A Project report on

A data driven approach to predict the success of bank telemarketing.

In Foundation of Financial Data Sciences (FE- 582- GROUP 9)

By

NIKHIL NIRHALE
VITEJ BARI
SENUJ JAIN
DEEPANSHU TYAGI

TABLE OF CONTENTS

ABSTRACT

We propose a data mining (DM) approach to predict the success of telemarketing calls for selling bank long-term deposits. A Portuguese retail bank was addressed, with data collected from 2008 to 2013, thus including the effects of the recent financial crisis. We analysed a large set of 150 features related with bank client, product and social-economic attributes. A semi-automatic feature selection was explored in the modelling phase, performed with the data prior to July 2012 and that allowed to select a reduced set of 22 features. We are comparing four DM models: logistic regression, decision trees (DTs), neural network (NN) and support vector machine. We will be selecting the best model for this data using advanced metrics using the Rminer Package.

INTRODUCTION

Marketing selling campaigns constitute a typical strategy to enhance business. Companies use direct marketing when targeting segments of customers by contacting them to meet a specific goal. Centralizing customer remote interactions in a contact centre eases operational management of campaigns.

Technology enables rethinking marketing by focusing on maximizing customer lifetime value through the evaluation of available information and customer metrics, thus allowing us to build longer and tighter relations in alignment with business demand.

Decision support systems (DSSs) use information technology to support managerial decision making. There are several DSSs sub-fields, such as personal and intelligent DSSs. Personal DSSs are related with small-scale systems that support a decision task of one manager, while intelligent DSSs use artificial intelligence techniques to support decisions. Another related DSS concept is Business Intelligence (BI), which is an umbrella term that includes information technologies, such as data warehouses and data mining (DM), to support decision making using business data. DM can play a key role in personal and intelligent DSSs, allowing the semi-automatic extraction of explanatory and predictive knowledge from raw data. Classification is the most common DM task and the goal is to build a data driven model that learns an unknown underlying function that maps several input variables, which characterize an item (e.g., bank client), with one labelled output target (e.g., type of bank deposit sell: “failure” or “success”).

In this project, we propose a personal and intelligent DSS that can automatically predict the result of a phone call to sell long term deposits by using a DM approach. Such DSS is valuable to assist managers in prioritizing and selecting the next customers to be

contacted during bank marketing campaigns. For instance, by using a Lift analysis that analyses the probability of success and leaves to managers only the decision on how many customers to contact. Consequently, the time and costs of such campaigns, would be reduced. Also, by performing fewer and more effective phone calls, client stress and intrusiveness would be diminished.

This research focus on targeting through telemarketing phone calls to sell long-term deposits. Within a campaign, the human agents execute phone calls to a list of clients to sell the deposit (outbound) or, if meanwhile the client calls the contact-centre for any other reason, he is asked to subscribe the deposit (inbound). Thus, the result is a binary unsuccessful or successful contact.

This study considers real data collected from a Portuguese retail bank, from May 2008 to June 2013, in a total of 49,732 phone contacts. For evaluation purposes, a time ordered split was initially performed, where the records were divided into training (four years) and test data (one year). The training data is used for feature and model selection and includes all contacts executed up to June 2012, in a total of 45,211 examples. The test data is used for measuring the prediction capabilities of the selected data-driven model, including the most recent 4521 contacts, from July 2012 to June 2013.

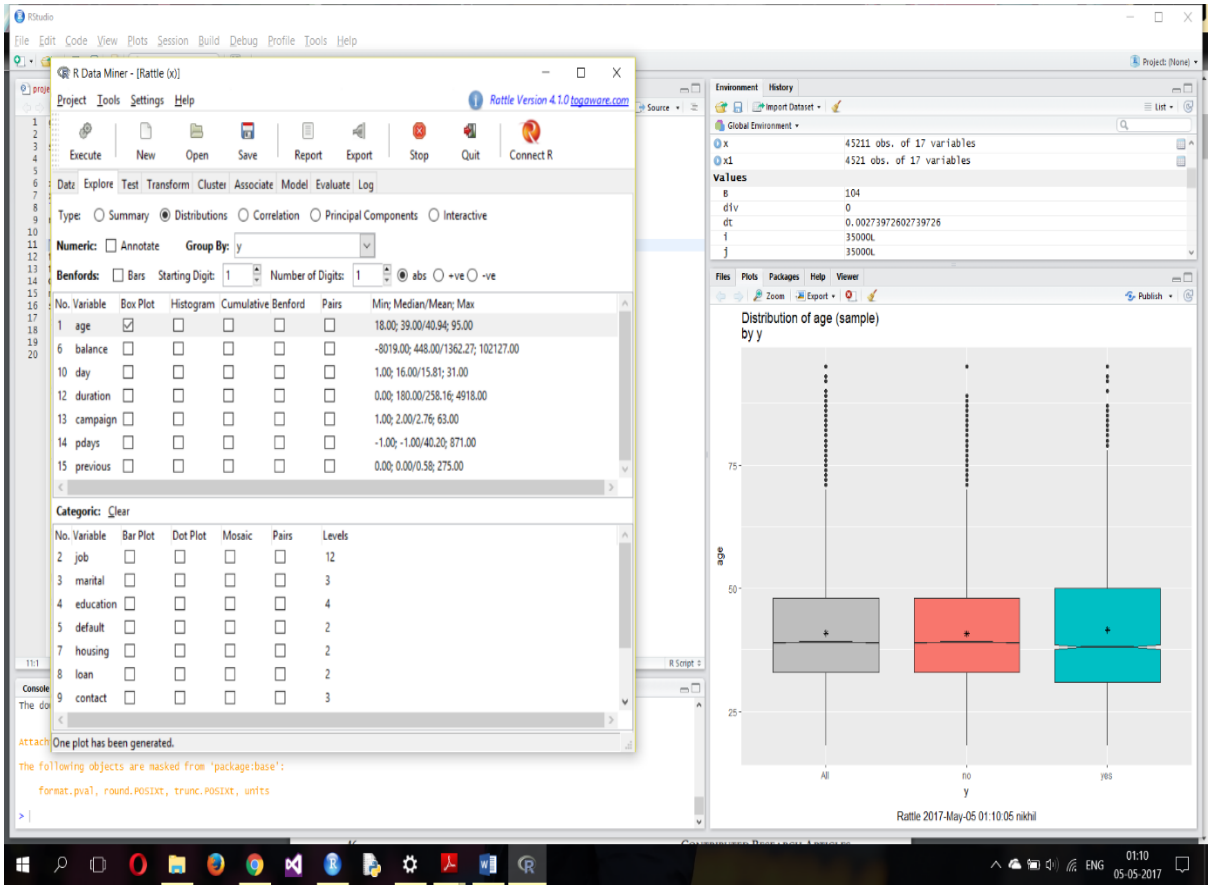
DATA SUMMARY AND EDA ALREADY INCLUDED IN PROGRESS REPORT.

