In this research I would be demonstrating how we can optimize banks efforts to increase their operations and reach targets in a specified time in India.

I will implement the SVM classifier on the bank customers data. Below are the points that I would like to demonstrate about this algorithm.

* Why only SVM?
* SVM functionality
* Pre – requisites of SVM
* Preprocessing
* Deferential and inferential statistics
* Training and testing the model
* Interpret the results using various metrics

1. **Why only SVM: -**

The SVM finds the linear margin as large as possible that separates these two regions. The marginal separators rest on the outpost points that are right on the front line of their respective regions. These points, marked as two bold triangles and one bold circle in the picture below, are named the ‘support vectors’ as they are supporting the separation boundary lines. In fact, the SVM learning task fully consists of determining these support vector points and the margin distance that separates the regions. After training all other non-support points are not used for prediction.

We have many classification techniques like logistic regression, Decision Tree, Support Vector Machines, Naïve Bayes and K-NN to make use in this current project, but the reason mainly going to SVM is because of its memory efficient, various kernels availability and best parameters to tune and to achieve best model for a better outcomes. It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient

About SVM Kernels: - There are mainly four kernels in wide range of use. They are Linear, Polynomial, radial and sigmoid kernels.

1. **Linear kernel: -** It can be used if the data that we have is linear and separable in lower dimensional spaces. Usually data which will get from external sources won’t have direct relationships with respect to the response variable that means it is nonlinear. To overcome this, we have nonlinear kernels like polynomial, radial basis and sigmoid. Which transforms the attributes from lower dimension to higher dimensions where it is said to be linearly separable. It is defined as

k(x, y) = x^\top y

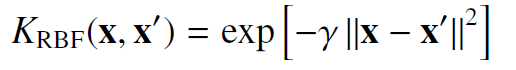
Where x and y are the input vectors or features.

**b. RBF Kernel: -**

The Radial basis function kernel, also called the RBF kernel, or Gaussian kernel,

is a kernel that is in the form of a radial basis function (more specifically, a Gaussian

function). The RBF kernel is defined as



Where **γ** is a parameter that sets the “spread” of the kernel.

It transforms the features from lower dimensional space to higher dimensional space where the features are believed to be linearly separable. It is the default kernel that comes in e1071 packages and be used for most of the classification problems.

1. **Polynomial Kernel: -**

The function [polynomial kernel](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.polynomial_kernel.html#sklearn.metrics.pairwise.polynomial_kernel) in SVM computes the degree-d polynomial kernel between two vectors. The polynomial kernel represents the similarity between two vectors. Conceptually, the polynomial kernels consider not only the similarity between vectors under the same dimension, but also across dimensions. When used in machine learning algorithms, this allows to account for feature interaction.

The polynomial kernel is defined as:

k(x, y) = (\gamma x^\top y +c_0)^d

where:

* x, y are the input vectors
* d is the kernel degree

If c_0 = 0the kernel is said to be homogeneous.

**d. Sigmoid Kernel: -**

The function [sigmoid kernel](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.sigmoid_kernel.html#sklearn.metrics.pairwise.sigmoid_kernel) computes the sigmoid kernel between two vectors. The sigmoid kernel is also known as hyperbolic tangent, or Multilayer Perceptron (because, in the neural network field, it is often used as neuron activation function). It is defined as:

k(x, y) = \tanh( \gamma x^\top y + c_0)

where:

* x, y are the input vectors
* \gammais known as slope
* c_0is known as intercept

**2. SVM functionality: -**

Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification and regression problems. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a coordinate. Then, we perform classification by finding the better hyper-plane that segregates the two classes very well (look at the below snapshot).

