - It is effective in high dimensional spaces.
    - It is effective in cases where the number of dimensions is greater than the number of samples.
    - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
    - SVMs provide compliance to the semi-supervised learning models. It can be used in areas where the data is labeled as well as unlabeled. It only requires a condition to the minimization problem which is known as the Transductive SVM.
    - Feature Mapping used to be quite a load on the computational complexity of the overall training performance of the model. However, with the help of Kernel Trick, SVM can carry out the feature mapping using simple dot product.

- ## Disadvantages:

    - It doesn’t perform well when we have large data set because the required training time is higher
    - It also doesn’t perform very well, when the data set has more noise i.e., target classes are overlapping
    - SVM doesn’t directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is included in the related SVC method of Python scikit-learn library.

9.APPLICATIONS

The aim of using SVM is to correctly classify unseen data. SVMs have a number of applications in several fields.

Some common applications of SVM are-

1.**Face detection**

It classifies the parts of the image as face and non-face. It contains training data of n x n pixels with a two-class face (+1) and non-face (-1). Then it extracts features from each pixel as face or non-face. Creates a square boundary around faces on the basis of pixel brightness and classifies each image by using the same process.

2.Classification of Images

SVMs can classify images with higher search accuracy. Its accuracy is higher than traditional query-based refinement schemes.

3. Bioinformatics

In the field of computational biology, the protein remote homology detection is a common problem. The most effective method to solve this problem is using SVM. In last few years, SVM algorithms have been extensively applied for protein remote homology detection. These algorithms have been widely used for identifying among biological sequences. For example, classification of genes, patients on the basis of their genes, and many other biological problems.

4. Protein Fold and Remote Homology Detection

Protein remote homology detection is a key problem in computational biology. **Supervised learning algorithms** on SVMs are one of the most effective methods for remote homology detection. The performance of these methods depends on how the protein sequences modelled. The method used to compute the kernel function between them

5. Generalized Predictive Control

We use SVM-based GPC to control chaotic dynamics with useful parameters. It provides excellent performance in controlling the systems. The system follows chaotic dynamics with respect to the local stabilization of the target.

Using SVMs for controlling chaotic systems has the following advantages-

- Allows use of relatively small parameter algorithms to redirect a chaotic system to the target.
- Reduces waiting time for chaotic systems.

6. Text and hypertext categorization – SVMs allow Text and hypertext categorization for both inductive and transudative models. They use training data to classify documents into different categories. It categorizes on the basis of the score generated and then compares with the threshold value.

10.CONCLUSION

Thus, we conclude that the SVMs cannot only make the reliable prediction but also can reduce redundant information. The SVMs also obtained results comparable with those obtained by other approaches.

The method of support vector machines as an alternative to the conservative logistic regression models was studied and its performance compared on the real credit data sets. Especially in combination with the non-linear kernel, SVM proved itself as a competitive approach and provided a slight edge on top of the logistic regression model.

We learned how these SVM algorithms work and also implemented them with real-life example. We also discussed the various applications of SVMs in our daily lives.

11.REFERENCES

[1]. Rashmi Jain “Tutorial on SVM and Tuning Parameter” [Accessed Feb. 21, 2017]

https://www.hackerearth.com/blog/developers/simple-tutorial-svm-parameter-tuning-python-r/

[2]. Master’s thesis by Michal Haltuf “Support Vector Machines for Credit Scoring”

https://svm.michalhaltuf.cz/support-vector-machines/

[3]. Rushikesh Pupale “SVM- an overview” [Accessed Jun 16,2018”]

https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989