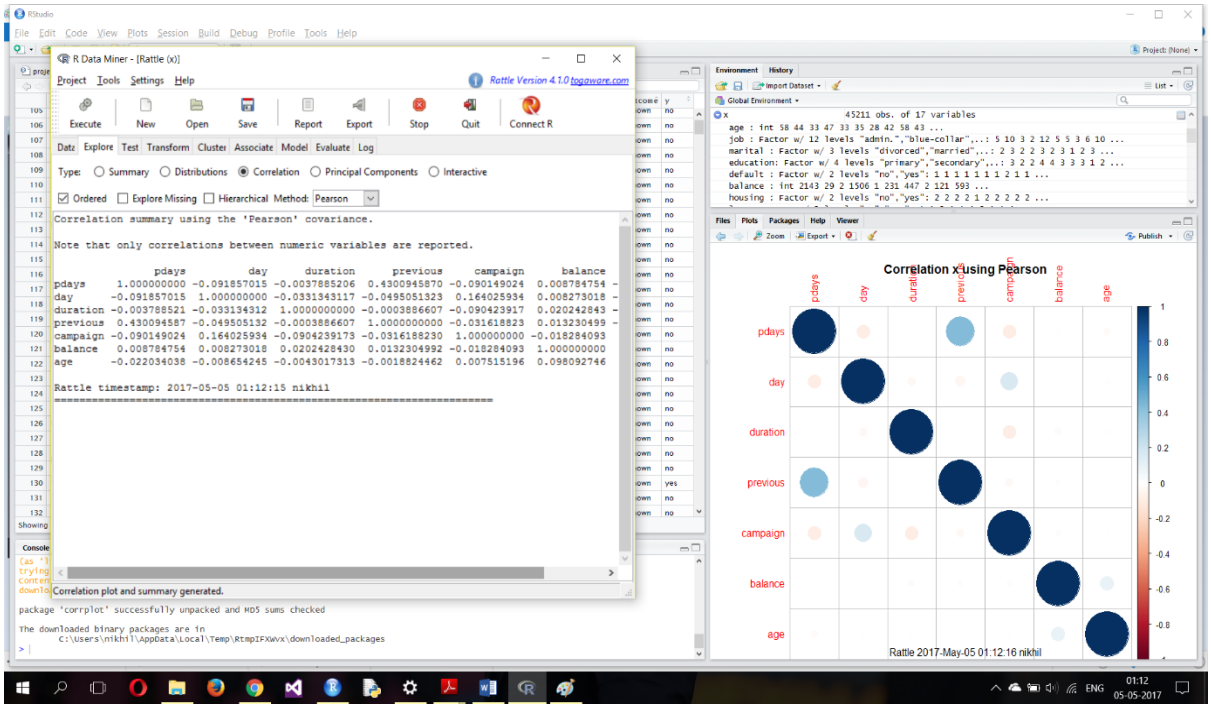
PACKAGES

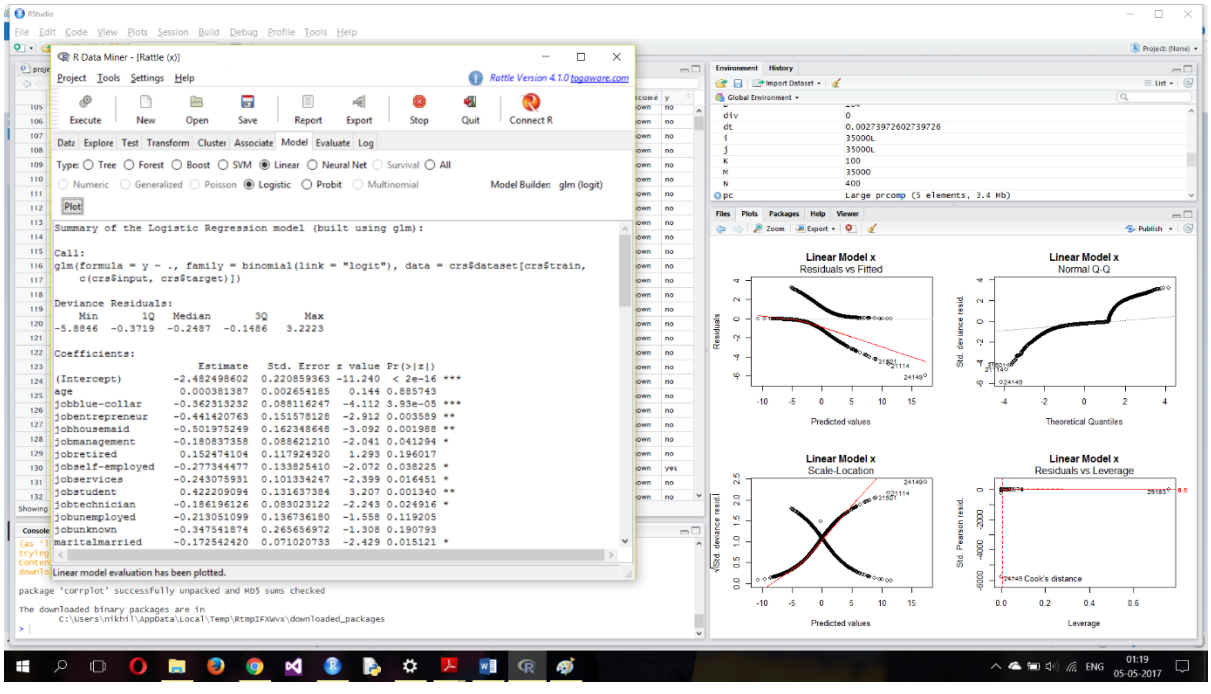
1.RATTLE

Rattle (the R Analytical Tool to Learn Easily) is a graphical data mining application written in and providing a pathway into (Williams,2009b). It has been developed specifically to ease the transition from basic data mining, as necessarily offered by GUIs, to sophisticated data analyses using a powerful statistical language.

In our Project we have used this package to conduct some basic EDA to look for outliers and missing values.We have conducted this analysis in our Progress report to conclude that our data does not include any missing values and effect of outliers is specifically not very relevant in this case. Below are some screenshots of the RATTLE GUI.




