Contents

ABSA BANK MARKET

ABSA bank is planning to make decisions on their marketing tactics and planning promotional activities for increasing the sales. For that, as an analyst, we are provided with a dataset of 30,000 ABSA customers who were contacted for one of the investment products mentioned and then we have to analyse the potential 100 customers who will be interested in purchasing any of the investment product.

According to the data provided, we have these as the attributes in the dataset:

Data available:

1.Gender: M=male; F=female

2.Age: an integer parameter

3.marital_status: widowed; married; single; divorced

4.education: basic; highsch; univ; postgrad

5.nb_depend_child: number of dependent children (0,1,2,3)

6.employ_status: employment status (full_time; part_time; unemployed; self_employ; retired)

7.yrs_current_job: years at the current employment

8.yrs_employed: total number of years employed so far

9.net_income: an integer parameter

10.spouse_work: yes; no

11.spouse_income: if the spouse works, what is his/her income?

12.residential_status: home owner (owner); tenant; home owner with a mortgage (owner_morg); living with parents (w_parents)

13.yrs_current_address: years at the current address

14. product: bonds; stocks; derivatives

15. purchase: yes; no

Questions:

1. An investigation of the data and a summary of your descriptive analyses;

2.A discussion on the pros and cons of the prediction methods that can be used to address AB SA's marketing problem;

3.A brief description (and the assumptions made, if any) of how selected prediction methods are applied;

4. R codes developed;

5. An evaluation of the results obtained by each prediction method tried on the data;

6. A comparative analysis of the results;

7.
Your final recommendation to ABSA on which customers the sales force should choose to tar get;

8.A discussion on any additional data that you think would be useful, if collected, to make bet ter predictions in the future.

Pre-requisites:

So, to solve all the problems, we have to look upon some pre-requisites that are needed to be fulfilled so that our analysis becomes smoother.

These are:

1. Knowledge of basic mathematics and concepts behind all the algorithms and models that are going to be applied throughout the analysis.
2. Knowledge of basic data manipulation techniques like changing the type of data for the required analysis.
3. Knowledge of all the packages that are being applied in the analysis and the sub attributes related to it
4. Basic Coding on R
5. Types of plots and other visualization techniques, that can be used to show out the results.
6. Idea about different kinds of shortcuts and techniques that can be used to shorten the coding in R

Data Manipulation:

This is done before solving any part of the analysis. It is required for making the data compatible for working with any model that we are going to develop and use it.

So, in the below pic, it is the structure format of our file initially.

```
> str(data)
'data.frame':	30000 obs. of  15 variables:
 $ gender             : chr  "M" "M" "F" "M" ...
 $ age                : int  29 21 21 30 30 38 43 21 20 38 ...
 $ marital_status     : chr  "married" "single" "married" "divorced" ...
 $ education          : chr  "basic" "basic" "highsch" "basic" ...
 $ nb_depend_child    : int  1 0 0 0 1 0 2 1 0 0 ...
 $ employ_status      : chr  "part_time" "self_employ" "part_time" "full_time" ...
 $ yrs_current_job    : int  1 2 1 1 3 4 8 0 0 2 ...
 $ yrs_employed       : int  5 2 2 6 8 11 18 1 1 7 ...
 $ net_income         : int  12750 25500 19125 31875 127500 47813 127500 25500 19125 31875 ...
 $ spouse_work        : chr  "yes" "no" "yes" "no" ...
 $ spouse_income      : int  37640 0 48968 0 0 54984 0 47028 0 0 ...
 $ residential_status : chr  "owner" "w_parents" "tenant" "tenant" ...
 $ yrs_current_address: int  9 5 5 12 1 4 12 6 6 10 ...
 $ product            : chr  "derivatives" "stocks" "bonds" "bonds" ...
 $ purchase           : chr  "yes" "yes" "yes" "no" ...
>
```

Here, we can see that there are columns where there is a dataset of 30,000 observations where we have 15 variables in which some columns are character (defined as "chr") and some integer columns we have integers (defined as "int"). So, when we will try to work with this dataset, there will be lot of complexities when fitting it on the models.

To be able to solve the problem, we can convert the data into another type, be it numeric or into factors, that are easy to handle and gives better results.

To do the following, we can do two different kinds of data manipulation and here, we are going to use both of the manipulation as there are different models that works on only numeric data and there are some which gives better solution with factors.

1. Using **dplyr** library and *mutate* function:

```
> dat1 <- mutate(data,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
+                spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
+                employ_status = factor(employ_status), purchase = factor(purchase))
> str(dat1)
'data.frame':	30000 obs. of  15 variables:
 $ gender             : Factor w/ 2 levels "F","M": 2 2 1 2 2 1 1 2 2 2 ...
 $ age                : int  29 21 21 30 30 38 43 21 20 38 ...
 $ marital_status     : Factor w/ 4 levels "divorced","married",..: 2 3 2 1 4 2 1 2 3 3 ...
 $ education          : Factor w/ 4 levels "basic","highsch",..: 1 1 2 1 3 2 3 1 2 1 ...
 $ nb_depend_child    : int  1 0 0 0 1 0 2 1 0 0 ...
 $ employ_status      : Factor w/ 5 levels "full_time","part_time",..: 2 4 2 1 1 1 1 4 2 1 ...
 $ yrs_current_job    : int  1 2 1 1 3 4 8 0 0 2 ...
 $ yrs_employed       : int  5 2 2 6 8 11 18 1 1 7 ...
 $ net_income         : int  12750 25500 19125 31875 127500 47813 127500 25500 19125 31875 ...
 $ spouse_work        : Factor w/ 2 levels "no","yes": 2 1 2 1 1 2 1 2 1 1 ...
 $ spouse_income      : int  37640 0 48968 0 0 54984 0 47028 0 0 ...
 $ residential_status : Factor w/ 4 levels "owner","owner_morg",..: 1 4 3 3 3 2 1 2 4 3 ...
 $ yrs_current_address: int  9 5 5 12 1 4 12 6 6 10 ...
 $ product            : Factor w/ 3 levels "bonds","derivatives",..: 2 3 1 1 1 3 1 2 2 3 ...
 $ purchase           : Factor w/ 2 levels "no","yes": 2 2 2 1 2 1 2 2 2 1 ...
>
```

In this, we can see that I have changed the dataset type. So, for changing, I have mutated all the columns with character as variable to factor, which is being used for mostly all the models that we will be going to use further.

