Support Vector Machine (SVM)

https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html
https://www.javatpoint.com/regression-vs-classification-in-machine-learning
https://www.baeldung.com/cs/svm-multiclass-classification

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labeled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms that Regression algorithms are used to predict the continuous values such as price, salary, age, etc. and Classification algorithms are used to predict/Classify the discrete values such as Male or Female, True or False, Spam or Not Spam, etc.

Types of ML Classification Algorithms:

Classification Algorithms can be further divided into the following types:

- **Logistic Regression**
- **K-Nearest Neighbours**
- **Support Vector Machines**
- **Kernel SVM**
- **Naïve Bayes**
- **Decision Tree Classification**
- **Random Forest Classification**

Regression is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables such as prediction of Market Trends, prediction of House prices, etc.

The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

Types of Regression Algorithm:

- **Simple Linear Regression**
- **Multiple Linear Regression**
- **Polynomial Regression**
- **Support Vector Regression**
- **Decision Tree Regression**
- **Random Forest Regression**

The regression Algorithm can be further divided into Linear and Non-linear Regression.

The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier.

Multiclass classification is a machine learning classification task that consists of more than two classes, or outputs.

Support Vector Machines - What are they?

SVM helps in creating not only a single hyperplane but instead it creates two parallel hyperplane in such a way that one of the hyperplane will be passing through the nearest positive point and the other hyperplane will be passing through the nearest negative point
And the distance between two parallel lines is called margin

A large margin effectively corresponds to a regularization of SVM weights which prevents overfitting. Hence, we prefer a large margin (or the right margin chosen by cross-validation) because it helps us generalize our predictions and perform better on the test data by not overfitting the model to the training data.

What is a hyperplane?

As a simple example, for a classification task with only two features (like the image above), you can think of a hyperplane as a line that linearly separates and classifies a set of data.

Intuitively, the further from the hyperplane our data points lie, the more confident we are that they have been correctly classified. We therefore want our data points to be as far away from the hyperplane as possible, while still being on the correct side of it.

The goal is to choose a hyperplane with the greatest possible margin between the hyperplane and any point within the training set, giving a greater chance of new data being classified correctly.

Support Vectors - nearest points that are passing through the hyperplane

How Does It Work?

In the base form, linear separation, SVM tries to find a line that maximizes the separation between a two-class data set of 2-dimensional space points. **To generalize, the objective is to find a hyperplane that maximizes the separation of the data points to their potential classes in an -dimensional space.** The data points with the minimum distance to the hyperplane (closest points) are called *Support Vectors*.

In the image below, the Support Vectors are the 3 points (2 blue and 1 green) laying on the scattered lines, and the separation hyperplane is the solid red line:

he computations of data points separation depend on a kernel function. There are different kernel functions: Linear, Polynomial, Gaussian, Radial Basis Function (RBF), and Sigmoid. Simply put, these functions determine the smoothness and efficiency of class separation, and playing around with their hyperparameters may lead to overfitting or underfitting.

In its most simple type, SVM doesn't support multiclass classification natively. It supports binary classification and separating data points into two classes. For multiclass classification, the same principle is utilized after breaking down the multiclassification problem into multiple binary classification problems.

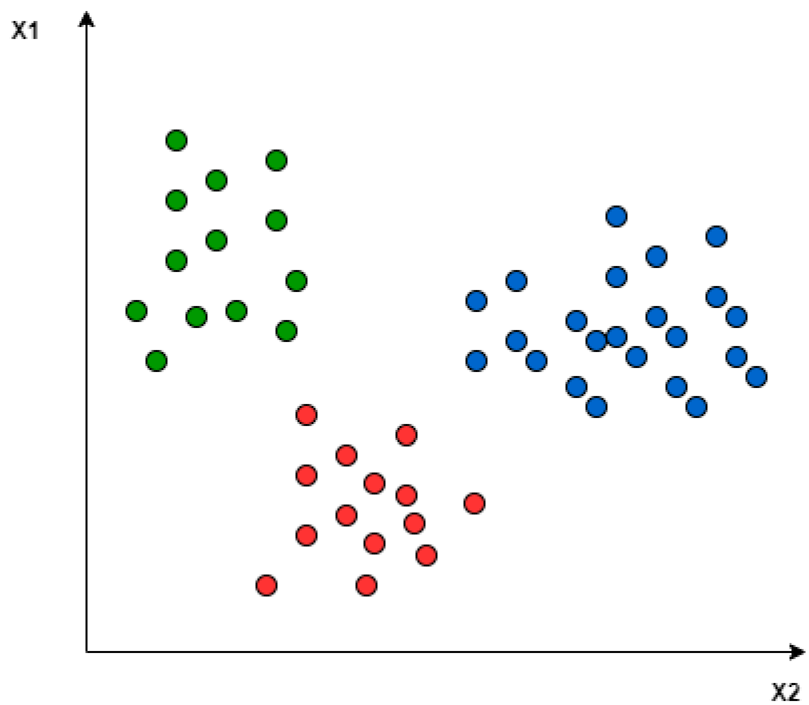
The idea is to map data points to high dimensional space to gain mutual linear separation between every two classes. **This is called a *One-to-One* approach, which breaks down the multiclass problem into multiple binary classification problems. A binary classifier per each pair of classes.**

Another approach one can use is *One-to-Rest*. In that approach, the breakdown is set to a binary classifier per each class.

A single SVM does binary classification and can differentiate between two classes. So that, according to the two breakdown approaches, to classify data points from classes data set:

- In the *One-to-Rest* approach, the classifier can use SVMs. Each SVM would predict membership in one of the classes.
- In the *One-to-One* approach, the classifier can use SVMs.