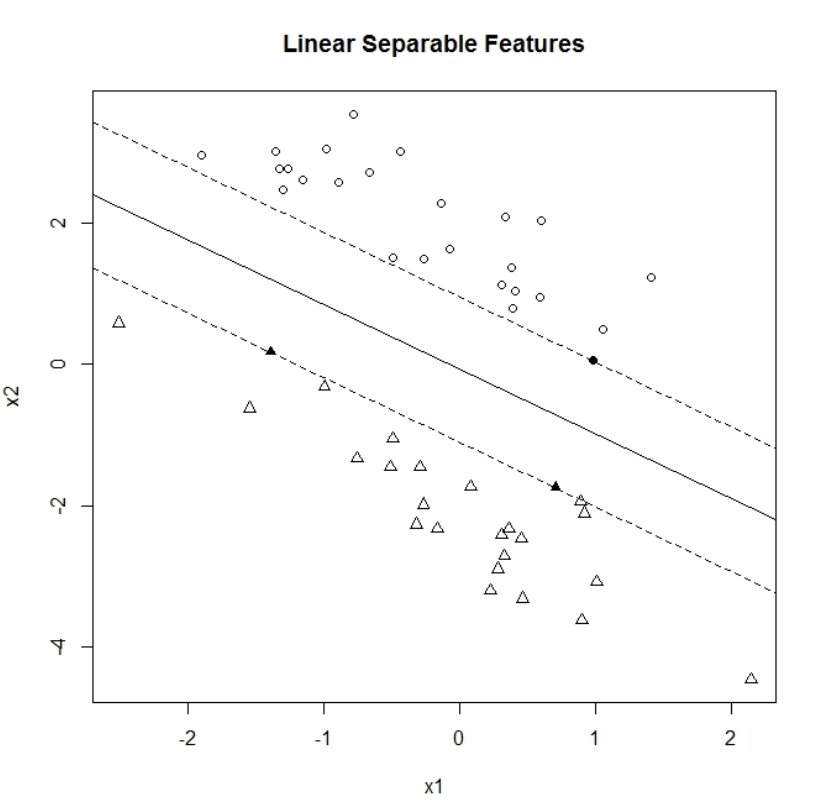
[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_1.png)

Support Vectors are simply the co-ordinates of from the observation we have trained. We call that separating line as a plane when only two co-ordinates exist. For n- dimension co-ordinates it is called as Hyper-Plane.

It follows mainly two thumb rules while selecting the hyperplane.

1. The hyper plane should better segregate the classes without any misclassification.
2. The soft margin should be as wide as possible.

From the below screen shot, the soft margin is the distance from plane which is in the thick line to the dotted line. The co-ordinates that lies on the dotted lines are called support vectors. Once the model is built on SVM, our model uses only support vectors for classification of the new or unseen data points. Hence, SVM requires less memorization to build a model and deploy in the



1. **Pre-requisites of SVM: -**
2. All the input features should be in same scale that is all should be standardized.
3. All the features should be independent of each other.
4. The input features should be normalized.
5. As most of the ML algorithms woks on numerical attributes, it is one of them.
6. **Preprocessing: -**

Till now we have learnt about SVM algorithm which is to be used in the problem. Before we discuss about pre-processing which is a mandate step in any machine learning algorithm, we should have data related to bank customers to achieve the purpose of this research paper.

I have found the Indian banks customers data with below features and let’s hope this will helps our purpose here.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| ID | Customer ID |
| Age | Customer's age in completed years |
| Experience | #years of professional experience |
| Income | Annual income of the customer ($000) |
| ZIPCode | Home Address ZIP code. |
| Family | Family size of the customer |
| CCAvg | Avg. spending on credit cards per month ($000) |
| Education | Education Level. 1: Undergrad; 2: Graduate; 3: Advanced/Professional |
| Mortgage | Value of house mortgage if any. ($000) |
| Personal Loan | Did this customer accept the personal loan offered in the last campaign? (Target attribute) |
| Securities Account | Does the customer have a securities account with the bank? |
| CD Account | Does the customer have a certificate of deposit (CD) account with the bank? |
| Online | Does the customer use internet banking facilities? |
| CreditCard | Does the customer use a credit card issued by UniversalBank? |

**Actual Implementation of a model in R: -**

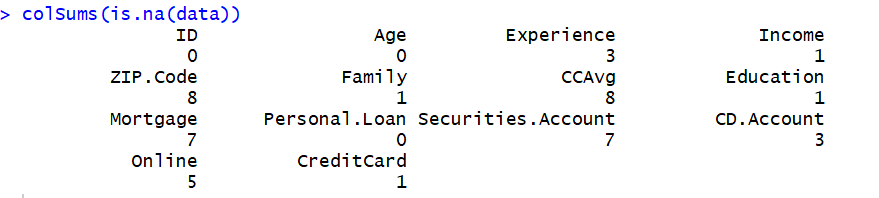
1. **Handling Missing Values: -**

A missing value can signify several different things. Perhaps the field was not applicable, the event did not happen, or the data was not available. It could be that the person who entered the data did not know the right value, or that field may allow Null values.