2. Using **fastDummies** library and *dummy_cols* function:

```
> dum_targ <- dummy_cols(dat1,select_columns=c("education","gender","marital_status","spouse_work",
+                                              "residential_status","product","employ_status","purchase"),
+                         remove_first_dummy= TRUE, remove_selected_columns= TRUE)
> str(dum_targ)
'data.frame':	30000 obs. of  25 variables:
 $ age                        : int  29 21 21 30 30 38 43 21 20 38 ...
 $ nb_depend_child            : int  1 0 0 0 1 0 2 1 0 0 ...
 $ yrs_current_job            : int  1 2 1 1 3 4 8 0 0 2 ...
 $ yrs_employed               : int  5 2 2 6 8 11 18 1 1 7 ...
 $ net_income                 : int  12750 25500 19125 31875 127500 47813 127500 25500 19125 31875 ...
 $ spouse_income              : int  37640 0 48968 0 0 54984 0 47028 0 0 ...
 $ yrs_current_address        : int  9 5 5 12 1 4 12 6 6 10 ...
 $ education_highsch          : int  0 0 1 0 0 1 0 0 1 0 ...
 $ education_postgrad         : int  0 0 0 0 1 0 1 0 0 0 ...
 $ education_univ             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ gender_M                   : int  1 1 0 1 1 0 0 1 1 1 ...
 $ marital_status_married     : int  1 0 1 0 0 1 0 1 0 0 ...
 $ marital_status_single      : int  0 1 0 0 0 0 0 0 1 1 ...
 $ marital_status_widowed     : int  0 0 0 0 1 0 0 0 0 0 ...
 $ spouse_work_yes            : int  1 0 1 0 0 1 0 1 0 0 ...
 $ residential_status_owner_morg: int  0 0 0 0 0 1 0 1 0 0 ...
 $ residential_status_tenant  : int  0 0 1 1 1 0 0 0 0 1 ...
 $ residential_status_w_parents : int  0 1 0 0 0 0 0 0 1 0 ...
 $ product_derivatives        : int  1 0 0 0 0 0 0 1 1 0 ...
 $ product_stocks             : int  0 1 0 0 0 1 0 0 0 1 ...
 $ employ_status_part_time    : int  1 0 1 0 0 0 0 0 1 0 ...
 $ employ_status_retired      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ employ_status_self_employ  : int  0 1 0 0 0 0 0 1 0 0 ...
 $ employ_status_unemployed   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ purchase_yes               : int  1 1 1 0 1 0 1 1 1 0 ...
> 
```

In this, we are mutating all the character columns into a numeric column where we are making the data in such a way that we are making the choices in terms of 1/0 where 1 means that the particular data is present in that variable column and 0 means absent. If any column doesn't has any 1 i.e., all columns are 0, then we will consider it in the last category, on the basis of which we are making the different choice columns.

So, with these two manipulation techniques, we will be going to implement the subsequent models that will help us to make predictions on the given data.

Libraries Used:

For the whole analysis, we will be going to use certain kinds of libraries that will help us to do all the work in the project as shown in the below snippet:

```
#####################Calling All the Libraries##################################
library('dplyr')
library('FSelector')
library('party')
library('rpart')
library('rpart.plot')
library('mlbench')
library('caret')
library('pROC')
library('tree')
library('C50')
library('e1071')
library('class')
library('caTools')
library('naivebayes')
library('fastDummies')
library('gmodels')
```

These libraries have different kinds of use and different functionalities. Some are being used for data manipulation, some are used for plotting, some are being used for making the analysis models, etc., which will make our work faster and easier.

Solutions:

1. An investigation of the data and a summary of my descriptive analysis:

As per the given data, we have initially a data of 30,000 customers who have already bought or didn't buy the product as per the data. So, to understand it in a better way, we can say that there are some customers who will purchase the particular commodity or not which is totally based on the given 14 values given by the bank and it will help to understand the trend of the customers regarding making purchase on the given commodity.

So first, we will read the data to our R environment using the following codes:

```
data <- read.csv(file.choose())

data_test <- read.csv(file.choose())
```

This statement allows us to select any csv file present in any directory in the machine. So with this, we add the Market.csv and Market_pred.csv to the environment.

So, to understand the data better, we will be doing some pre-processing analysis where we will find the basic structure of the data:

```
> str(data)
'data.frame':	30000 obs. of  15 variables:
 $ gender             : chr  "M" "M" "F" "M" ...
 $ age                : int  29 21 21 30 30 38 43 21 20 38 ...
 $ marital_status     : chr  "married" "single" "married" "divorced" ...
 $ education          : chr  "basic" "basic" "highsch" "basic" ...
 $ nb_depend_child    : int  1 0 0 0 1 0 2 1 0 0 ...
 $ employ_status      : chr  "part_time" "self_employ" "part_time" "full_time" ...
 $ yrs_current_job    : int  1 2 1 1 3 4 8 0 0 2 ...
 $ yrs_employed       : int  5 2 2 6 8 11 18 1 1 7 ...
 $ net_income         : int  12750 25500 19125 31875 127500 47813 127500 25500 19125 31875 ...
 $ spouse_work        : chr  "yes" "no" "yes" "no" ...
 $ spouse_income      : int  37640 0 48968 0 0 54984 0 47028 0 0 ...
 $ residential_status : chr  "owner" "w_parents" "tenant" "tenant" ...
 $ yrs_current_address: int  9 5 5 12 1 4 12 6 6 10 ...
 $ product            : chr  "derivatives" "stocks" "bonds" "bonds" ...
 $ purchase           : chr  "yes" "yes" "yes" "no" ...
> 
```