R Data Miner - [Rattle (x)]
Project Tools Settings Help
Rattle Version 4.1.0 togaware.com
Execute New Open Save Report Export Stop Quit Connect R
Data Explore Test Transform Cluster Associate Model Evaluate Log
Type: Summary Distributions Correlation Principal Components Interactive
Ordered Explore Missing Hierarchical Method: Pearson
Correlation summary using the 'Pearson' covariance.
Note that only correlations between numeric variables are reported.
pdays day duration previous campaign balance
pdays 1.000000000 -0.091857015 -0.0037885206 0.4300945870 -0.090149024 0.008784754
day -0.091857015 1.000000000 -0.0331343117 -0.0495051323 0.164025934 0.008273018
duration -0.003788521 -0.033134312 1.0000000000 -0.0003886607 -0.090423917 0.020242843
previous 0.430094587 -0.049505132 -0.0003886607 1.0000000000 -0.031618823 0.013230499
campaign -0.090149024 0.164025934 -0.0904239173 -0.0316188230 1.000000000 -0.018284093
balance 0.008784754 0.008273018 0.0202428430 0.0132304992 -0.018284093 1.000000000
age -0.022034038 -0.008654245 -0.0043017313 -0.0018824462 0.007515196 0.098092746
Rattle timestamp: 2017-05-05 01:12:15 nikhil
Correlation plot and summary generated.
package 'corrplot' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\nikhil\AppData\Local\Temp\RtmpIFXWvx\downloaded_packages
Correlation x using Pearson
pdays day duration previous campaign balance age
Rattle 2017-May-05 01:12:16 nikhil
01:12
05-05-2017


R Data Miner - [Rattle (x)]
Project Tools Settings Help
Rattle Version 4.1.0 togaware.com
Execute New Open Save Report Export Stop Quit Connect R
Data Explore Test Transform Cluster Associate Model Evaluate Log
Type: Tree Forest Boost SVM Linear Neural Net Survival All
Numeric Generalized Poisson Logistic Probit Multinomial
Model Builder: glm (logit)
Plot
Summary of the Logistic Regression model (built using glm):
Call:
glm(formula = y ~ ., family = binomial(link = "logit"), data = crs$dataset[crs$train,
c(crs$input, crs$target)])
Deviance Residuals:
Min 1Q Median 3Q Max
-5.8846 -0.3719 -0.2487 -0.1486 3.2223
Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.482498602 0.220859363 -11.240 < 2e-16 ***
age 0.000381387 0.002654185 0.144 0.885743
jobblue-collar -0.362313232 0.088116247 -4.112 3.93e-05 ***
jobentrepreneur -0.441420763 0.151578128 -2.912 0.003589 **
jobhousemaid -0.501975249 0.162348648 -3.092 0.001988 **
jobmanagement -0.180837358 0.088621210 -2.041 0.041294 *
jobretired 0.152474104 0.117924320 1.293 0.196017
jobself-employed -0.277344477 0.133825410 -2.072 0.038225 *
jobservices -0.243075931 0.101334247 -2.399 0.016451 *
jobstudent 0.422209094 0.131637384 3.207 0.001340 **
jobtechnician -0.186196126 0.083023122 -2.243 0.024916 *
jobunemployed -0.213051099 0.136736180 -1.558 0.119205
jobunknown -0.347541874 0.265656972 -1.308 0.190793
maritalmarried -0.172542420 0.071020733 -2.429 0.015121 *
Linear model evaluation has been plotted.
package 'corrplot' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\nikhil\AppData\Local\Temp\RtmpIFXWvx\downloaded_packages
Linear Model x
Residuals vs Fitted
Residuals
Predicted values
Linear Model x
Normal Q-Q
Std. deviance resid.
Theoretical Quantiles
Linear Model x
Scale-Location
√|Std. deviance resid.|
Predicted values
Linear Model x
Residuals vs Leverage
Std. Pearson resid.
Cook's distance
Leverage
01:19
05-05-2017

2.RMINER

This package facilitates the use of data mining algorithms in classification and regression (including time series forecasting) tasks by presenting a short and coherent set of functions. It is a very powerful package and simplifies the data mining task by many bounds.

For e.g. To Fit a SVM or NN model on a data set the data needs to scale to mean of 0 and standard deviation of 1. In many R packages this must be done manually by writing relevant set of codes. But Rminer does this for us inherently, and is done specifying a parameter to choose the scaling.

Set Of Functions provided by Rminer :-

CasesSeries-Create a training set (data. Frame) from a time series using a sliding window.

Crossvaldata-Computes k-fold cross validation for rminer models.

Delevels-Reduce (delete) or replace levels from a factor variable (useful for preprocessing datasets).

Fit-Fit a supervised data mining model (classification or regression) model

Holdout-Computes indexes for holdout data split into training and test sets.

Importance- Measure input importance (including sensitivity analysis) given a supervised data mining model.

Imputation-Missing data imputation (e.g. substitution by value or hotdeck method).

Lforecast-Compute long term forecasts.

Mgraph-Mining graph function

Mining-Powerful function that trains and tests a particular fit model under several runs and a given validation method

Mmetric-Compute classification or regression error metrics.

Mparheuristic-Function that returns a list of searching (hyper)parameters for a particular classification or regression model

predict.fit-predict method for fit objects (rminer)

rminer-Internal-Internal rminer Functions

sa_fri1-Synthetic regression and classification datasets for measuring input importance of supervised learning models
savemining- Load/save into a file the result of a fit (model) or mining functions.
sin1reg-sin1 regression dataset
vecplot-VEC plot function (to use in conjunction with Importance function).

We would be using the Functions Highlighted in Red.

DATA MINING METHODS

1.LOGISTIC REGRESSION.

Logistic regression is a method for fitting a regression curve, y = f(x), when y is a categorical variable. The typical use of this model is predicting y given a set of predictors x. The predictors can be continuous, categorical or a mix of both.
Unlike actual regression, logistic regression does not try to predict the value of a numeric variable given a set of inputs. Instead, the output is a probability that the given input point belongs to a certain class.
Logistic Regression Model seeks to:

- model the probability of an event occurring depending on the values of the independent variables, which can be categorical or numerical.
- estimate the probability that an event occurs for a randomly selected observation versus the probability that an event does not occur.
- predict the effect of a series of variables on a binary response variables.
- classify observations by estimating the probability that an observation in a particular category.

