The Task:

The purpose of this analysis is to predict the survival of passengers of the Titanic. So, to solve this problem, the nature of the relationship between of each variable with Survived must be understood along with the individual characteristic of each variable. A statistical model is then adopted to further the analysis and arrive at the results and interpretation.

The Dataset:

The dataset contains the following variables:

**PassengerId**

**Survived**

**Pclass**

**Name**

**Sex**

**Age**

**SibSp**

**Parch**

**Ticket**

**Income**

**Fare**

**Cabin**

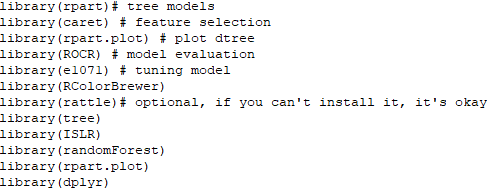
**Embarked**

The statistical model:

Classification is the method of predicting the class of a given input data point. Classification problems are common in machine learning and they fall under the Supervised learning method.

SVM (Support Vector Machine) is a supervised machine learning algorithm which is mainly used to classify data into different classes. Unlike most algorithms, SVM makes use of a hyperplane which acts like a decision boundary between the various classes. SVM can be used to generate multiple separating hyperplanes such that the data is divided into segments and each segment contains only one kind of data.

Setting up the R model by loading the required libraries:



Data:

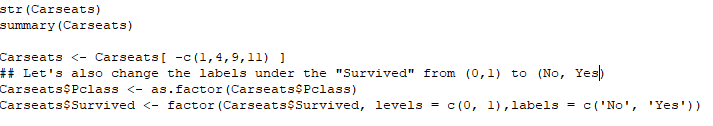
In the next step, the data is read into the R environment from the file.

setwd("C:\\Users\\ADMIN\\Desktop\\R Models\\Decision Tree")

Carseats <- read.csv("Titanic.csv")

Data selection and data type modification:

In the given dataset, the columns which are named as “PassengerId”, “Name”, “Ticket” and “Fare” contains customerid details, names of the passengers, ticket details and fare are of no relevance to our model and is thus excluded. Two of the variables, Pclass and Survived are converted to factors and Survived was in binary form so we converted 0 = “No” and 1 = “Yes”.



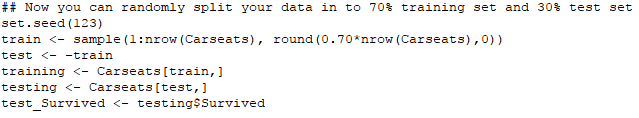
Checking for missing values:

After the data has been cleaned off all the outliers, it is then checked for any missing values in the following manner:

264 missing values were found and has been removed.

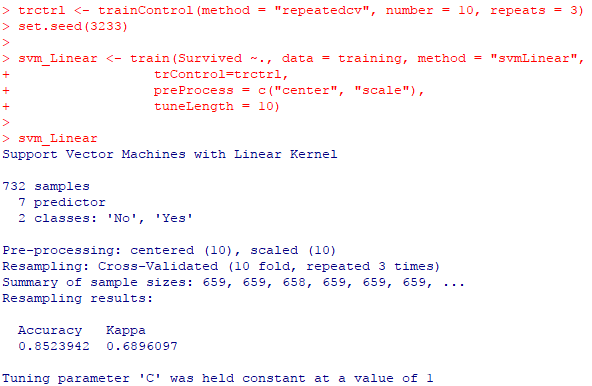
Splitting the data:

The data is then split into two parts. 70 percent of the data is split into ‘development’ and the remaining 30 percent is named ‘validation’. The development part is for training the decision tree model and the validation part is for testing the robustness of the model.