We can observe here that we have 30,000 observations with 15 columns with 8 columns with character as their values and 7 of them with integer as the data category. If we further investigate in the data after doing data manipulation explained as in pre-requisite section, we get the following:

```
> dat1 <- mutate(data,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
+                spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
+                employ_status = factor(employ_status), purchase = factor(purchase))
> summary(dat1)
 gender         age          marital_status     education      nb_depend_child       employ_status    yrs_current_job  yrs_employed
 F:14987   Min.   :20.00   divorced:8326   basic   : 4954   Min.   :0.0000   full_time  :11643   Min.   : 0.0    Min.   : 0.000
 M:15013   1st Qu.:26.00   married :8072   highsch :13056   1st Qu.:0.0000   part_time  : 5687   1st Qu.: 0.0    1st Qu.: 3.000
           Median :34.00   single  :9901   postgrad: 3250   Median :0.0000   retired    :  435   Median : 2.0    Median : 7.000
           Mean   :37.53   widowed :3701   univ    : 8740   Mean   :0.7886   self_employ: 6034   Mean   : 3.9    Mean   : 9.489
           3rd Qu.:46.00                                    3rd Qu.:1.0000   unemployed : 6201   3rd Qu.: 6.0    3rd Qu.:14.000
           Max.   :88.00                                    Max.   :3.0000                       Max.   :35.0    Max.   :54.000
   net_income      spouse_work spouse_income     residential_status yrs_current_address        product       purchase
 Min.   :      0   no :23297   Min.   :     0   owner      : 6553   Min.   : 1.000      bonds      :10021   no :14609
 1st Qu.: 19125    yes: 6703   1st Qu.:     0   owner_morg: 6354   1st Qu.: 4.000      derivatives:10094   yes:15391
 Median : 38250                Median :     0   tenant     :16514   Median : 7.000      stocks     : 9885
 Mean   : 44808                Mean   :  9915   w_parents  :  579   Mean   : 7.101
 3rd Qu.: 57375                3rd Qu.:     0                       3rd Qu.:10.000
 Max.   :178500                Max.   :174797                       Max.   :15.000
```

Here, we can see that we have:

Variable Name	Type of observation	Values
Gender	Classification	F: 14987 M: 15013
Age	Numeric	Min: 20 Max: 88 Mean: 37.53
Marital_Status	Classification	Divorced: 8326 Married: 8072 Single: 9901 Widowed: 3701
Education	Classification	Basic: 4954 Highschool: 13056 Postgrad: 3250 Univ: 8740
nb_depend_child	Numeric	Min: 0 Max: 3 Mean: 0.7886
employ_status	Classification	Full_time: 11643 part_time: 5687 retired: 435 self_employ: 6034 unemployed: 6201
yrs_current_job	Numeric	Min: 0 Max: 35 Mean: 3.9
yrs_employed	Numeric	Min: 0 Max: 178500 Mean: 44808
spouse_work	Classification	No: 23297 Yes: 6703
spouse_income	Numeric	Min: 0 Max: 174797 Mean: 9915
residential_status	Classification	Owner: 6553 Owner_morg: 6354 Tenant: 16514 W_parents: 579
yrs_current_address	Numeric	Min: 1 Max: 15 Mean: 7.101
product	Classification	Bonds: 10021 Derivatives: 10094 Stocks: 9885
Purchase	Classification	No: 14609 Yes: 15391

So, to understand how we can work with the data, we need to calculate information gain and other kinds of plot that can help us to get a starting point from where we can do data exploration which would help in making our models that can be used for the rest of the work. For that, we will use the following code:

```
info_gain <- information.gain(purchase~., data = dat1)

info_gain %>%

  arrange(desc(attr_importance))
```

So, after executing this, we get the following output:

```
> info_gain <- information.gain(purchase~., data = dat1)
> info_gain %>%
+   arrange(desc(attr_importance))
                          attr_importance
product                      0.0411421923
age                          0.0309859060
nb_depend_child              0.0158975320
yrs_employed                 0.0149834405
net_income                   0.0128183573
marital_status               0.0126433405
yrs_current_job              0.0054506218
yrs_current_address          0.0042054235
spouse_work                  0.0039437468
spouse_income                0.0039371980
gender                       0.0012636246
education                    0.0004595292
employ_status                0.0004501333
residential_status           0.0002307776
> 
```

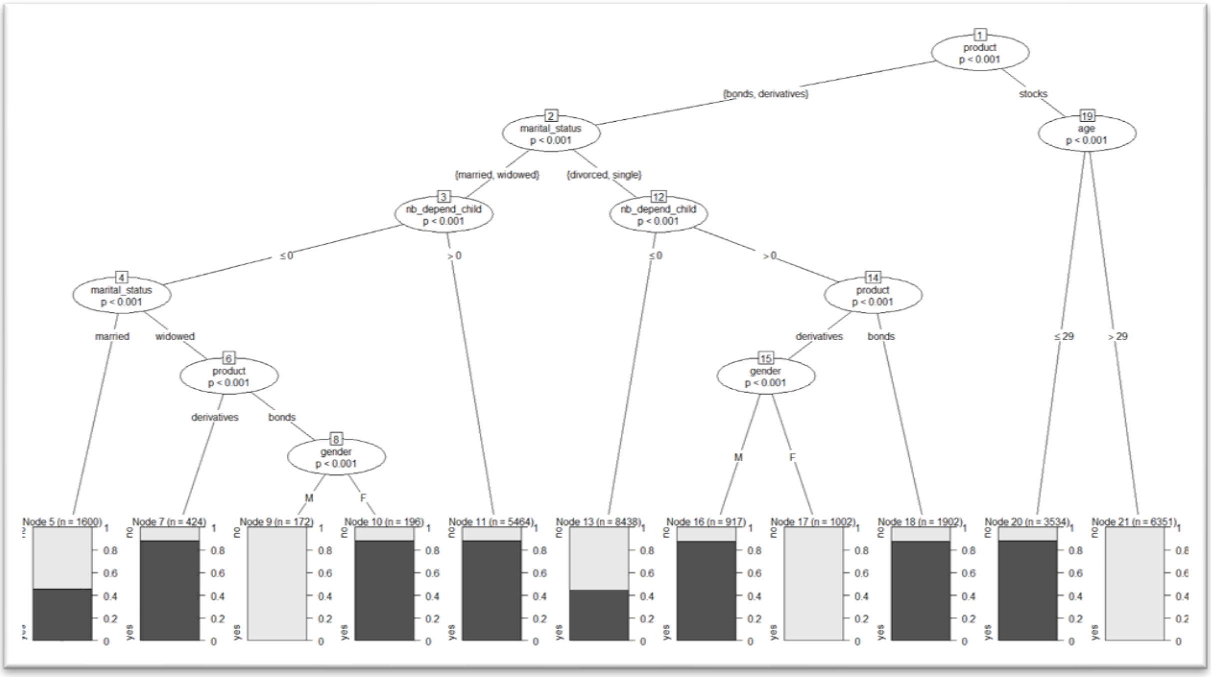
From this, we get that the most useful information from which we can do exploitary analysis is using product at first level. If we go into more depth regarding information gain, and do entropy analysis, we can directly make an ID3 decision tree using this.