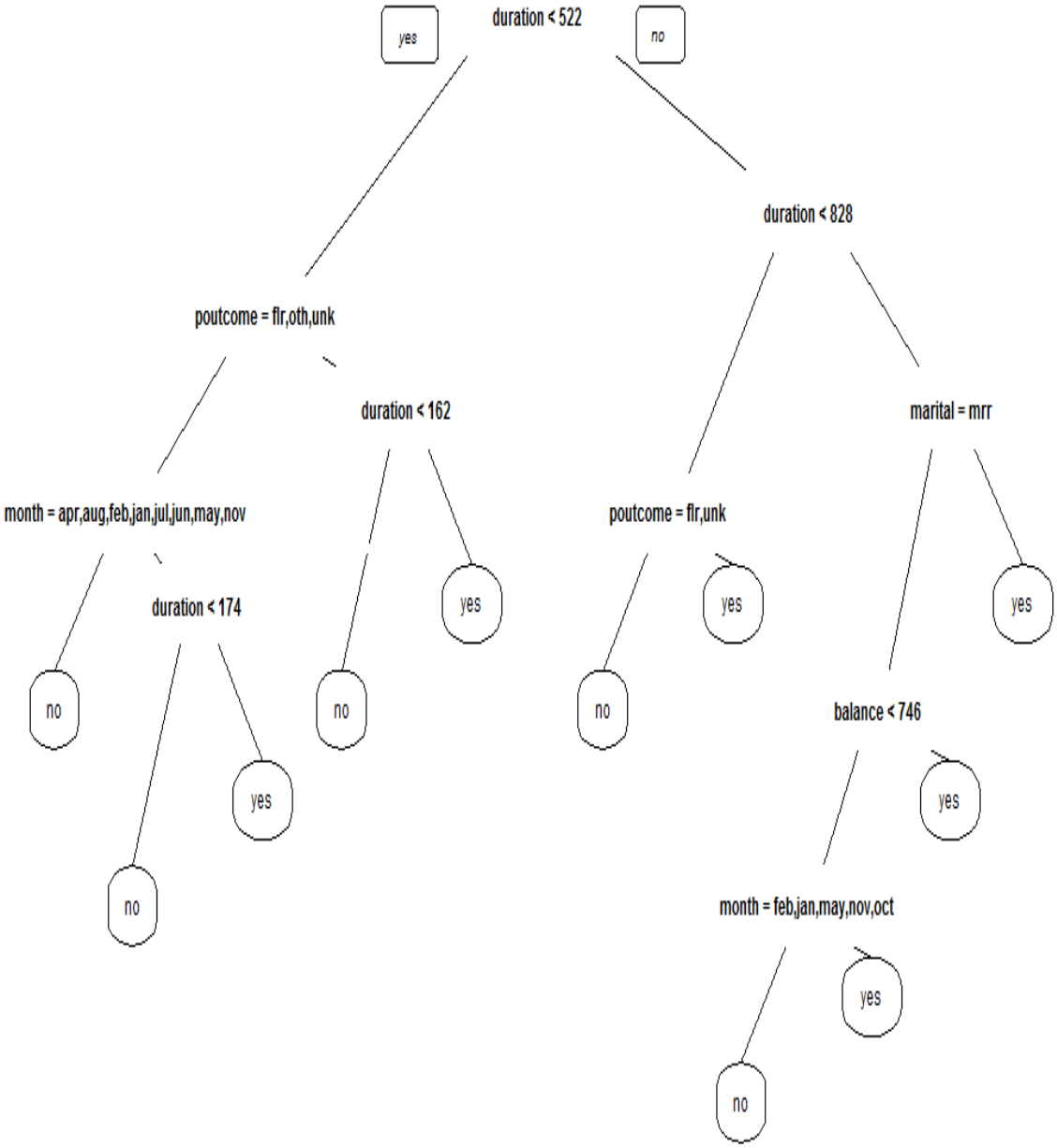
While using our data, we divide the data in training set and testing set. Then we model our Logistic regression model using the training data set. This model considers all the independent variables which are responsible for the dependent variables. Then we predict the dependent variables and check its accuracy with the testing data. The better the prediction the more the accuracy we get from modelling the Logistic Regression.
The central premise of Logistic Regression is the assumption that your input space can be separated into two nice 'regions', one for each class, by a linear boundary. By Linear boundary we mean that, for two dimensions, it's a straight line and no curving. For three

dimensions, it’s a plane. And so on. But for this to make sense, the data points must be separable into the two aforementioned regions by a linear boundary. If your data points do satisfy this constraint, they are said to be linear-separable. Then the model depending on the independent variables, decided the placement of the dependent variable and predict the data.

2. DECISION TRESS.

Decision tree algorithm partitions the data samples into two or more subsets so that the samples within each subset are more homogeneous than in the previous subset. Decision trees are of two main types, one is Classification tree, and the other is Regression tree. For our dataset, we used Classification tree. In the classification setting, the classification error rate is used as a criterion for making the binary splits.

DECISION-TREE FOR THIS DATA.



3. SVM (SUPPORT VECTOR MACHINE).

Support Vector Machines are very specific class of algorithms, characterized by usage of kernels, absence of local minima, sparseness of the solution or on number of support vectors, etc.

Support Vector Machine can be applied not only to classification problems but also to the case of regression. Still it contains all the

main features that characterize maximum margin algorithm: a non-linear function is leaned by linear learning machine mapping into high dimensional kernel induced feature space. The capacity of the system is controlled by parameters that do not depend on the dimensionality of feature space.

In SVM regression, the input $\mathbf{x}$ is first mapped onto a *m*-dimensional feature space using some fixed (nonlinear) mapping, and then a linear model is constructed in this feature space. Using mathematical notation, the linear model (in the feature space) $f(\mathbf{x}, \omega)$ is given by

$$f(\mathbf{x}, \omega) = \sum_{j=1}^{m} \omega_j g_j(\mathbf{x}) + b$$

where $g_j(\mathbf{x}), j = 1, \ldots, m$ denotes a set of nonlinear transformations, and *b* is the "bias" term. Often the data are assumed to be zero mean (this can be achieved by preprocessing), so the bias term is dropped.

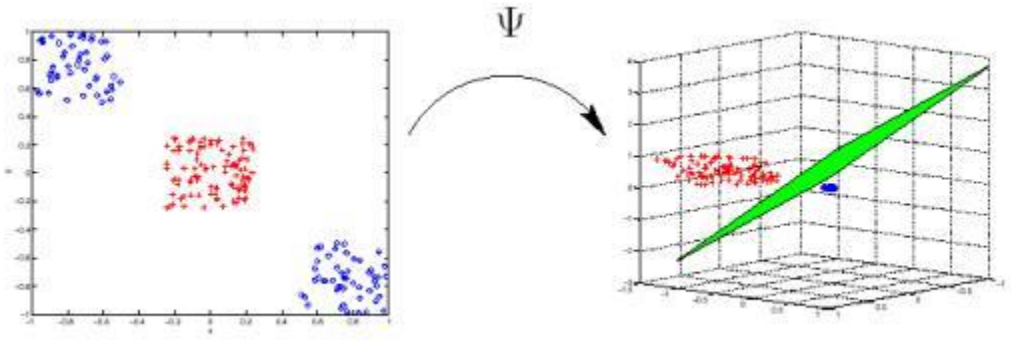


Figure. Non-linear mapping of input examples into high dimensional feature space.

