# **SVM**

SVM is a powerful supervised algorithm that works best on smaller datasets but complex ones. Support vector Machine can be used for both regression and classification tasks. SVMs are adaptable and efficient in a variety of applications because they can manage high-dimensional data and non-linear relationships.

SVM algorithms are very effective as we try to find t maximum separating hyperplane between to different classes available is the target feature.

The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible.

The dimension of the hyperplane depends upon the number of features.

Lets’ consider two classes A and B

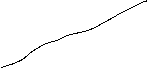
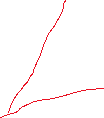
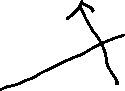
Our main objective is to segregate the given dataset in the best way. The distance between the nearest points is known as margin. SVM searches for the maximum marginal hyperplane in the following steps.

Class A



v

margin



v

Support vectors Class B



The hyperplane by one equation can be shown as this one

g(x->) = w-> Tx-> + w0

g(x->) >= 1,

g(x->) <= 1,

Here we have a vector of weights plus omega O(W0 ) and this equation will deliver valuer greater that f for all the input vectors which belongs to the class 1, in this case the circles , and also we seale this hyperplane so that it will deliver values smaller than -1 for all values that belongs to class number 2,

We know that the distance between a point and a hyperplane is computed by this equation, so the total margin which is composed by this distance will be computed by this equation.

The aim is that minimizing this term will maximize the separability. When we minimize this weight vector we will have the biggest margin here that will split there two classes.

Minimizing w(vector) is a nonlinear optimization task, solved by the KARUSH-KUHN-TUCKER(KKT)conditions using Langrage multipliers theta i.

So when we solve these equations. Trying to minimize this omega vector, we will maximize the margin between the two classes which will maximize the separability two classes

## **EXAMPLE**

Suppose we have these 2 features, x1 and x2 and we have these 3 values, we want to design, or to find the best hyperplane that will divide this 2 classes.

(2.0)



(1.1)



(2.0)



We can see that the best division line will be a parallel line to the line this connects these 2 values. We can define this weight vector, which is (2,3) point – (1,1) point => w->

= (2,3) – (1,1) = (a, 2a). We have constant a and 2 times that constant

a weight w-> = (a,2a)

a +2a+w0 = -1 using (1,1)

2a +6a +w0 = 1 using (2,3)

w0 = 1- 8a 5a =2

a = 2/3

w0 = 1 – 8(2/5) = (5-16)/5

w0 = -11/ 5 g(x->) = 2/5x1 + 4/5x2 – 11/5

w-> = (2/5, 4/5) g(x->) = x1 + 2x2 – 5,5

w-> = (2a, 2a)

These are called support vectors w-> = (2/5,4/5) the g(x->) is define the green hyperplane and the hyperplane classifies the elements.

## **Pseudocode SVM**

from sklearn.svm import SVC

from sklearn.datasets import make classification

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

# Generate synthetic classification data

X, y = make\_classification(n\_samples=100, n\_features=4, n\_classes=2, random\_state=42)

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Create an SVM classifier with a linear kernel

svm = SVC(kernel='linear')

# Train the classifier on the training data

svm.fit(X\_train, y\_train)

# Make predictions on the testing data

y\_pred = svm. predict(X\_test)

# Calculate the accuracy of the classifier

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)