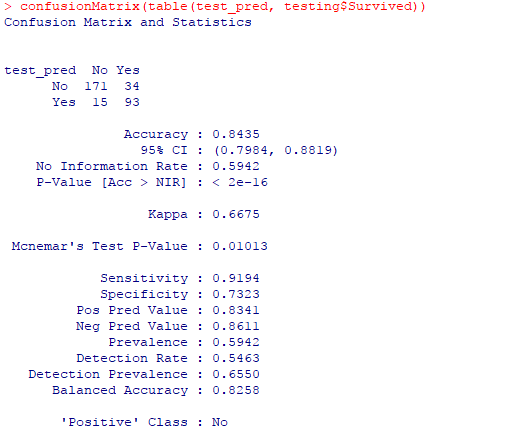
**Training the SVM model with Linear Kernel:**

So, you start of by drawing a random hyperplane and then you check the distance between the hyperplane and the closest data points from each class. These closest data points to the hyperplane are known as support vectors. And that’s where the name comes from, support vector machine. The hyperplane is drawn based on these support vectors and an optimum hyperplane will have a maximum distance from each of the support vectors. And this distance between the hyperplane and the support vectors is known as the margin. To sum it up, SVM is used to classify data by using a hyperplane, such that the distance between the hyperplane and the support vectors is maximum.

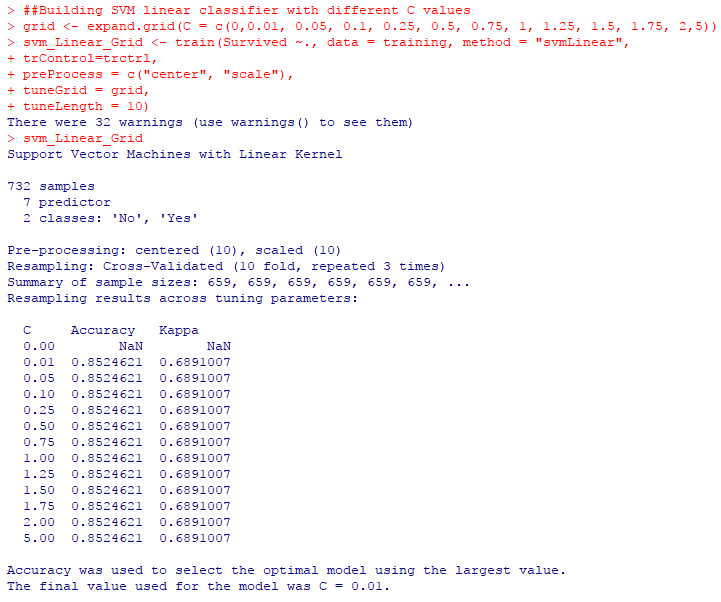


**C Parameter** is used for controlling the outliers — low **C** implies we **are** allowing more outliers, high **C** implies we **are** allowing fewer outliers. High **C** (cost) means the cost of misclassification is increased. This means a flexible kernel **will** become more squiggly to avoid misclassifying observations in the training set.

A validation of training set has been ran using confusion matrix.

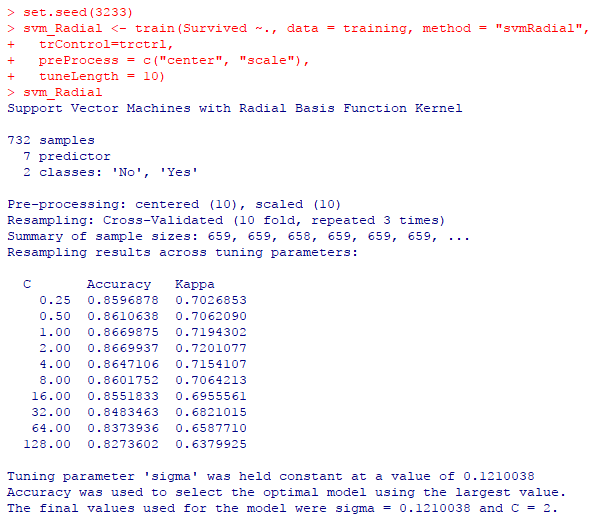


**Running** the model:

We have trained the machine using the below algorithm in such a way that it will check all the possible accuracies by evaluating over-all combinations of C. The final value for the model was C = 0.01 because at C = 0.01 the accuracy is 85%. which is the highest among all other values.

