***SUPPORT VECTOR MACHINE(SVM)***

**Intoduction:**

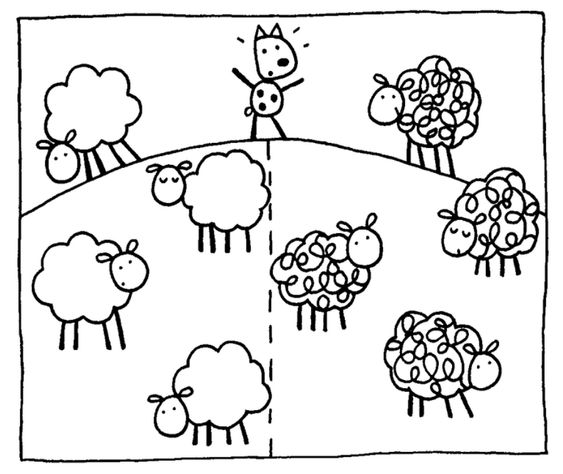
Support Vector Machine (SVM) is one of the AI (ML) Managed calculations. There are a lot of calculations in ML, yet, gathering for SVM is consistently extraordinary as a result of its power while managing the information. So here in this article, we will cover practically every one of the important things that need to drive for any sort of information w.r.t SVM. Prior to getting profound into the point, let us get wet by perhaps looking out for a way to improve on a portion of the essential wording connected with SVM. I genuinely want to believe that you will appreciate it! It very well may be a piece extensive and sure it will not frustrate you!

Fig1.pinterest.com

**SVM(theoretical Introduction):**

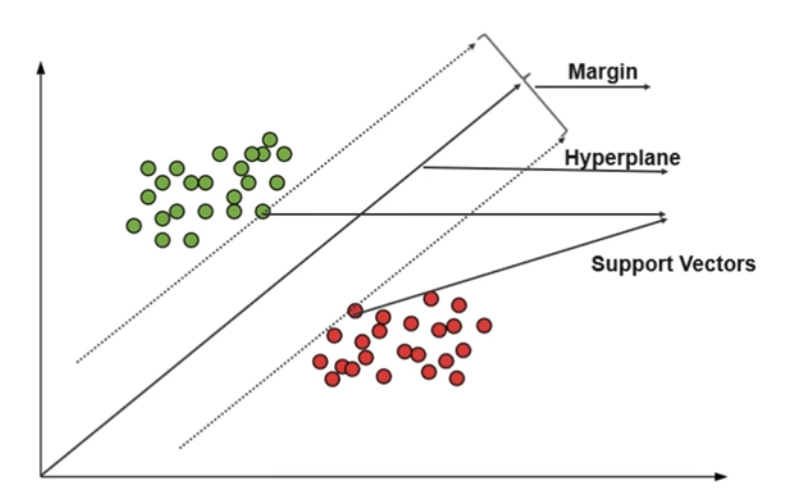


Fig 2:edureka.com

* SVM – Comes under Supervised ML
* SVM can perform both Classification & Regression
* Goal – Create the best decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data points in the correct category – Hyperplane.
* Out-of-the-box classifier
* For a better understanding of SVM, we will learn,

@ Maximal Margin Classifier

@ Support Vector Classifier

@ Support Vector Machine

**SVM – Maximal Margin Classifier:**

Before we know about Maximal Margin Classifier (MMC), let us start from the basics, we all know the terms 1 Dimensional (1D), 2 Dimensional (2D), and 3 Dimensional (3D). So what it is?

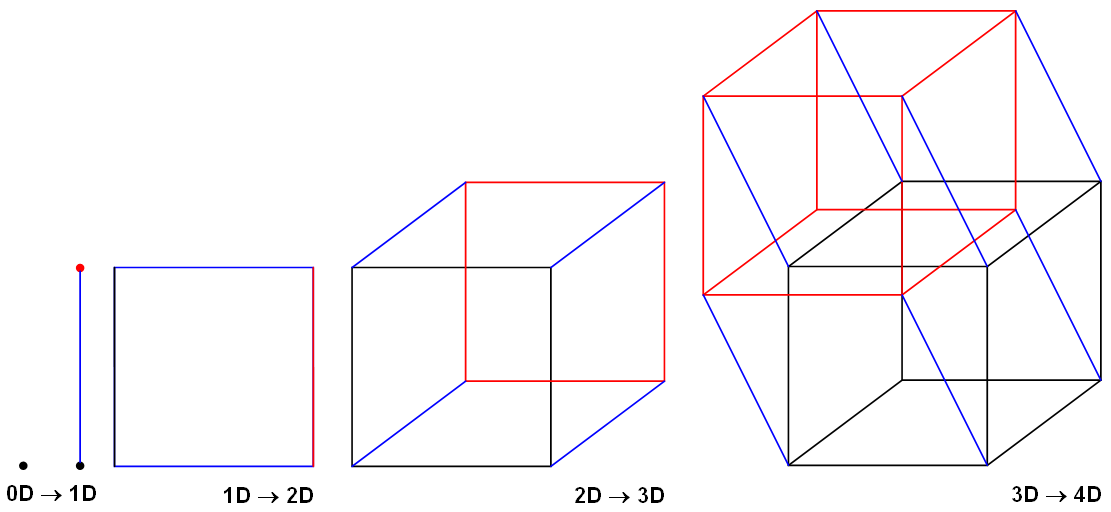
In short, Dimension means measurement (total amount of measurable space or surface occupied). In simple, a number of dimensions are how many values are needed to locate points on a shape