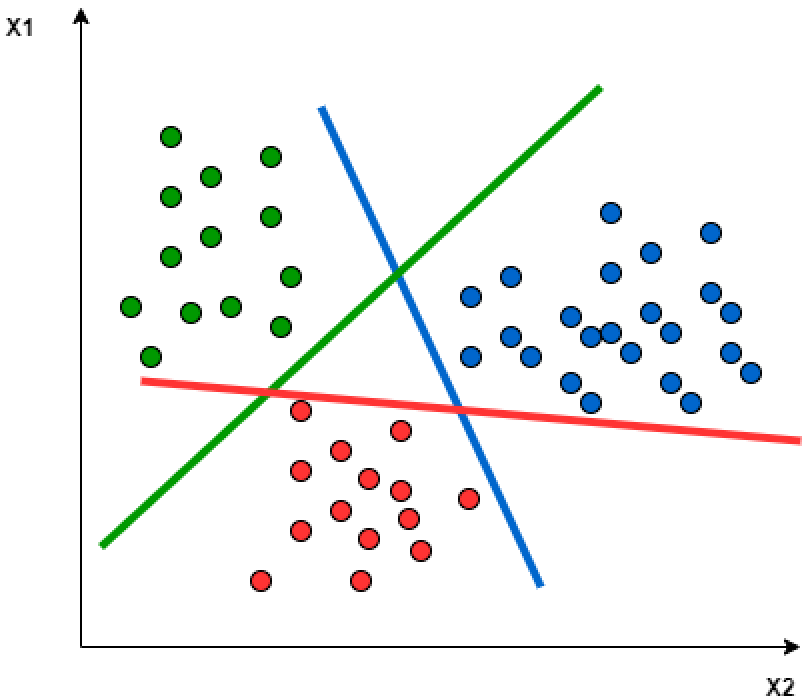
Let's take an example of 3 classes classification problem; green, red, and blue, as the following image:



Applying the two approaches to this data set results in the followings:

In the *One-to-One* approach, we need a hyperplane to separate between every two classes, neglecting the points of the third class. This means the separation takes into account only the points of the two classes in the current split. For example, the red-blue line tries to maximize the separation only between blue and red points. It has nothing to do with green points:

In the *One-to-Rest* approach, we need a hyperplane to separate between a class and all others at once. This means the separation takes all points into account, dividing them into two groups; a group for the class points and a group for all other points. For example, the green line tries to maximize the separation between green points and all other points at once:



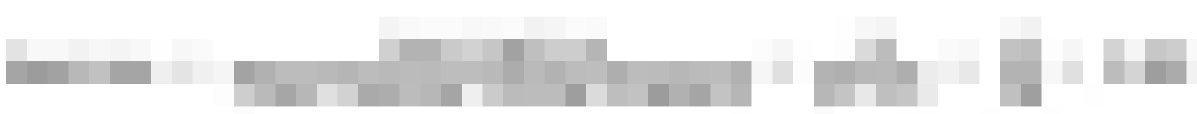
F1-score: This is the harmonic mean of Precision and Recall and gives a better measure of the incorrectly classified cases than the Accuracy Metric.

$$\text{F1-score} = \left(\frac{\text{Recall}^{-1} + \text{Precision}^{-1}}{2}\right)^{-1} = 2 * \frac{(\text{Precision}*\text{Recall})}{(\text{Precision}+\text{Recall})}$$

F1-Score

We use the Harmonic Mean since it penalizes the extreme values.

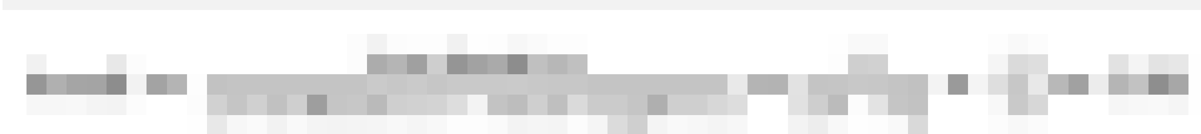
1 — Precision: It is implied as the measure of the correctly identified positive cases from all the predicted positive cases. Thus, it is useful when the costs of False Positives is high.



$$\text{Precision} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Positive})} = \frac{25}{(25+5)} = \frac{25}{30} = 0.83$$

Precision

2 — Recall: It is the measure of the correctly identified positive cases from all the actual positive cases. It is important when the cost of False Negatives is high.



$$\text{Recall} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})} = \frac{25}{(25+5)} = \frac{25}{30} = 0.83$$

Recall

To summarise the differences between the F1-score and the accuracy,

- Accuracy is used when the True Positives and True negatives are more important while F1-score is used when the False Negatives and False Positives are crucial
- Accuracy can be used when the class distribution is similar while F1-score is a better metric when there are imbalanced classes as in the above case.
- In most real-life classification problems, imbalanced class distribution exists and thus F1-score is a better metric to evaluate our model on.

The solution to our problem, i.e., the optimal (maximum-margin) hyperplane remains unchanged if we remove all training instances but the support vectors. That is why they are given the name 'support vectors'. These training instances can be thought of as 'supporting' or 'holding up' the optimal hyperplane.

Out of the known metrics for validating machine learning models, we choose *Accuracy* and *F1 as* they are the most used in supervised machine learning.

For the accuracy score, it shows the percentage of the true positive and true negative to all data points. So, it's useful when the data set is balanced.

For the f1 score, it calculates the harmonic mean between precision and recall, and both depend on the false positive and false negative. So, it's useful to calculate the f1 score when the data set isn't balanced.