Therefore, Analysis Services provides two distinctly different mechanisms for imputing missing values. The first method controls the handling of nulls at the level of the mining structure. The second method differs in implementation for each algorithm, but generally defines how missing values are processed and counted in models that permit null values.

1. **Checking the number of NA's column wise if exixts**

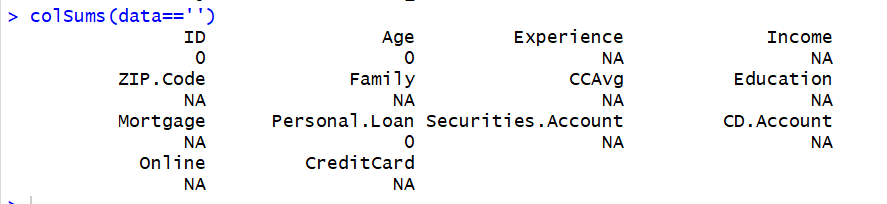
*->colSums(is.na(data))*



From the above screen shot, we could see that NA’s existed for Experience, Income, ZIP.Code, Family, CCAvg, Education, Mortgage, Securities.Account, CD.Account, Online, CreditCard which are need to be Imputed.

1. **Checking the empty Values**

*->colSums(data==’ ’)*



From the above screen shot, we could see that empty values are zero for all the variables. Hence, no need of imputation for the empty values.

1. **Imputing NA’s**

In R, we have many libraries to impute missing values, one such library is MICE.

It replaces NA's with mean of the variable if it is a numeric and default value if it is a categorical. Here it imputes values for Family, CCAvg, Education, Mortage, Securities.Account, CD.Account, Online and Creditcard features.

*->install.packages('mice')*

**About MICE library: -**

MICE (Multivariate Imputation via Chained Equations) is one of the commonly used packages by R users. Creating multiple imputations as compared to a single imputation (such as mean) takes care of uncertainty in missing values.

MICE assume that the missing data are Missing at Random (MAR), which means that the probability that a value is missing depends only on observed value and can be predicted using them. It imputes data on a variable by variable basis by specifying an imputation model per variable.

For example: Suppose we have y1, y2….yk variables. If y1 has missing values, then it will be regressed on other variables y2 to yk. The missing values in y1 will be then replaced by predictive values obtained. Similarly, if y2 has missing values, then y1, y3 to yk variables will be used in prediction model as independent variables. Later, missing values will be replaced with predicted values.

By default, linear regression is used to predict continuous missing values. Logistic regression is used for categorical missing values. Once this cycle is complete, multiple data sets are generated. These data sets differ only in imputed missing values. Generally, it’s considered to be a good practice to build models on these data sets separately and combining their results.

#**Selecting the one of the datasets of 5 generated *from the above.***

*->library('mice')*

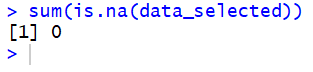
*->md.pattern(ActualData)*

*->imputed\_Data <- mice(ActualData, m=5, maxit = 50, method = 'pmm', seed = 500)*

*->data\_Selected<- complete(imputed\_Data,2)*

From the below screen shot, we can say that the dataset with name ‘data\_selected’ has no NA’s and empties. This dataset is enough for further data analysis.

*->Sum(is.na(‘data\_selected’)*



**ii. Removing Identities: -**

As a part of normalization in traditional database system, most of the tables contains primary keys and foreign keys which are of no use in data mining and building a model. Hence, need to remove such attributes. Here in our dataset such features are ID and Zip.code.

These might cause lot of noise and enough to misguide our model resulted in incorrected objective function.