ID3 is a type of decision tree where the nodes are calculated using combination of information gain and entropy and whole tree is made. So,using same idea on the given data, we can get a basic idea how the data is being divided based on the sum of entropy and other steps.

So, we used the following codes:

```
reg_tst <- ctree(purchase ~ .,dat1)

summary(reg_tst)

plot(reg_tst)
```

We get the following output:



From this, we can say that with reference to information gain and ID3 logic, it shows how we can use information gain to get a sustainable approach to our dataset and how we could get the best solution.

But we can find more info about the data using different techniques like plots and charts that can give insights regarding the frequency of any variable based on purchase.

2. A discussion on the pros and cons of the prediction methods that can be used to address ABSA's marketing problem

ABSA's marketing problem can be addressed using several methods but, in this case, we will be using Machine Learning models which are normally used for classification. The classification algorithms used for the following are:

a. K-Nearest Neighbour
b. Logistic Regression
c. Decision tree (CART)
d. Decision tree (5.0)
e. SVM
f. Naïve-bayes
g. Decision tree (ID3)

So, there are different types of algorithms all together as the mathematics behind each and every algorithm is different. So, the major advantages and disadvantages of each and every algorithm are:

ALGORITHM	ADVANTAGE	DISADVANTAGE
K-Nearest Neighbour	Very Intuitive and Simple No Assumptions required No extra training steps Easy to Implement	Slow Algorithm Needs Homogenous features Sensitive to Outliers Can't treat missing values
Logistic Regression	Easy to Implement and Interpret Easily extends to multiclass Good accuracy for simple dataset Less inclined to overfit	Can overfit if obs<features Assumptions done on linear relation Can only predict discrete functions Tough to get complex relationships
Decision Tree (CART)	Can Inherit multiclass classification Can handle nonlinear relationships	Vulnerable to small variance in data Can become underfit if class are imbalanced
Decision Tree (5.0)	All Purpose Classifier Excludes unimportant feature Works on small and large dataset Efficient than other complex model	Easy to Overfit/Underfit Biased towards splits on larger lvl. High vulnerability if change in data Can become difficult to interpret
SVM	Good on unstructured data Less Risk in overfitting Works better than ANN in some cases	Finding "optimal" kernel is tough Takes time to train Tuning the model is tough
Naïve-Bayes	Simple and easy to implement Not enough training data required It can handle both continuous and discrete data	Assumes the data to be independent Has chances of 0 frequency problem Estimations can be wrong
Decision Tree (ID3)	Fast and short tree Works well in small and large dataset	Data may get overfit in very small dataset One attribute tested at a time

So, with this, we can say all algorithms have some advantages and some disadvantages.

3. A brief description (and the assumptions made, if any) of how selected prediction met hods are applied

There are some specific reasons that are to be considered before selecting any of the given algorithm. We can see that all the selected algorithms are all considered as classification algorithms and none of the regression algorithms are being used expect linear regression.

The major reason for not using any kind of regression model is that regression model works mainly on the situation where we want continuous numeric results but, in this case, we require solutions that gives us solution in discrete numeric i.e., one single value. For example, we take an output snippet of linear regression and logistic regression, one of them is a regression model and other is classification model.

```
Call:
lm(formula = ROLL ~ UNEM + HGRAD + INC, data = datavar)

Residuals:
     Min        1Q    Median        3Q       Max
-1148.840  -489.712    -1.876   387.400  1425.753

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -9.153e+03  1.053e+03  -8.691 5.02e-09 ***
UNEM         4.501e+02  1.182e+02   3.809 0.000807 ***
HGRAD        4.065e-01  7.602e-02   5.347 1.52e-05 ***
INC          4.275e+00  4.947e-01   8.642 5.59e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 670.4 on 25 degrees of freedom
Multiple R-squared: 0.9621, Adjusted R-squared: 0.9576
F-statistic: 211.5 on 3 and 25 DF,  p-value: < 2.2e-16
```

Figure 1: Output of linear regression (source: towardsdatascience.com)

```
## Call:
## glm(formula = admit ~ gre + gpa + rank, family = "binomial",
##     data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

Figure 2: Output of logistic regression (source: towardsdatascience.com)

As we compare both of the model outputs, we can clearly see that the linear regression is giving us values that are simply residuals, that can be only used when there is any continuous line is present to link the result with the plotted line whereas the logistic regression is giving us deviance residuals, which is a kind of an output which can be used to depict a set of values using a suitable single value, which is discrete and maybe unique.

Also, when we use the dataset for a regression model, we have to use a dataset where we have all the required value in terms of integer, float or any other kind of numeric value whereas in classification algorithm, it is necessary to contain at least one variable column which contains all values as a factor or discrete value i.e., a value which is a whole number or a Boolean value.

So, to apply those techniques, we used R and RStudio for our work, in which we used all the suitable libraries that are required by the algorithms that would make the processing easier and faster.

Also, for applying those techniques, I made some algorithms on papers so that I could understand what I could do to solve the problem efficiently

4. R-Codes deployed

In the following boxes, the code which was used to develop the analysis models and get the required solution:

