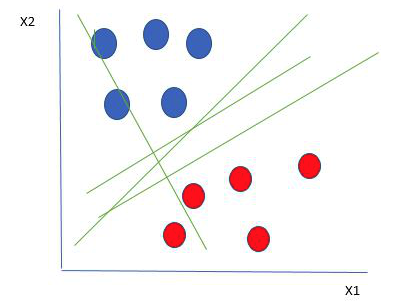
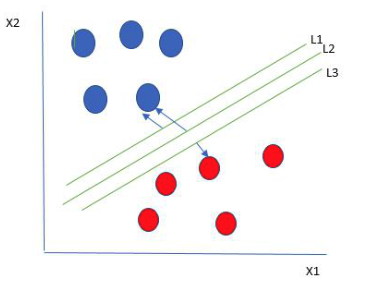
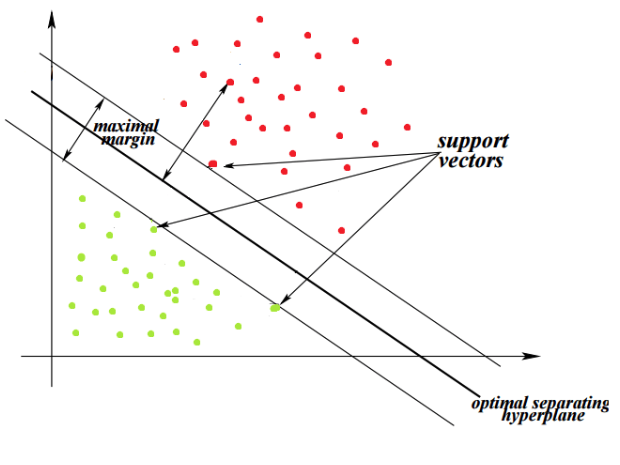
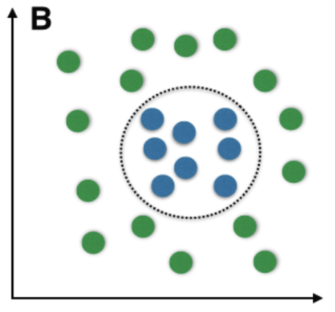
**What is a Support Vector Machine (SVM)?**

A Support Vector Machine is a supervised machine learning algorithm which can be used for both classification and regression problems or other tasks like outliers’ detection.

**SVM Goal:** The objective of SVM algorithm is to **find optimal hyperplane/decision boundary** in an N-dimensional space that distinctly classifies the data points. The decision boundary is the set of points of that separator hyperplane that distinctly classifies the data points. The **optimal hyperplane** is the one that represents the **largest separation or margin between the two classes**.

The **dimension of the hyperplane** depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. In the figures below the hyperplane (decision boundary) is a line as there are two features of x1 and x2.

*Left: Linearly Separable Data Right: Non-Linearly Separable Data*

If the number of input features is three, then the hyperplane becomes a 2-D plane. **It becomes difficult to imagine when the number of features exceeds three and are non-linear separable.**That is why we use Kernel trick.

**SVM Kernel: SVM Kernel** is useful in **non-linear separation problems**. The goal of SVM Kernel is to converts **non separable problem** to **separable problem**. The SVM kernel is a function that takes low dimensional input space and **transforms** it into higher-dimensional space. Based on these **transformations**, SVM finds an optimal boundary between the possible outputs.

**Kernel Selection:**

The kernel is selected based on the type of data and the type of transformation. By default, the kernel is Radial Basis Function Kernel (RBF). When training an SVM with the Radial Basis Function (RBF) kernel, two parameters must be considered: C and gamma.

**Kernel Parameters:**

**Linear Kernel:** Linear SVM is a parametric model, when we can easily separate data with hyperplane by drawing a straight line is Linear SVM It can be used as a dot product between any two observations.

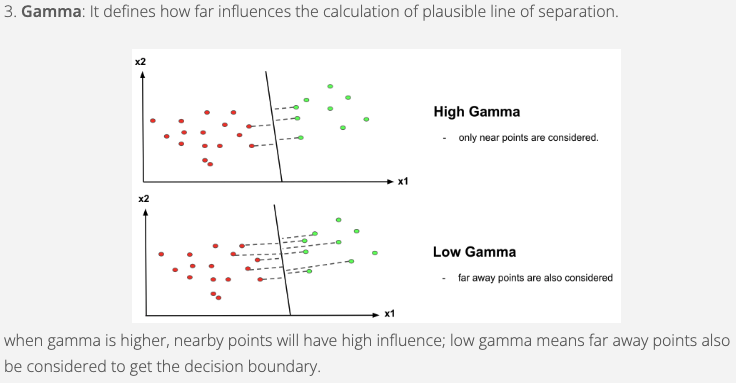
**Non-Linear Kernel:**

### **1)Polynomial Kernel:** It is more generalized form of linear kernel and distinguish curved or nonlinear input space.