1. **Removing ID and ZIP columns as they are identities and not required**

**About dplyr library: -**

dplyr is a package for data manipulation, written and maintained by Hadley Wickham. It provides some great, easy-to-use functions that are very handy when performing exploratory data analysis and manipulation

*->Install.packages(‘dplyr’)*

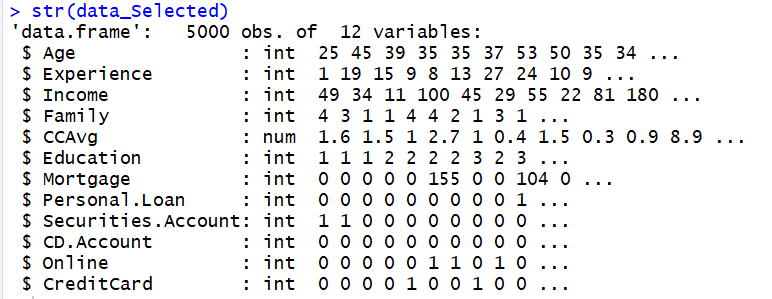
*-> data\_Selected <- data\_Selected %>% select(-c(ID, ZIP.Code))*

1. **Feature Encoding: -**

Feature encoding is a part of feature engineering and necessary for data preprocessing. Since most of the algorithms accept numerical attributes to build a model. Hence, we need to check if there any such variables.

From the below screen shot, we could see most of the variables are integers and numerical. We need to identify if there are any categorical features based on the number of factors and uniqueness of the variable.

*-> str(data\_Selected)*



From the above screenshot, could see Experience, Income and Mortgage have high scale integer values and these can be converted into numericals.

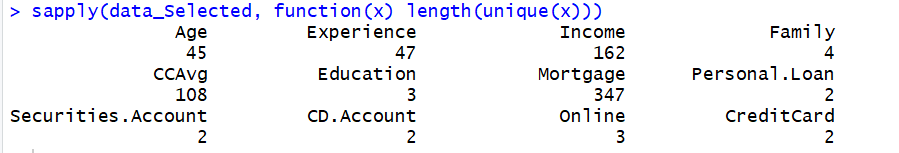
*->data\_Selected$Experience<- as.numeric(data\_Selected$Experience)*

*-> data\_Selected$Income <- as.numerica(data\_Selected$Income)*

*-> data\_Selected$Mortgage <- as.numeric (data\_Selected$Mortgage)*

Rest of the variables can be converted into factors based on the number of unique values.

*->sapply(data\_Selected, function(x) length(unique(x)))*



From the above screen shot, we could see family, Education, Personal loan, Securities. Account, CD.Account, Online, CreditCard have unique values less than 5. Hence, we can convert them as factors and then dummify.

*->data\_Selected$Personal.Loan -> as.factor(data\_Selected$Personal.Loan)*

*->data\_Selected$Family -> as.factor(data\_Selected$Family)*

*->data\_Selected$Securities.Account -> is.factor(data\_Selected$Securities.Account)*

*->data\_Selected$CD.Account -> as.factor(data\_Selected$CD.Account)*

*->data\_Selected$Online -> as.factor(data\_Selected$Online)*

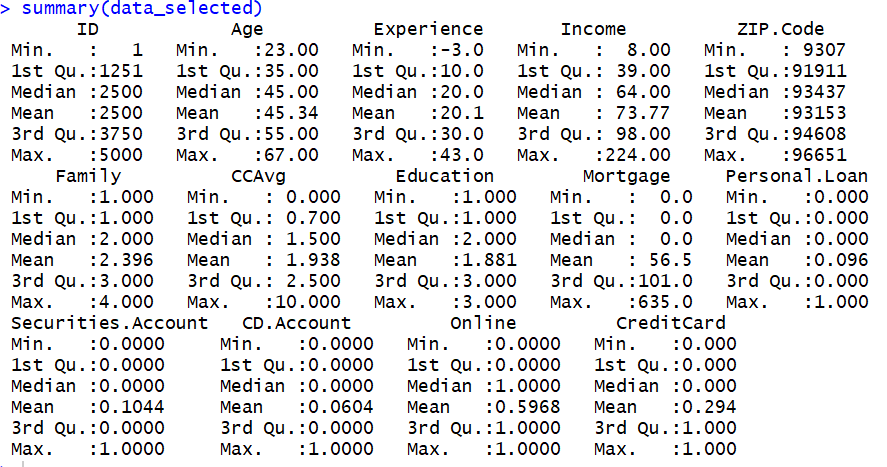
*->data\_Selected$CreditCard -> as.factor(data\_Selected$CreditCard)*