```
####################Calling All the Libraries##################################
library('dplyr')
library('FSelector')
library('party')
library('rpart')
library('rpart.plot')
library('mlbench')
library('caret')
library('pROC')
library('tree')
library('C50')
library('e1071')
library('class')
library('caTools')
library('naivebayes')
library('fastDummies')
library('gmodels')
###################Loading the CSV's#####################################
data <- read.csv(file.choose())
data_test <- read.csv(file.choose())
str(data)
summary(data)
######################### Modifying the CSV for use ####################################
dat1 <- mutate(data,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status), purchase = factor(purchase))
summary(dat1)
str(dat1)

dat_test <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
summary(dat_test)
str(dat_test)
########################## Finding info gain and other insight from the data ##################################################
info_gain <- information.gain(purchase~., data = dat1)
info_gain %>%
  arrange(desc(attr_importance))

reg_tst <- ctree(purchase ~ .,dat1)
summary(reg_tst)
plot(reg_tst)
```

```
################################ knn algorithm ##################################################################################
dum_targ <- dummy_cols(dat1,select_columns=c("education","gender","marital_status","spouse_work",
                                "residential_status","product","employ_status","purchase"),
              remove_first_dummy= TRUE, remove_selected_columns= TRUE)
str(dum_targ)
k_train <- dum_targ[1:25000,]
k_test <- dum_targ[25001:30000,]
k_train_label <- dum_targ[1:25000,25]
k_test_label <- dum_targ[25001:30000,25]

k_test_pred <- knn(train = k_train, test = k_test,cl = k_train_label, k=10)
CrossTable(x = k_test_label, y = k_test_pred, prop.chisq = F)


##################################### logisticreg ##############################################################################

set.seed(150000)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train2 <- dat1[ind == 1,]
test2 <- dat1[ind == 2,]
logit <- glm(purchase~., data = train2, family = "binomial")
summary(logit)
confint.default(logit)
predict(logit, newdata = test2, type = "response")
predres <- predict(logit, newdata = test2, type = "response")
predres <- round(predres)
predres<-ifelse(predres=="0","no","yes")
predres <- as.factor(predres)
confusionMatrix(predres,test2$purchase)

test3 <- dat_test
test3 <- select(test3, -purchase)
purchase <- predict(logit, newdata = test3, type = "response")
str(purchase)
purchase <- round(purchase)
test3 <- cbind(test3,purchase)
test3$purchase<-ifelse(test3$purchase=="0","No","Yes")
test3$purchase <- as.factor(test3$purchase)
str(test3)
logit_test <- C5.0(purchase ~ .,test3)
summary(logit_test)
plot(logit_test)
write.csv(test3,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_logistic.csv")
```

```
############################################### Decision Tree CART ####################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train <- dat1[ind == 1,]
test <- dat1[ind == 2,]
tree <- rpart(purchase ~., data = train, cp=0.015)
rpart.rules(tree)
rpart.plot(tree)
printcp(tree)
plotcp(tree)
p <- predict(tree, test, type = 'class')
confusionMatrix(p, test$purchase)

p1 <- predict(tree, test, type = 'prob')
p1 <- p1[,2]
r <- multiclass.roc(test$purchase, p1, percent = TRUE)
roc <- r[['rocs']]
r1 <- roc[[1]]
plot.roc(r1,
      print.auc=TRUE,
      auc.polygon=TRUE,
      grid=c(0.1, 0.2),
      grid.col=c("green", "red"),
      max.auc.polygon=TRUE,
      auc.polygon.col="lightblue",
      print.thres=TRUE,
      main= 'ROC Curve')

test1 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
test1$purchase <- as.factor(test1$purchase)
str(test1)
test1 <- select(test1, -purchase)
purchase <- predict(tree, test1, type = 'class')
test1 <- cbind(test1,purchase)
str(test1)
test1$purchase
reg2_tst <- C5.0(purchase ~ .,test1)
summary(reg2_tst)
plot(reg2_tst)
```

```
################################################ Decision Tree C5.0 ########################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train1 <- dat1[ind == 1,]
test1 <- dat1[ind == 2,]
tree1 <- C5.0(purchase ~., data = train1, cp = 0.0025)
plot(tree1)

p5 <- predict(tree1, test1, type = 'class')
confusionMatrix(p5, test1$purchase)

p15 <- predict(tree1, test1, type = 'prob')
p15 <- p15[,2]
r5 <- multiclass.roc(test1$purchase, p15, percent = TRUE)
roc5 <- r5[['rocs']]
r15 <- roc5[[1]]
plot.roc(r15,
     print.auc=TRUE,
     auc.polygon=TRUE,
     grid=c(0.1, 0.2),
     grid.col=c("green", "red"),
     max.auc.polygon=TRUE,
     auc.polygon.col="lightblue",
     print.thres=TRUE,
     main= 'ROC Curve')

test2 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
          spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
          employ_status = factor(employ_status))
test2$purchase <- as.factor(test2$purchase)
str(test2)
test2 <- select(test2, -purchase)
purchase <- predict(tree1, test2, type = 'class')
test2 <- cbind(test2,purchase)
str(test2)
test2$purchase
reg3_tst <- C5.0(purchase ~ .,test2)
summary(reg3_tst)
plot(reg3_tst)

write.csv(test2,"D://UCC STUFF//IS6052//Assignment//Market_pred_result2.csv")
```