The quality of estimation is measured by the loss function $L(y, f(\mathbf{x}, \omega))$. SVM regression uses a new type of loss function called ε-insensitive loss function proposed by Vapnik :

$$L_\varepsilon(y, f(\mathbf{x}, \omega)) = \begin{cases} 0 & \text{if } |y - f(\mathbf{x}, \omega)| \le \varepsilon \\ |y - f(\mathbf{x}, \omega)| - \varepsilon & \text{otherwise} \end{cases}$$

4. ARTIFICIAL NEURAL NETWORK.

Artificial neural networks (ANNs) or connectionist systems are a computational model used in machine learning, computer science and other research disciplines, which is based on a large collection of connected simple units called artificial neurons, loosely analogous to axons in a biological brain. Connections between neurons carry an activation signal of varying strength.

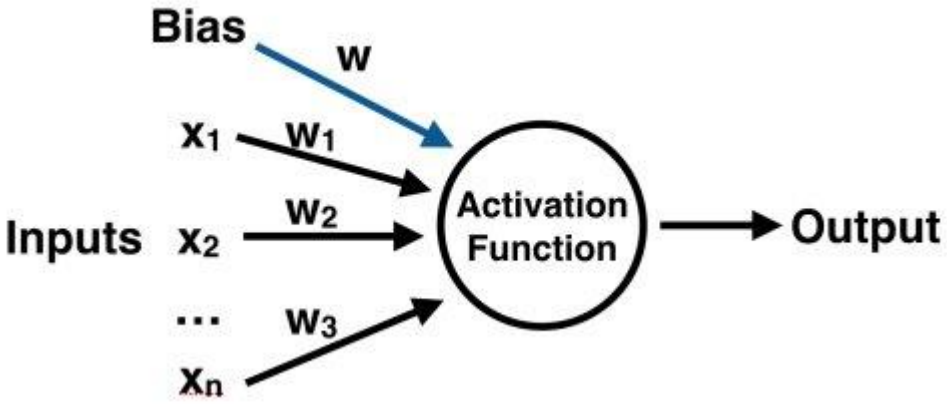
Such systems can be trained from examples, rather than explicitly programmed, and excel in areas where the solution or feature detection is difficult to express in a traditional computer program.

The signals and state of artificial neurons are real numbers, typically between 0 and 1. There may be a threshold function or limiting function on each connection and on the unit itself, such that the signal must surpass the limit before propagating. Back propagation is the use of forward stimulation to modify connection weights, and is sometimes done to train the network using known correct outputs. However, the success is unpredictable: after training, some systems are good at solving problems while others are not. Training typically requires several thousand cycles of interaction.

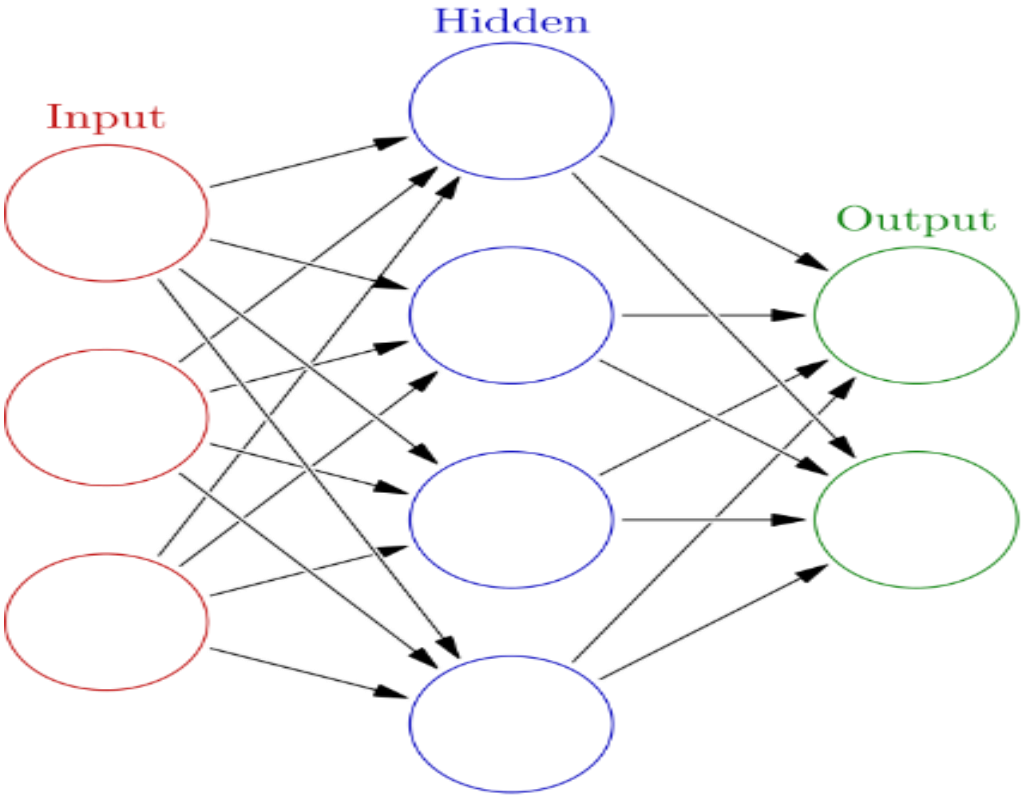
Neural Networks are a machine learning framework that attempts to mimic the learning pattern of natural biological neural networks. Biological neural networks have interconnected neurons with dendrites that receive inputs, then based on these inputs they produce an output signal through an axon to another neuron. We will try to mimic this process through the use of Artificial Neural Networks (ANN), which we will just refer to as neural networks from now on. The process of creating a neural network begins with the most basic form, a single perceptron.

To create a neural network, we simply begin to add layers of perceptron’s together, creating a multi-layer perceptron model of a

neural network. You'll have an input layer which directly takes in your feature inputs and an output layer which will create the resulting outputs. Any layers in between are known as hidden layers because they don't directly "see" the feature inputs or outputs.