1. **Descriptive Statistics and Inferential: -**

**Measure of Central Tendency: -**

These are ways of describing the central position of a frequency distribution for a group of data. In this case, the frequency distribution is simply the distribution and pattern of marks scored by the 100 students from the lowest to the highest. We can describe this central position using several statistics, including the mode, median, and mean.

*-> summary(data\_selected)*

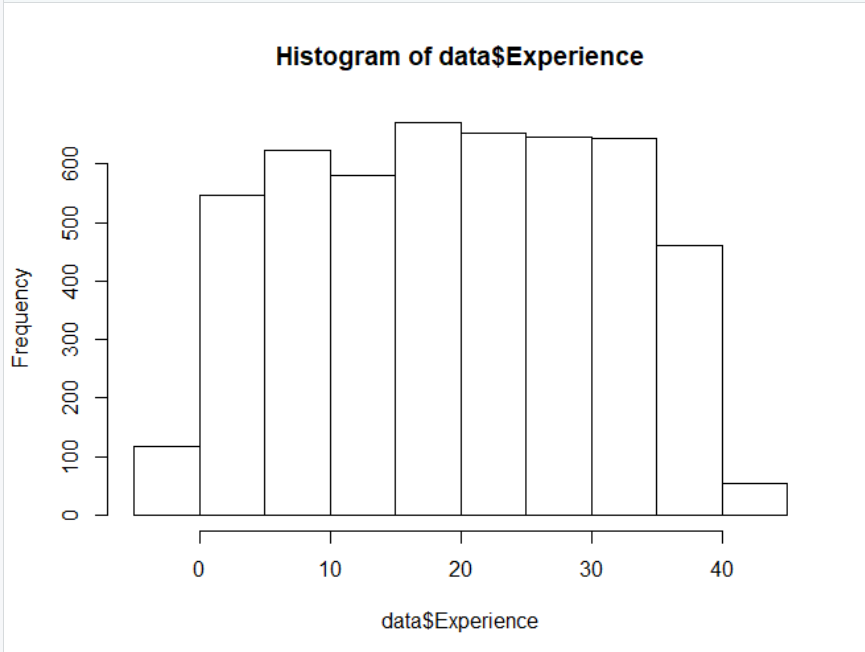


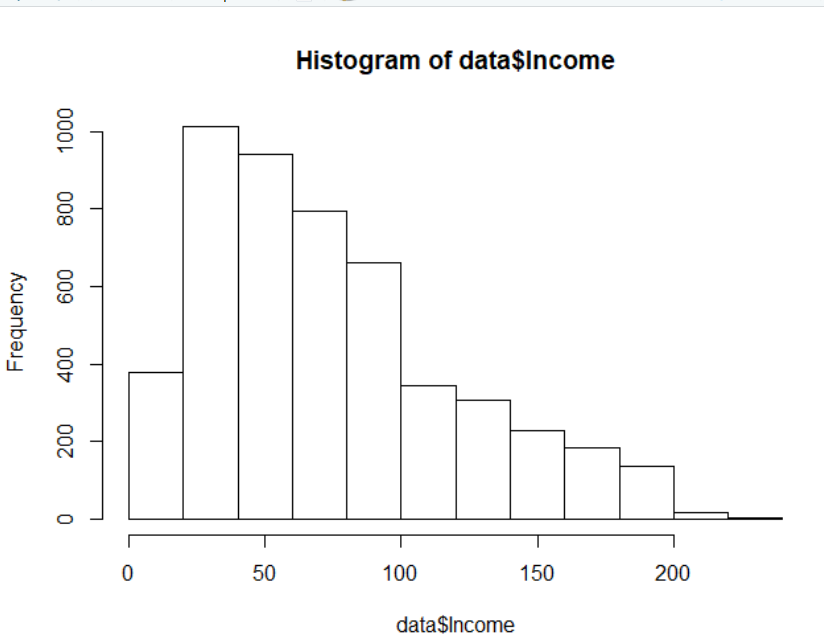
**Measure of Spread: -**

Measures of spread help us to summarize how spread out these scores are. To describe this spread, a number of statistics are available to us, including the range, quartiles, absolute deviation, variance and [standard deviation](https://statistics.laerd.com/statistical-guides/measures-of-spread-standard-deviation.php).

For any dataset, the data distribution is prior check as most of the algorithms prefer the data ingested to be in normally distributed. Hence, Plotting the histograms to check if there is any skewness and kurtosis.