```
############################################### SVM ##############################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train2 <- dat1[ind == 1,]
test2 <- dat1[ind == 2,]
svmtest <- svm(purchase ~., data = train2)

summary(svmtest)
print(svmtest)
svmpred <- predict(svmtest,test2, type = 'class')
confusionMatrix(svmpred, test2$purchase)
plot(svmpred,test2$purchase)

psvm <- predict(svmtest, test2, type = 'prob')
plot(psvm)
test3 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
test3$purchase <- as.factor(test3$purchase)
str(test3)
test3 <- select(test3, -purchase)
purchase <- predict(svmtest, test3, type = 'class')
test3 <- cbind(test3,purchase)
str(test3)
test3$purchase
reg4_tst <- C5.0(purchase ~ .,test3)
summary(reg4_tst)
plot(reg4_tst)

write.csv(test3,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_svm.csv")
############################################naivebayes###########################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train3 <- dat1[ind == 1,]
test3 <- dat1[ind == 2,]
naive_test <- naive_bayes(purchase ~., data = train3)

summary(naive_test)
print(naive_test)
plot(naive_test)
naivepred <- predict(naive_test,test2, type = 'class')
confusionMatrix(naivepred, test2$purchase)
plot(naivepred,test2$purchase)

pnb <- predict(naive_test, test3, type = 'prob')
plot(pnb)
test4 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
test4$purchase <- as.factor(test4$purchase)
str(test4)
test4 <- select(test4, -purchase)
purchase <- predict(naive_test, test4, type = 'class')
test4 <- cbind(test4,purchase)
str(test4)
test4$purchase
reg5_tst <- C5.0(purchase ~ .,test4)
summary(reg5_tst)
plot(reg5_tst)

write.csv(test3,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_nb.csv")
```

```
################################################ Decision Tree ID3 #####################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train <- dat1[ind == 1,]
test <- dat1[ind == 2,]
treex <- ctree(purchase ~., data = train)
print(treex)
plot(treex)

p <- predict(tree, test, type = 'response')
confusionMatrix(p, test$purchase)

test1x <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
test1x$purchase <- as.factor(test1x$purchase)
str(test1x)
test1x <- select(test1x, -purchase)
purchase <- predict(treex, test1x, type = 'response')
test1x <- cbind(test1x,purchase)
str(test1x)
test1x$purchase
reg2_tstx <- C5.0(purchase ~ .,test1x)
summary(reg2_tstx)
plot(reg2_tstx)

write.csv(test1x,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_ID3.csv")
```

These codes were developed using RStudio version 1.4.1717 and using R version 4.1.1("Kick Things")

5. An evaluation of the results obtained by each prediction method tried on the data

To do evaluation of the results, I have used seven techniques as mentioned before. So, if we move with the order of the code, it will be

1. KNN
2. Logistic Regression
3. Decision Tree (CART)
4. Decision Tree (5.0)
5. SVM
6. Naïve Bayes
7. Decision Tree (ID3)

So, my order of result evaluation will be same as mentioned above:

1. KNN

So, to evaluate KNN, this code snippet was used:

```
################################ knn algorithm ####################################################################################
dum_targ <- dummy_cols(dat1,select_columns=c("education","gender","marital_status","spouse_work",
                    "residential_status","product","employ_status","purchase"),
          remove_first_dummy= TRUE, remove_selected_columns= TRUE)
str(dum_targ)
k_train <- dum_targ[1:25000,]
k_test <- dum_targ[25001:30000,]
k_train_label <- dum_targ[1:25000,25]
k_test_label <- dum_targ[25001:30000,25]

k_test_pred <- knn(train = k_train, test = k_test,cl = k_train_label, k=10)
CrossTable(x = k_test_label, y = k_test_pred, prop.chisq = F)
```

So, we get the following results:

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  5000


               | prc_test_pred
prc_test_label |         0 |         1 | Row Total |
---------------|-----------|-----------|-----------|
             0 |      1675 |       755 |      2430 |
               |     0.689 |     0.311 |     0.486 |
               |     0.663 |     0.305 |           |
               |     0.335 |     0.151 |           |
---------------|-----------|-----------|-----------|
             1 |       853 |      1717 |      2570 |
               |     0.332 |     0.668 |     0.514 |
               |     0.337 |     0.695 |           |
               |     0.171 |     0.343 |           |
---------------|-----------|-----------|-----------|
  Column Total |      2528 |      2472 |      5000 |
               |     0.506 |     0.494 |           |
---------------|-----------|-----------|-----------|


> 1675+1717
[1] 3392
> 3392/5000
[1] 0.6784
```

So, Inference of this is that if we go with the following algorithm, we get a confusion matrix for the following and here we don't get any accuracy or any other output. So, using the basic concept of confusion matrix and using the following formulae, we get this

$$Accuracy = \left(\frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positive + False\ Negative}\right) \times 100$$

Accuracy = 67.84%

So, after seeing low accuracy, I planned to stop there and go further ahead to other algorithms which can give more accurate solutions.

2. Logistic Regression

So, to get solution from logistic regression, this code snippet was used:

```
############################# logisticregression##############################

set.seed(150000)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train2 <- dat1[ind == 1,]
test2 <- dat1[ind == 2,]
logit <- glm(purchase~., data = train2, family = "binomial")
summary(logit)
confint.default(logit)
predict(logit, newdata = test2, type = "response")
predres <- predict(logit, newdata = test2, type = "response")
predres <- round(predres)
predres<-ifelse(predres=="0","no","yes")
predres <- as.factor(predres)
confusionMatrix(predres,test2$purchase)

test3 <- dat_test
test3 <- select(test3, -purchase)
purchase <- predict(logit, newdata = test3, type = "response")
str(purchase)
purchase <- round(purchase)
test3 <- cbind(test3,purchase)
test3$purchase<-ifelse(test3$purchase=="0","No","Yes")
test3$purchase <- as.factor(test3$purchase)
str(test3)
logit_test <- C5.0(purchase ~ .,test3)
summary(logit_test)
plot(logit_test)
write.csv(test3,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_logistic.csv")
```

So, after execution of the codes, we get the following output:

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1741  908
       yes 1177 2060

               Accuracy : 0.6458
                 95% CI : (0.6334, 0.658)
    No Information Rate : 0.5042
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2909