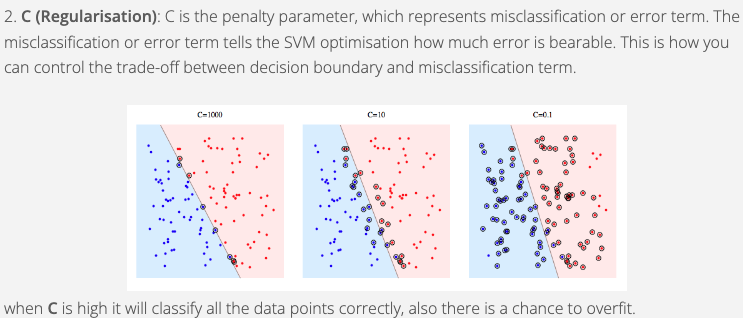
**2)Radial Basis Function (RBF) Kernel:** It is one of the most preferred and used kernel functions in SVM for non-linear data. It is default of SVM that maps input space in indefinite dimensional space. The parameters of RBF are Gamma and C.

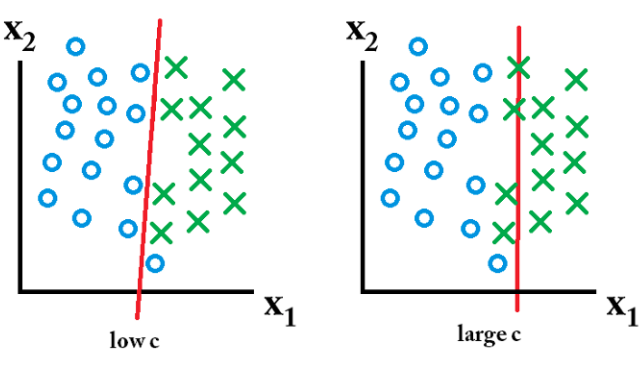
**Gamma parameter**: Gamma is a hyperparameter that ranges from 0 to 1 and it defines how far influences the calculation of plausible line of separation, which in turn affects how tightly the decision boundaries.

When gamma is higher, nearby points will have high influence and this **cause points to be closer together (This causes overfitting)**. Low gamma means far away points are similar and so points are grouped together and have smoother decision boundaries (maybe less accurate) are considered to get the decision boundary **(This causes underfitting)**.



**C parameter:** C is the penalty parameter, which represents misclassification or error term. The misclassification or error term tells the SVM optimization how much error is bearable and so controls the amount of regularization applied to the data. This is how you can control the trade-off between decision boundary and misclassification error. **Large values of C** mean low regularization which in turn causes the training data to fit very well and classify all training examples correctly (may cause **overfitting**). Lower values of C mean higher regularization which makes the decision surface smooth and causes the model to be more tolerant of errors (may lead to underfitting and lower accuracy).





# **What is the Best Value for C and Gamma?**

grid search finds the better parameters of Gamma and C. grid search also finds the best degree for poly kernel also as follows:

svm.SVC(kernel='poly', degree=[2,4,6])

We just need to import GridSearchCV from sklearn.grid\_search, setup a parameter grid and then pass the algorithm, parameter grid and number of cross validations to the GridSearchCV method. svm.SVC(kernel='poly', degree=[2,4,6])

Without GridSearchCV, we would need to loop over the parameters and then run all the combinations of parameters. Then, after a cross-validated result, we would also need to add the code to find the best average CV results across all the combinations of parameters that is very difficult.

GridSearchCV(estimator=SVC(),

param\_grid={'C': [0.1, 1, 10, 100, 1000],

'gamma': [1, 0.1, 0.01, 0.001, 0.0001],

'kernel': ['rbf', 'linear']})

**Example:**

**Linear Kernel Results:**

precision recall f1-score support

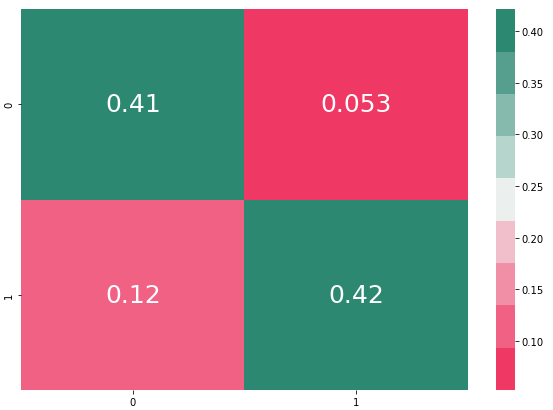
0 0.78 0.89 0.83 35

1 0.89 0.78 0.83 41

accuracy 0.83 76

macro avg 0.83 0.83 0.83 76

weighted avg 0.84 0.83 0.83 76



Non-Linear Kernel Results:

Non-linear (RBF)

[[27 8]

[ 7 34]]

precision recall f1-score support

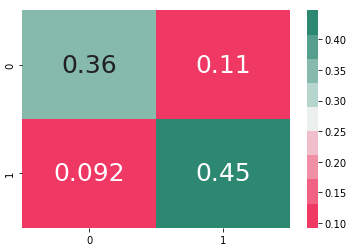
0 0.79 0.77 0.78 35

1 0.81 0.83 0.82 41

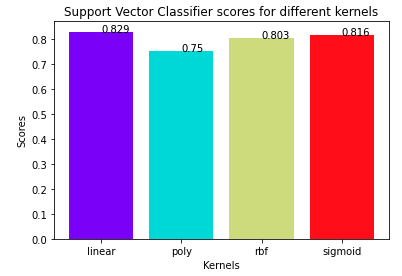
accuracy 0.80 76

macro avg 0.80 0.80 0.80 76

weighted avg 0.80 0.80 0.80 76



**Linear Kernel or Non-Linear Kernel?**

****

Based on the above, we chose the results of Linear Kernel.