As we cannot remove skewness and kurtosis from the original data, and can mitigate them by doing some transformations like log, square root etc. here in our case, the skewness for income and Experience variable is negligible can model predict well without any transformations.





1. **Training and Testing the Model: -**

To train and test the SVM, first we need to split the data set into two set i.e., training set for training the model and test set for validating the model. Below R code will help us to perform this task.

**#Splitting the data into test and train for training and validation.**

*->library(caTools)*

*->set.seed(1)*

*->sample<- sample.split(data\_Selected$Personal.Loan, SplitRatio = .75)*

*->Training\_set <- subset(data\_Selected, sample==T)*

*->Test\_set<- subset(data\_Selected, sample==F)*

We have split the data into training and testing sets. Let’s build the model using training set and validate it using test set.

**#Building the model**

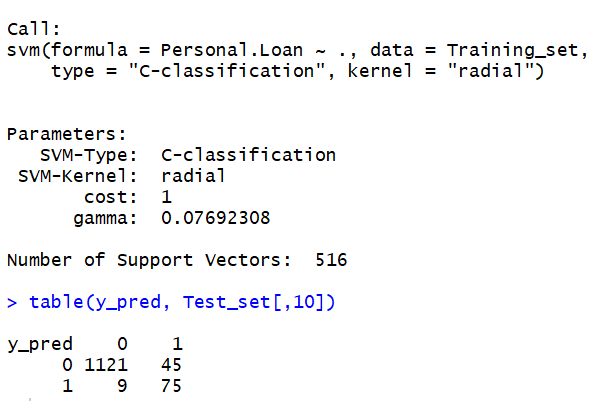
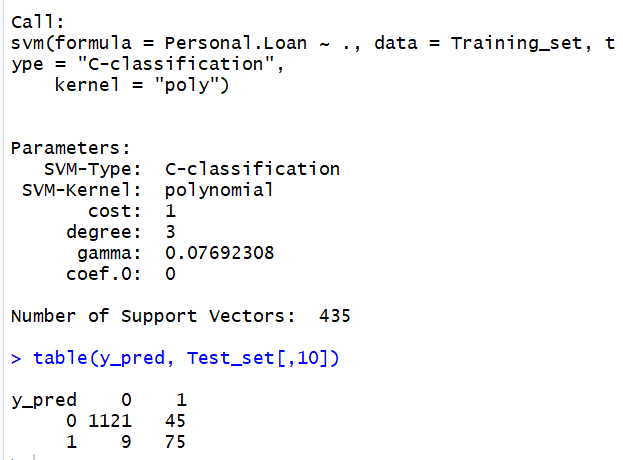
*->library('e1071')*