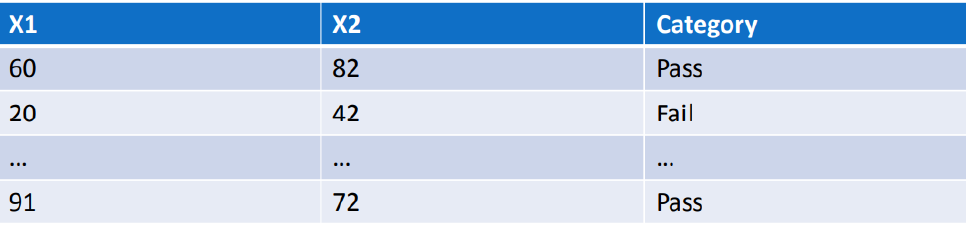
Fig 3:wikipedia.com

1. ***No dimension or 0-D*** – A point (the only position exists)A point really has no size at all! But we show them as dots so we can see where they are!
2. ***One dimension or 1-D*** – A-line (with two points)Now let’s allow the point to move in one direction. We get a line. We need just one value to find a point on that line.
3. ***Two-dimension or 2-D*** – A-plane Now let us allow the point to move in a different direction. And we get a plane. We need two values to find a point on that plane.
4. ***Three dimensions or 3-D*** – A-Solid (maybe a cube)Now we let that point move in another completely different direction and we have three dimensions.

Note: A point is a hyperplane in 1-dimensional space, a line is a hyperplane in 2-dimensional space, and a plane is a hyperplane in 3-dimensional space.

So now I hope we get some clarity w.r.t the dimensionality concept. So here for SVM, we will be using a term called HYPERPLANE, this plays important role in classifying the data into different groups (will see in detail very soon here in this article!).

Let us assume from the below-given figure, we have a dataset that has X1 and X2 as independent features and Category as a dependent feature, sometimes we call it as target or label,

Fig 4:image.google.com

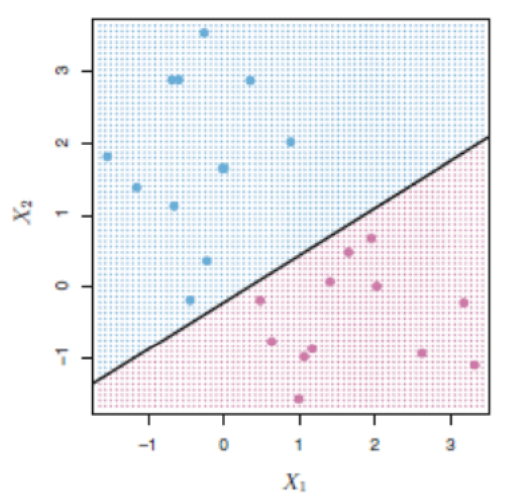
Let us assume, from the above-mentioned dataset with 2 independent features (X1 & X2) are plotted in 2D space or graph in simple and separated by a line for class category (Pass or Fail).