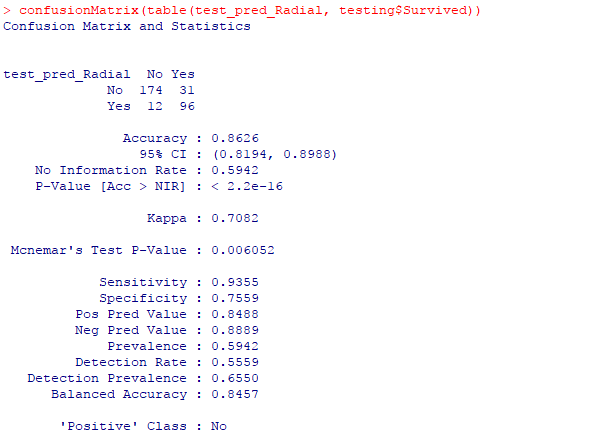
**Training the SVM model with Non-Linear Kernel(RBF):**

Until this point, we were plotting our data on 2-dimensional space. So, we had only 2 variables, x and y. A simple trick would be transforming the two variables x and y into a new feature space involving a new variable z. Basically, we’re visualizing the data on a 3-dimensional space. When we transform the 2D space into a 3D space we can clearly see a dividing margin between the 2 classes of data. And now we can go ahead and separate the two classes by drawing the best hyperplane between them. This sums up the idea behind Non-linear SVM. This is also known as **Radial Base Function(RBF)**.

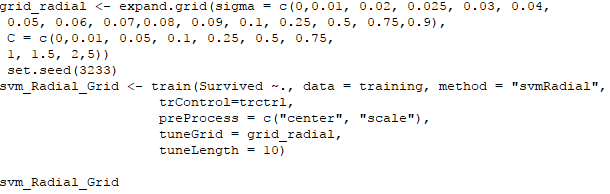
****

**Gamma** is the parameter for a nonlinear support vector machine (**SVM**) with a Gaussian radial basis function kernel. The **gamma** parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The **gamma** parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors. The final value for the model was C = 2 and sigma(gamma) = 0.12 where the accuracy is 86.7%.

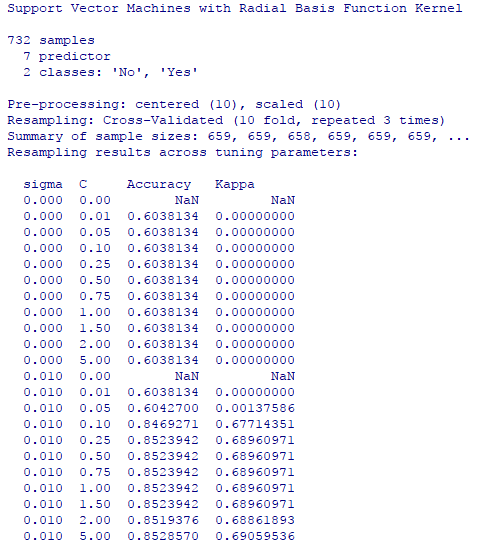
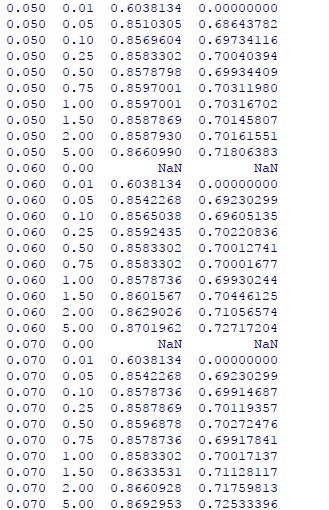
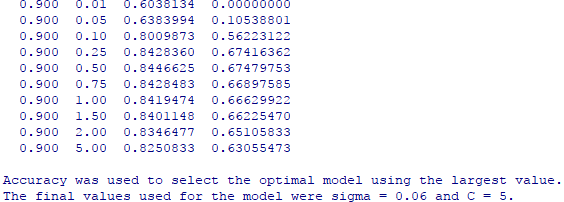
A validation of non-linear training set has been ran using confusion matrix.

****

**Testing & tuning our classifier with different values of C & sigma:**



We were getting an accuracy of **86.7%**. So, in this case with values of C = 2 & sigma= 0.12, we are getting good results.  
Let’s try to test & tune our classifier with different values of C & sigma. We will use grid search to implement this.  
grid\_radial data frame will hold values of sigma & C. Value of grid\_radial will be given to train() method’s tuneGrid parameter.

The final model has shown the best accuracy to be 87% at sigma(gamma) = 0.06 and C = 5.

**Validation of the final model:**

Checking the confusion matrix to understand how good the model is with a tuned SVM.