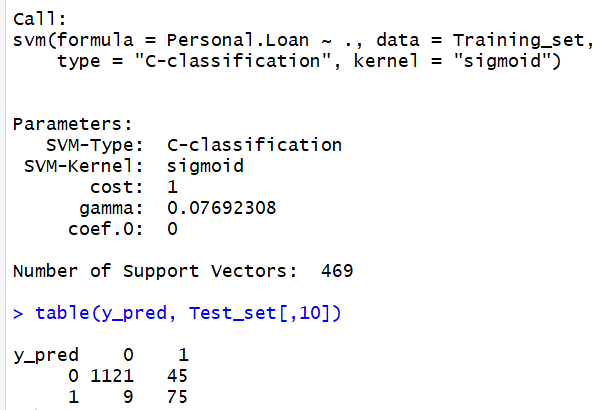
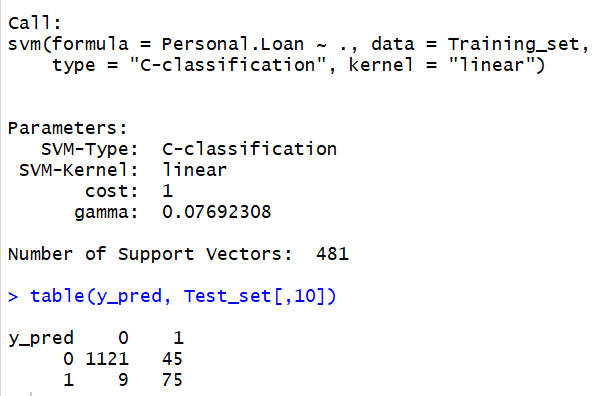
*->model<- svm (formula= Personal.Loan~.,data = Train\_data,*

*type='C-classification',*

*kernel='linear', probability= TRUE, gamma=0.1, cost=10)*

As we have built the model using the default using default parameter values. Let’s interpret the model using the below R code to select the best kernel.





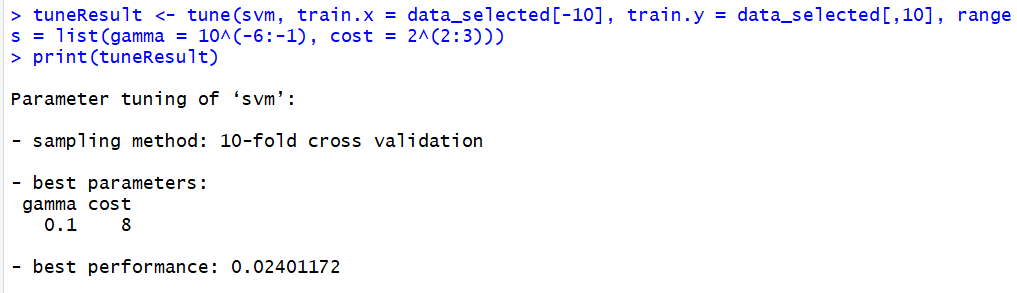
From the above four screen shots that are taken for four different kernels, we could see that all the models have same accuracy and misclassification rate. How ever the model that was built using radial basis required very less support vectors compared to other models. Hence, lets select confirm RBF kernel for our model.

Now the lets fine tune our model to achieve the best results as we have used only the default parameters till now.

*-> tuneResult <- tune(svm, train.x = data\_selected[-10], train.y = data\_selected[,10], ranges = list(gamma = 10^(-6:-1), cost = 2^(2:3)))*

*->print(tuneResult)*

Below screen shot is the result after performing hyper parameter tuning which is also called performance tuning.



Now let’s store the above best parameter model into one reference and predict test values using it.

*->tunedModel <- tuneResult$best.model*

Let’s predict using the below R code and calculate the accuracy and misclassification rate using confusion matrix.

**#Predicting on test data**

*->y\_pred <- predict (tunedModel, Test\_Data[-8])*

**7.Interpret the results using confusion Matrix and ROC: -**

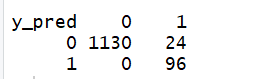
Since it’s a classification task, we take reference as confusion matrix statistics and receiver Characteristics curve for selecting the best model.