Fig 5:google.image.com

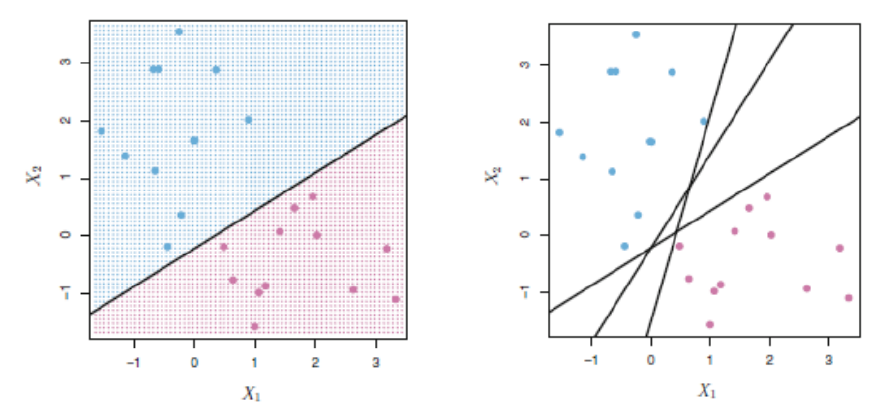
There are many possibilities to separate the two category classes – But which is best among all?

Fig 6:google.image.com

We cannot predict it right, so the solution for the problem is HYPERPLANE. It can be picturized by the below figure in a generalized way,

If you observed in the above figure, we can clearly see like among three-line, line (H3) which has the maximum margin with the data points and also classified between the data properly, if you see line (H2) the margin is small with one data and large with another data, whereas, line (H1), it has not classified between the data itself.

**Support Vector Classifier (SVC):**

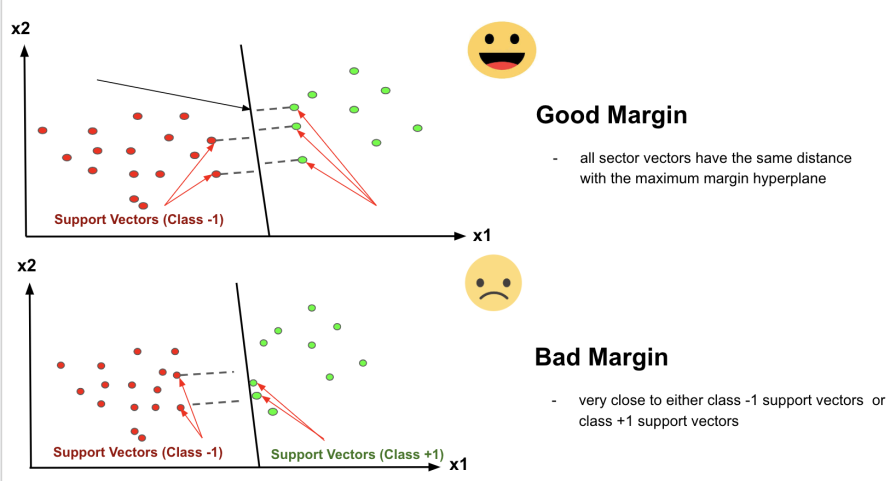
Many have confusion with the terms SVM and SVC, the simple answer is if the hyperplane that we are using for classification is in linear condition, then the condition is SVC.The distance of the vectors from the hyperplane is called the margin which is a separation of a line to the closest class points. We would like to choose a hyperplane that maximizes the margin between classes. The graph below shows what good margins and bad margins are.

Fig 7:image.google.com

Again Margin can be sub-divided into,

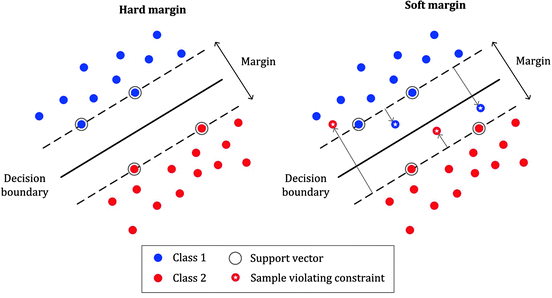
1. Soft Margin – As most of the real-world data are not fully linearly separable, we will allow some margin violation to occur which is called soft margin classification. It is better to have a large margin, even though some constraints are violated. Margin violation means choosing a hyperplane, which can allow some data points to stay on either the incorrect side of the hyperplane and between the margin and correct side of the hyperplane.
2. 2. Hard Margin – If the training data is linearly separable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is as large as possible.
3. Note: In order to find the maximal margin, we need to maximize the margin between the data points and the hyperplane.