A PERCEPTRON MODEL (ABOVE)
A NEURAL NETWORK WITH MULTIPLE PERCEPTRONS (BELOW)



APPLICATION OF DATA MINING METHODS TO THE DATA

SCREESNSHOTS OF THE CODE

```
getwd()
rm(list=ls())
setwd("F:/Study/Financial data Science/project")


x<-read.csv("bank-full.csv",sep = ";")
x1<-read.csv("bank.csv",sep=";")

library(rattle)
library(rminer)

fit_1<-fit(y~.,data = x,model = "mlp",task="p",scale="inputs")
fit_1_1<-fit(y~.,data = x,model = "mlp",task="c",scale="inputs")
f_NN<-predict(fit_1,newdata=x1)
f_NN_1<-predict(fit_1_1,newdata=x1)
CF_NN<-data.frame(x1$y,nn=f_NN)
CF_NN_1<-data.frame(x1$y,nn=f_NN_1)
table(real=CF_NN_1[,1],NN=CF_NN_1[,2])
savemodel(fit_1,"NN")


fit_2<-fit(y~.,data = x,model = "svm",task="p",scale="inputs")
fit_2_2<-fit(y~.,data = x,model = "svm",task="c",scale="inputs")
f_SVM<-predict(fit_2,newdata=x1)
f_SVM_2<-predict(fit_2_2,newdata=x1)
CF_SVM<-data.frame(x1$y,svm=f_SVM)
CF_SVM_2<-data.frame(x1$y,svm=f_SVM_2)
table(real=CF_SVM_2[,1],SVM=CF_SVM_2[,2])
savemodel(fit_2,"SVM")


fit_3<-fit(y~.,data = x,model = "lr",task="p")
fit_3_3<-fit(y~.,data = x,model = "lr",task="c")
f_LR<-predict(fit_3,newdata=x1)
f_LR_3<-predict(fit_3_3,newdata=x1)
CF_LR<-data.frame(x1$y,lr=f_LR)
CF_LR_3<-data.frame(x1$y,lr=f_LR_3)
table(real=CF_LR_3[,1],LR=CF_LR_3[,2])
savemodel(fit_3,"LR")


fit_4<-fit(y~.,data = x,model = "dt",task="p")
fit_4_4<-fit(y~.,data = x,model = "dt",task="c")
f_DT<-predict(fit_4,newdata=x1)
f_DT_4<-predict(fit_4_4,newdata=x1)
CF_DT<-data.frame(x1$y,dt=f_DT)
CF_DT_4<-data.frame(x1$y,dt=f_DT_4)
savemodel(fit_4,"DT")
table(real=CF_DT_4[,1],DT=CF_DT_4[,2])



#------ROC---------------
ROC_NN<-mmetric(CF_NN[,1],CF_NN[,c(2,3)],"ROC")
AUC_NN<-ROC_NN$roc$auc
ROC_NN<-ROC_NN$roc$roc
```

46:32 (Top Level)

```r
#------ROC---------------
ROC_NN<-mmetric(CF_NN[,1],CF_NN[,c(2,3),],"ROC")
AUC_NN<-ROC_NN$roc$auc
ROC_NN<-ROC_NN$roc$roc
plot(ROC_NN[,1],ROC_NN[,2],type="l",col="green",xlab="FPR",ylab="TPR")
ROC_SVM<-mmetric(CF_SVM[,1],CF_SVM[,c(2,3),],"ROC")
AUC_SVM<-ROC_SVM$roc$auc
ROC_SVM<-ROC_SVM$roc$roc
lines(ROC_SVM[,1],ROC_SVM[,2],type="l",col="red")
ROC_LR<-mmetric(CF_LR[,1],CF_LR[,c(2,3),],"ROC")
AUC_LR<-ROC_LR$roc$auc
ROC_LR<-ROC_LR$roc$roc
lines(ROC_LR[,1],ROC_LR[,2],type="l",col="blue")
ROC_DT<-mmetric(CF_DT[,1],CF_DT[,c(2,3),],"ROC")
AUC_DT<-ROC_DT$roc$auc
ROC_DT<-ROC_DT$roc$roc
lines(ROC_DT[,1],ROC_DT[,2],type="l",col="orange")
abline(0,1)
legend(0.6,0.4,c("NN","SVM","LR","DT"),fill =  c("green","red","blue","orange"))
#--------LIFT----------------
LIFT_NN<-mmetric(CF_NN[,1],CF_NN[,c(2,3),],"LIFT")
AUC_LIFT_NN<-LIFT_NN$lift$area
LIFT_NN<-LIFT_NN$lift$alift
plot(LIFT_NN[,1],LIFT_NN[,2],type="l",col="green",xlab="SAMPLE SIZE",ylab="RESPONSES")
LIFT_SVM<-mmetric(CF_SVM[,1],CF_SVM[,c(2,3),],"LIFT")
AUC_LIFT_SVM<-LIFT_SVM$lift$area
LIFT_SVM<-LIFT_SVM$lift$alift
lines(LIFT_SVM[,1],LIFT_SVM[,2],type="l",col="red")
LIFT_LR<-mmetric(CF_LR[,1],CF_LR[,c(2,3),],"LIFT")
AUC_LIFT_LR<-LIFT_LR$lift$area
LIFT_LR<-LIFT_LR$lift$alift
lines(LIFT_LR[,1],LIFT_LR[,2],type="l",col="blue")
LIFT_DT<-mmetric(CF_DT[,1],CF_DT[,c(2,3),],"LIFT")
AUC_LIFT_DT<-LIFT_DT$lift$area
LIFT_DT<-LIFT_DT$lift$alift
lines(LIFT_DT[,1],LIFT_DT[,2],type="l",col="orange")
abline(0,1)
legend(0.6,0.4,c("NN","SVM","LR","DT"),fill =  c("green","red","blue","orange"))

#----------IMPORTANCE ANALYSIS-------------------
IMP_NN<-Importance(fit_1,data = x,method = "SA")
IMP_NN_DSA<-Importance(fit_1,data = x,method = "DSA")
IMP_DT<-Importance(fit_4,data = x,method = "SA")
IMP_SVM<-Importance(fit_2,data = x,method = "SA")
IMP_LR<-Importance(fit_3,data = x,method = "SA")
vecplot(IMP_NN_DSA,sort = "decreasing",graph = "VEC",xlab = "Value range of attributes",ylab = "pr
sum(IMP_DT$value)
sum(IMP_SVM$value)
sum(IMP_LR$value)
IMP_NN_DSA$data
sum(IMP_NN$value)
IMP_NN$data
imp_table<-cbind.data.frame(features=colnames(x),relative_importance=IMP_NN$imp,SA=IMP_NN$value)
write.csv(imp_table,"relativeimportanceandSA.csv")
```