 Mcnemar's Test P-Value : 4.378e-09

            Sensitivity : 0.5966
            Specificity : 0.6941
         Pos Pred Value : 0.6572
         Neg Pred Value : 0.6364
             Prevalence : 0.4958
         Detection Rate : 0.2958
   Detection Prevalence : 0.4501
      Balanced Accuracy : 0.6454

       'Positive' Class : no
```

So, from here, we get,

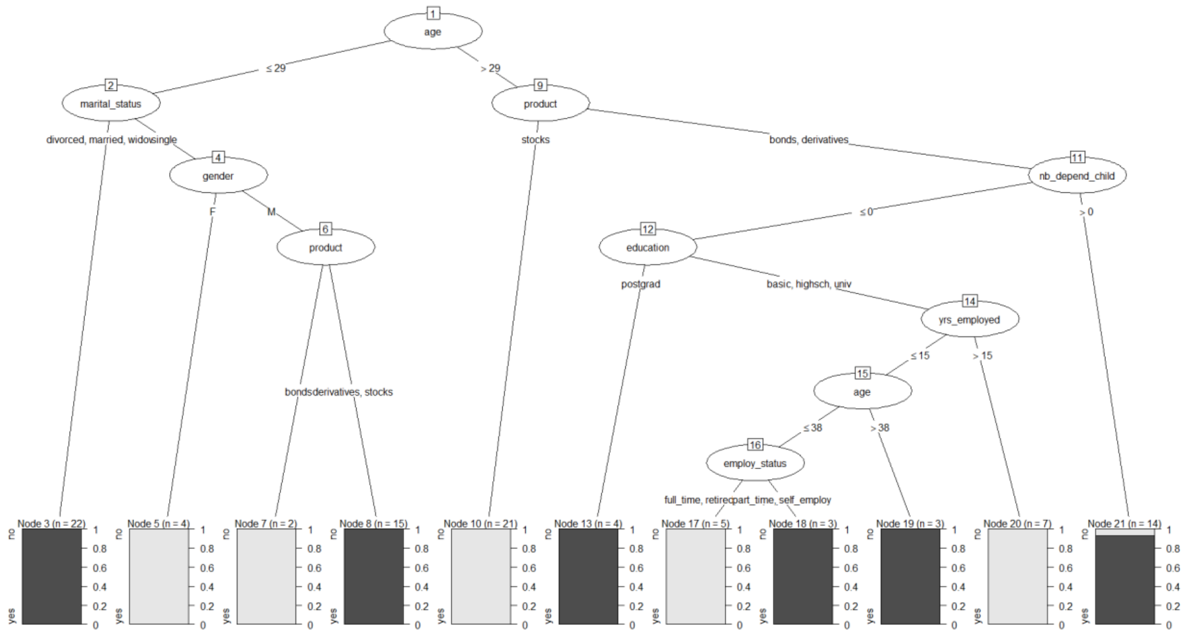
Accuracy = 64.58%; Kappa = 29.93%

But still we wanted to check how well it is working with the prediction dataset, so, we made the same model run on the prediction dataset, where we get the following outcome:

```
> test3$purchase
  [1] No  No  No  Yes Yes No  Yes No  No  No  No  Yes No  No  Yes No  Yes No  Yes Yes Yes Yes No  Yes No  No  No  Yes No  Yes Yes
 [32] No  Yes Yes Yes Yes No  Yes Yes No  Yes No  Yes Yes Yes No  Yes No  No  Yes Yes No  Yes No  Yes Yes No  Yes Yes No  No  No
 [63] Yes No  No  No  Yes No  Yes Yes Yes No  Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes No  No  Yes Yes Yes No  No  No  No  No  Yes
 [94] Yes Yes No  No  No  No  Yes
Levels: No Yes
```

Here, we can see our model is able to predict some result out of the given dataset but we are not sure how well it is able to give perfect results, so we use a decision tree so that we can understand the result better and we can get an approximate accuracy, so we get the outcome as:

```
Call:
C5.0.formula(formula = purchase ~ ., data = test3)


C5.0 [Release 2.07 GPL Edition]         Thu Dec 16 07:14:36 2021
-------------------------------

Class specified by attribute `outcome'

Read 100 cases (15 attributes) from undefined.data

Decision tree:

product = stocks: No (35/3)
product in {bonds,derivatives}:
:...yrs_current_address > 9: No (13/4)
    yrs_current_address <= 9:
    :...age <= 43: Yes (45/1)
        age > 43:
        :...marital_status in {divorced,single}: No (4)
            marital_status in {married,widowed}: Yes (3)


Evaluation on training data (100 cases):

            Decision Tree
          ----------------
          Size      Errors

             5    8( 8.0%)   <<


           (a)   (b)    <-classified as
          ----  ----
            45     1    (a): class No
             7    47    (b): class Yes


        Attribute usage:

        100.00% product
         65.00% yrs_current_address
         52.00% age
          7.00% marital_status