Fig 8:medium.com

**Support Vector Machine (SVM):**

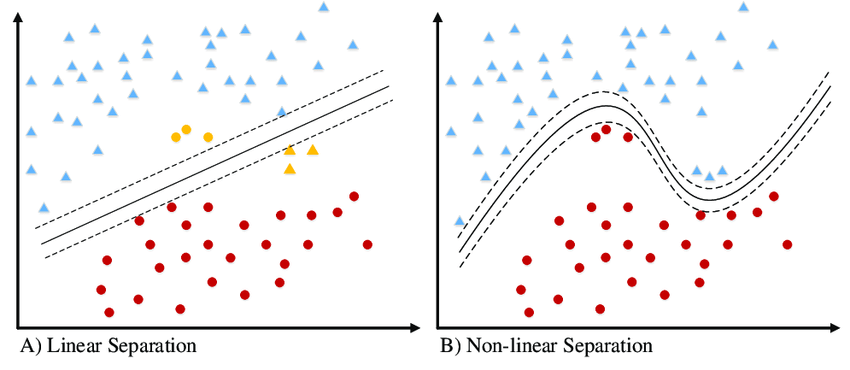
The limitation of SVC is compensated by SVM non-linearly. And that’s the difference between SVM and SVC. If the hyperplane classifies the dataset linearly then the algorithm we call it as SVC and the algorithm that separates the dataset by non-linear approach then we call it as SVM.

Fig 9:google.image.com

SVM has a technique called the kernel trick. These are functions that take low dimensional input space and transform it into a higher-dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in non-linear separation problems.

In a more precise manner, Nonlinear can be explained by another visual representation as below, where it can be clearly seen that how the data points which is not able to linearly classified is converted into higher dimensional, then get separated linearly in higher space, which is back non-linearly separated in the original dimension of lower space.

Still confused with the above figures, if you still need more clarity, see the below video for crystal clear idea, how the data are transformed into higher dimensional and from there how linearly got separated then back to the normal plane,

There are some famous and most frequently used Non-linear kernels in SVM are,

1. Polynomial SVM Kernel

2. Gaussian Radial Basis Function (RBF)

3. Sigmoid Kernel

**SVM – Advantage, Disadvantage & Applications:**

**Advantage:**

* More effective for high dimensional space
* Works well with even unstructured and semi-structured data like text, Images, and trees
* Handles non-linear data efficiently by using the kernel trick
* SVM has an L2 Regularization feature. So, it has good generalization capabilities which prevent it from over-fitting
* A small change to the data does not greatly affect the hyperplane and hence the SVM. So the SVM model is stable

**Disadvantage:**

* Not suitable for large dataset
* Sensitive to outliers (If you have more in the dataset then SVM is not the right choice!)
* Hyperparameters like cost (C) and gamma of SVM, is not that easy to fine-tune and also hard to visualize their impact
* SVM takes a long training time on large dataset
* SVM model is difficult to understand and interpret by human beings, unlike Decision Trees.
* One must do feature scaling of variables before applying SVM

**Applications:**

* Handwriting recognition
* Face Detection
* Text and hypertext categorization
* Image Classification
* Bioinformatics (protein classification and cancer classification)

**Coding and Dataset:**

|  |
| --- |
| import pandas as pd  df=pd.read\_csv("/content/drive/MyDrive/csv2/Iris (1).csv")  df |

We have use colab in this we first of all imported pandas library with an abbreviation pd because pd is efficient to be written instead of pandas ,then we take a variable named df and passes the dataset of iris(a flower) in csv format. After this we printed the variable df.

|  |
| --- |
| from sklearn import preprocessing  le = preprocessing.LabelEncoder()  le.fit(df.Species)  df["Species"]=le.transform(df.Species)  df["Species"]  y=df.loc[:,"Species"]  y  x=df.loc[:,["SepalLengthCm","SepalWidthCm","PetalLengthCm","PetalWidthCm"]]  x |

In this block we imported preprocessing library from sklearn, then stored lable encoder function from preprocessing in “le” variable.in the next line we fit the feature Species and apply the transform function on it ,this function will give a specific code across each value.

Then we printed the species feature.

After this the species was labed as target output and stored in y variable.In the end all the other features were stored as input in x variable.

|  |
| --- |
| from sklearn.model\_selection import train\_test\_split  x\_train,x\_test,y\_train,y\_test= train\_test\_split(x,y,test\_size=0.30,random\_state=0) |

In this part train test model was called from sklearn library and the whole dataset was divide into training and testing portion with 70/30 ratio and random state was 0 which means data was randomly selected from full dataset for training and testing.

|  |
| --- |
| from sklearn import svm  clf=svm.SVC()  clf.fit(x\_train,y\_train)  clf.score(x\_test,y\_test) |

Now we have applied SVM by fetching it from sklearn and applying SVC technique on the model then fitting the training dataset and training it on svm then finally passing the tested /unseen dataset to model for checking the accuracy .We got 97% accuracy on this dataset!