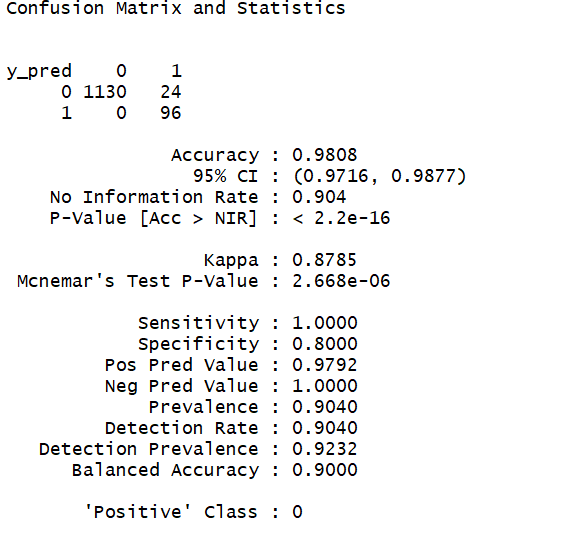
**#Confusion Matrix to see the accuracy**

*->library(caret)*

*->confusionMatrix(table(y\_pred, Test\_Data[,8]))*

From the below confusion matrix and statistics, we can see that for 0, the misclassification is 24, i.e. 24variables were predicted 1 by the model whereas their value was 0. Similarly, for 1, misclassification is 0. The model accuracy is 98.08% which is far good after performing hyper parameter tuning and selecting the best hyper parameters.





The above results tell that out of total 1250 customers, only 96 bank customers high probability of taking loans offered by banks. Hence, Banks can directly target these customers and can offer loans.

**Receiver Operating Characteristics Curve: -**

The ROC curve is used for calculating AUC (Area under the curve) which is used for measuring the performance of a binomial classifier. The ROC is a curve generated by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings while the AUC is the area under the ROC curve. As a rule of thumb, a model with good predictive ability should have an AUC closer to 1 (1 is ideal) then to 0.5.

**## ROC Curve and calculating the area under the curve (AUC)library**

*->library (ROCR)*

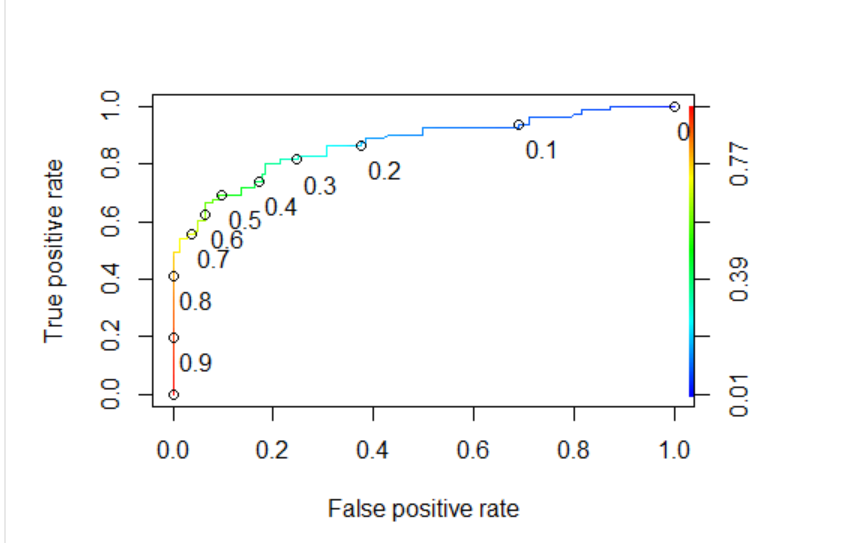
*->predictions <- predict(model, newdata=Test\_Data[-10], type='response')*

*->ROCRpred <- prediction(as.numeric(predictions), Test\_Data$Personal.Loan)*

*->ROCRperf <- performance(ROCRpred, measure = "tpr", x.measure = "fpr")*

*plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7), print.cutoffs.at = seq(0,1,0.1))*

The curve in the below characteristics shows that true positive rate is increasing as false positive rate is increasing, it slightly equal to 1 which is ideal. Hence, the model we have built for the prediction of customers who has high probability of accepting bank offers is looking good and can deploy.



**Conclusion: -**

In this research, we can conclude thatIncome, Experience, Family Size, Mortgage, Credit Card plays important role in India in deciding whether customer accepts a loan offered by Banks.

**References:**

<http://eeecon.uibk.ac.at/~zeileis/papers/Ensemble-2005.pdf> https://en.wikipedia.org/wiki/Support\_vector\_machine