PREDICTION ANALYSIS AND OBSERVATIONS

CONFUSION TABLES

NN CONFUSION TABLE

	NN	
REAL	NO	YES
NO	3856	144
YES	230	291

ACCURACY: 91.72%

SVM CONFUSION TABLE

	SVM	
REAL	NO	YES
NO	3936	64
YES	319	202

ACCURACY: 91.71%

DT CONFUSION TABLE

	DT	
REAL	NO	YES
NO	3896	104
YES	348	173

ACCURACY: 90%

LR CONFUSION TABLE

	LR	
REAL	NO	YES
NO	3912	88
YES	352	169

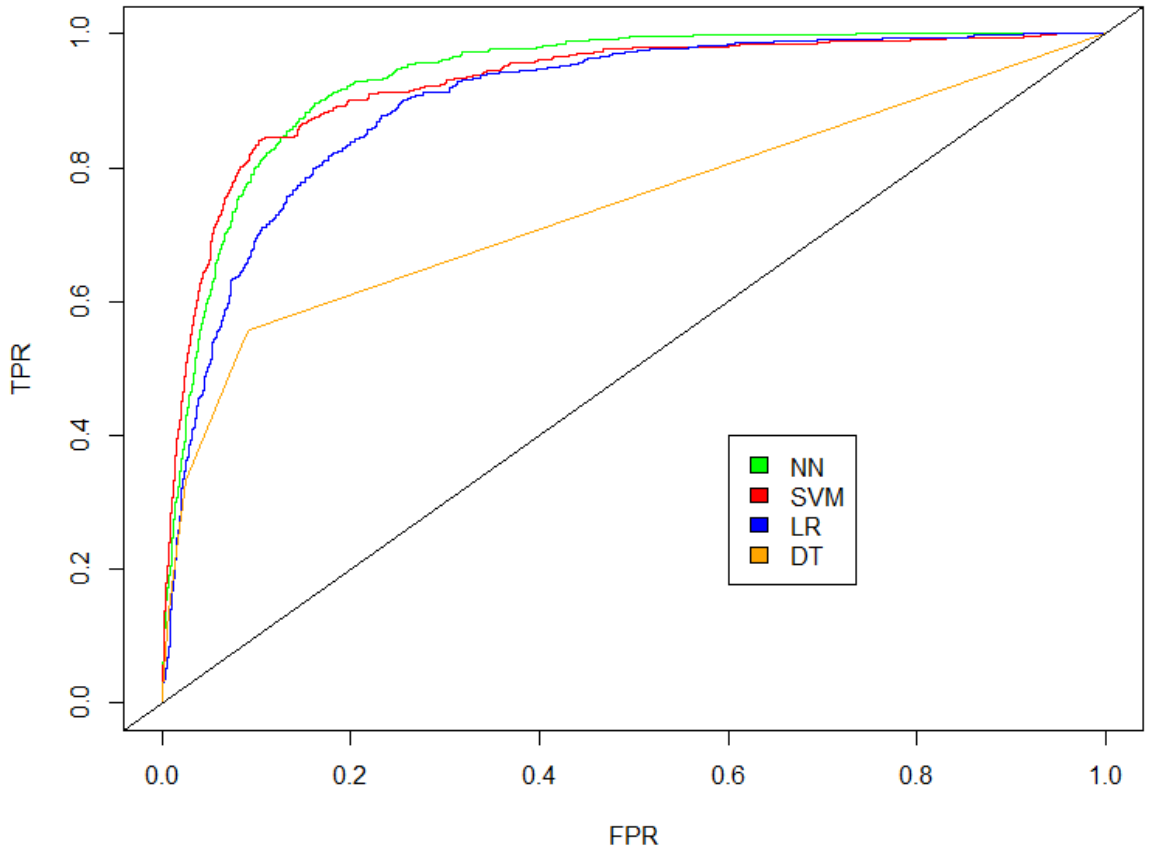
ACCURACY: 90%

As we could see that the accuracy rates are shown by the confusion tables are quite good for all the methods, which creates ambiguity. We know that DT and LR models are very rigid for classifying such uncorrelated and nonlinear data. We are going to use some advanced metrics to highlight this fact.

ROC AND LIFT CURVE ANALYSIS

1. ROC (Receiver Operating characteristics) CURVE

   The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm and can be calculated as (1 − specificity). The ROC curve is thus the sensitivity as a function of fall-out.



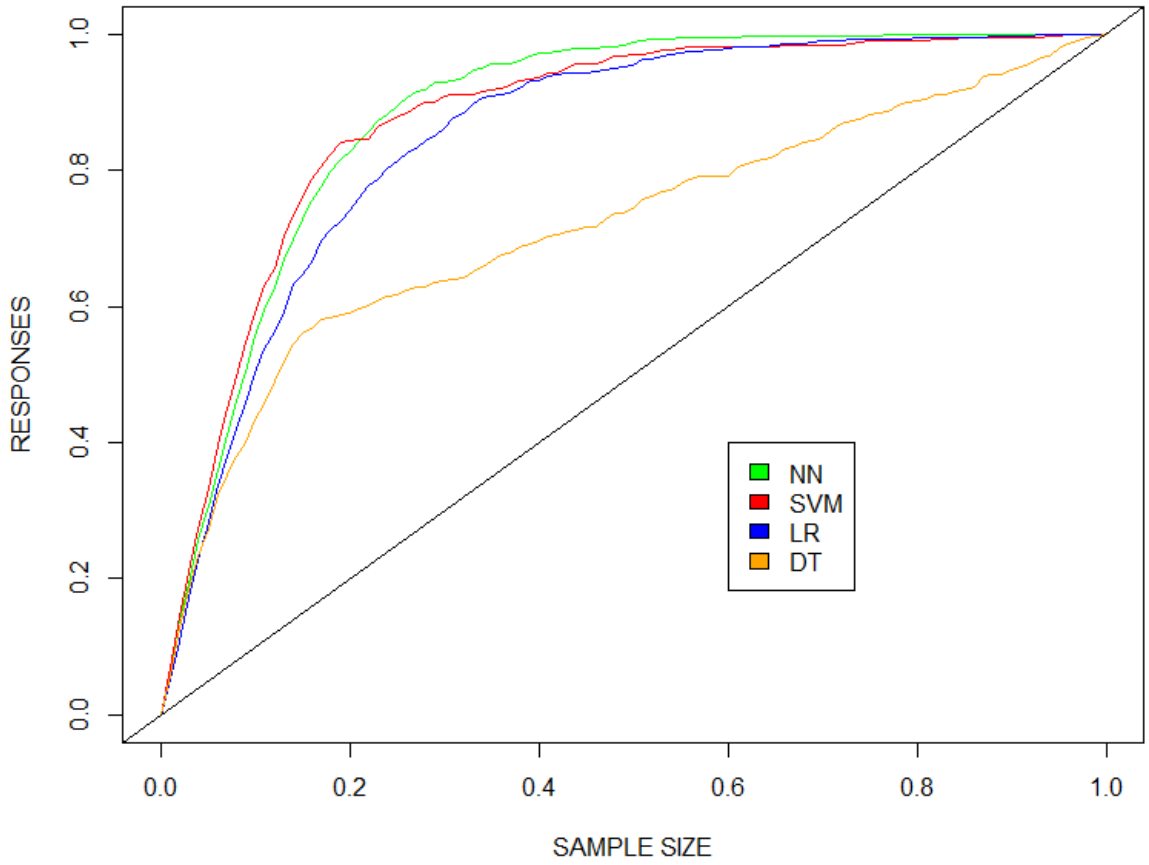
AREA UNDER THE CURVE FOR ALL MODELS

MODEL	NN	SVM	LR	DT
AUC	**0.9325019**	0.9250221	0.8969952	0.741935

2. LIFT CURVE

In data mining and association rule learning, lift is a measure of the performance of a targeting model (association rule) at predicting or classifying cases as having an enhanced response (with respect to the population as a whole), measured against a random choice targeting model. A targeting model is doing a good job if the response within the target is much better than

the average for the population as a whole. Lift is simply the ratio of these values: target response divided by average response.



AREA UNDER THE CURVE FOR ALL MODELS

MODEL	NN	SVM	LR	DT
AUC	**0.882524**	0.8759981	0.8510077	0.7177063

Looking the ROC AND LIFT metrics we could conclude that the NN model is the best classification method for this model. But to cement this fact from the input point of view we conduct sensitivity analysis.

3. SENSITIVITY AND IMPORTANCE ANALYSIS

1. RELATIVE IMPORTANCE TABLE

FEAUTURES	RELATIVE_IMPORTANCE
AGE	0.019478
JOB	0.012609
MARITAL	0.003256
EDUCATION	0.008671
DEFAULT	0.00224
BALANCE	0.004295
HOUSING	0.088887
LOAN	7.01E-05
CONTACT	0.002457
DAY	0.053672
MONTH	0.160348
DURATION	0.130027
CAMPAIGN	0.006945
PDAYS	0.005068
PREVIOUS	0.409708
POUTCOME	0.092268

Relative importance as the name suggest gives relative attribute weights for the model. And this means it creates forcefull attribute co-relations, which says that the model creates a perfect fit for the attributes as the sum is 1, but which never is the case. That is why we study non-relative Sensitivity analysis.

2. RELATIVE IMPORTANCE TABLE WITH SENSITIVITY ANALYSIS

FEATURES	RELATIVE IMPOTANCE	SENSITIVITY ANALYSIS
AGE	0.019478	0.01614
JOB	0.012609	0.010448
MARITAL	0.003256	0.002698
EDUCATION	0.008671	0.007185
DEFAULT	0.00224	0.001856
BALANCE	0.004295	0.003559
HOUSING	0.088887	0.073654
LOAN	7.01E-05	5.81E-05
CONTACT	0.002457	0.002036
DAY	0.053672	0.044474
MONTH	0.160348	0.132868
DURATION	0.130027	0.107743
CAMPAIGN	0.006945	0.005755
PDAYS	0.005068	0.004199
PREVIOUS	0.409708	0.339494
POUTCOME	0.092268	0.076455
SUM	1	0.8286241

SVM SA SUM- 0.8082777

LR SA SUM- 0.7590074

DT SA SUM- 0.2792326

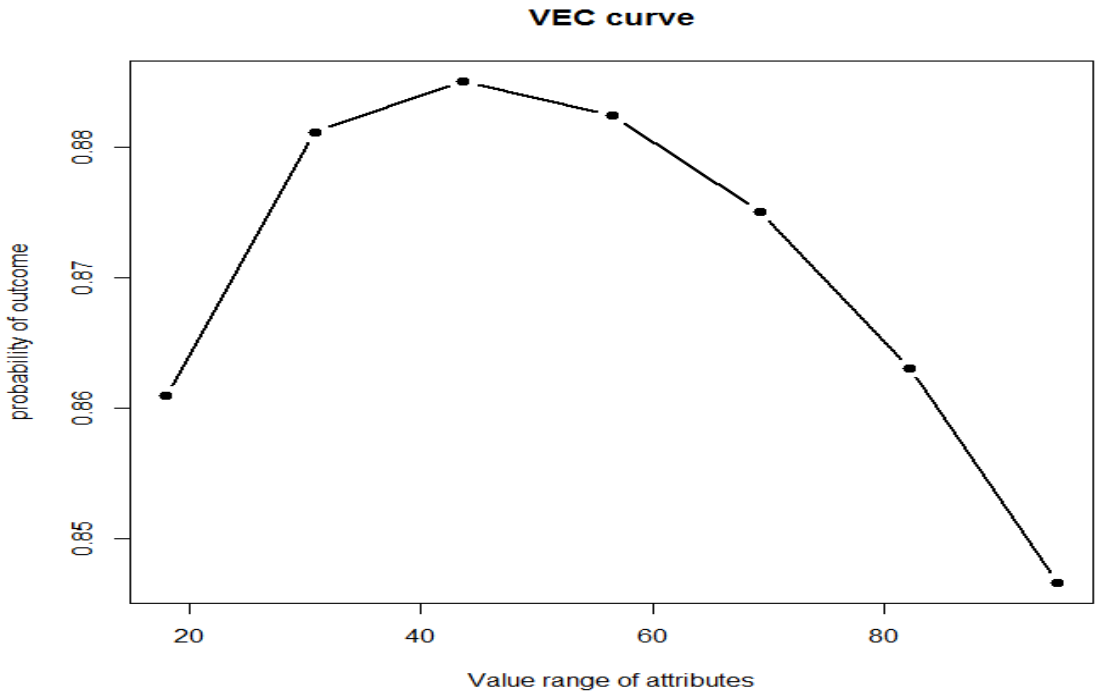
As we could see the SA sum is the also the highest for NN model i.e. if we keep all the attributes zero ,even then there is a chance that we will get an outcome of this model. The lower this chance the better the model fits the data.

And from the remaining observations DT is most surely not a good fit for our data because of the very low sensitivity sum.

Hence looking at the two tables above which reflect the feature sensitivity of the NN model, we could conclude which factors are to be considered while making the calls. For, example previously contacted customers who said yes are highly likely to give a positive response to this telemarketing drive and consequently secure the bank deposit with a positive outcome.

4. VEC-VARIABLE EFFECTIVE CURVE

The VEC curve shown above shows the probability of outcome for the top 6 features



CONCLUSIONS

The obtained results are credible for the banking domain and provide valuable knowledge for the telemarketing campaign manager. In our observations we concluded a good classification model and applied sensitivity analysis to get the features or attributes which effect the outcome the most.

The result of this is that the company could give valuable time and importance to a customer base which is largely going to affect their business. Such analysis could lead to huge savings in time and money for the relevant institution. Which is why institutions have started giving increasing importance to the power of data analytics, which is the stepping stone to AI.

REFERENCES

https://www.wikipedia.org/

https://cran.r-project.org/web/packages/rminer/rminer.pdf

https://cran.r-project.org/web/packages/rattle/rattle.pdf

[Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014

http://www.kdnuggets.com/2016/08/begineers-guide-neural-networks-r.html