Time: 0.0 secs
```



Seeing both the outcomes, we can infer that logistic regression model was accurate enough to predict 92 correct answers and 8 incorrect and according to the decision tree, the net answer must be equal to 46 no and 54 yes, but we can't consider it yet as there are more algorithms to be checked out.

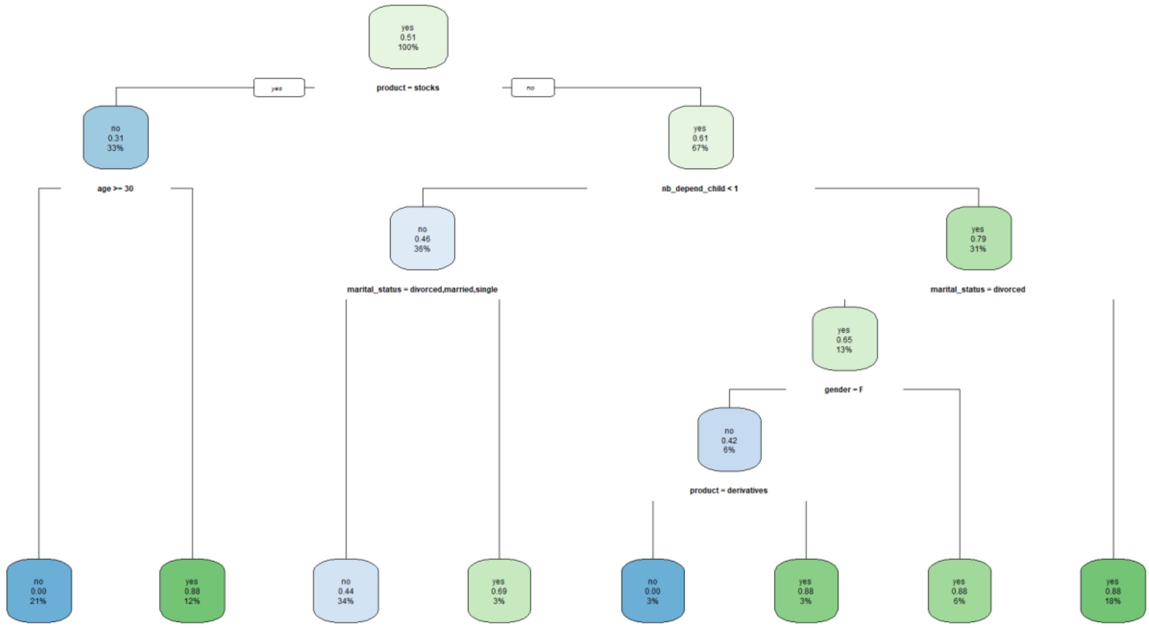
Also, from the tree plotted out, we can see that the most important factor considered by logistic regression is age and then followed by marital_status.

3. Decision Tree (CART)

To get the solution using this particular algorithm, we can go for the following code:

```
############################################### Decision Tree CART #######################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train <- dat1[ind == 1,]
test <- dat1[ind == 2,]
tree <- rpart(purchase ~., data = train, cp=0.015)
rpart.rules(tree)
rpart.plot(tree)
printcp(tree)
plotcp(tree)
p <- predict(tree, test, type = 'class')
confusionMatrix(p, test$purchase)

p1 <- predict(tree, test, type = 'prob')
p1 <- p1[,2]
r <- multiclass.roc(test$purchase, p1, percent = TRUE)
roc <- r[['rocs']]
r1 <- roc[[1]]
plot.roc(r1,
     print.auc=TRUE,
     auc.polygon=TRUE,
     grid=c(0.1, 0.2),
     grid.col=c("green", "red"),
     max.auc.polygon=TRUE,
     auc.polygon.col="lightblue",
     print.thres=TRUE,
     main= 'ROC Curve')

test1 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
       spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
       employ_status = factor(employ_status))
test1$purchase <- as.factor(test1$purchase)
str(test1)
test1 <- select(test1, -purchase)
purchase <- predict(tree, test1, type = 'class')
test1 <- cbind(test1,purchase)
str(test1)
test1$purchase
reg2_tst <- C5.0(purchase ~ .,test1)
summary(reg2_tst)
plot(reg2_tst)
```

So, from the following code we get this as the output regarding the prediction model:




```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  2573  872
       yes  353 2232

               Accuracy : 0.7968
                 95% CI : (0.7865, 0.8069)
    No Information Rate : 0.5148
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5954

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8794
            Specificity : 0.7191
         Pos Pred Value : 0.7469
         Neg Pred Value : 0.8634
             Prevalence : 0.4852
         Detection Rate : 0.4267
   Detection Prevalence : 0.5713
      Balanced Accuracy : 0.7992

       'Positive' Class : no
```

So, we get the following result:

Accuracy = 79.68% ; kappa = 59.54%

The graph above gives us the idea how well our decision tree algorithm is able to make branches based on the given data and given parameters.

So, after this, we see an increase in our data accuracy that can help us to consider it as one of the viable solutions for the problem. But we are not sure yet as we have to run it on the prediction dataset too.

So, after running the code for the prediction dataset, we get the following:

```
> test1$purchase
  [1] no  no  yes no  yes no  yes no  no  no  no  no  no  no  no  no  no  no  yes yes no  yes no  yes no  no  yes yes yes yes yes
 [32] no  yes yes yes yes no  yes no  yes no  no  no  no  no  yes yes no  no  yes yes no  yes no  no  yes no  no  yes no  no  yes
 [63] no  no  yes no  no  yes no  yes no  no  yes no  no  no  yes no  yes yes no  no  no  no  yes yes no  yes no  yes yes no  no
 [94] yes yes no  yes no  yes no
Levels: no yes
```

```
Call:
C5.0.formula(formula = purchase ~ ., data = test1)


C5.0 [Release 2.07 GPL Edition]         Thu Dec 16 07:49:00 2021
-------------------------------

Class specified by attribute `outcome'

Read 100 cases (15 attributes) from undefined.data

Decision tree:

nb_depend_child > 0:
:...product in {bonds,derivatives}: yes (25/1)
:   product = stocks:
:   :...age <= 30: yes (6)
:       age > 30: no (12)
nb_depend_child <= 0:
:...marital_status = widowed: yes (4)
    marital_status in {divorced,married,single}:
    :...product in {bonds,derivatives}: no (36)
        product = stocks:
        :...age <= 30: yes (8)
            age > 30: no (9)


Evaluation on training data (100 cases):

            Decision Tree
          ----------------
          Size      Errors

             7    1( 1.0%)   <<


           (a)   (b)    <-classified as
          ----  ----
            57     1    (a): class no
                  42    (b): class yes


        Attribute usage:

        100.00% nb_depend_child
         96.00% product
         57.00% marital_status
         35.00% age


Time: 0.0 secs
```

So, we can see that our decision tree is able to predict 99 cases correctly and it is showing that our solution set should have 58 no and 42 yes but we can't say that surely for now as the cross validation is being tested based on the initial decision tree.

If we see the attribute usage, it shows our decision tree is using dependable child as the major factor and then the product.

4. Decision Tree (5.0)

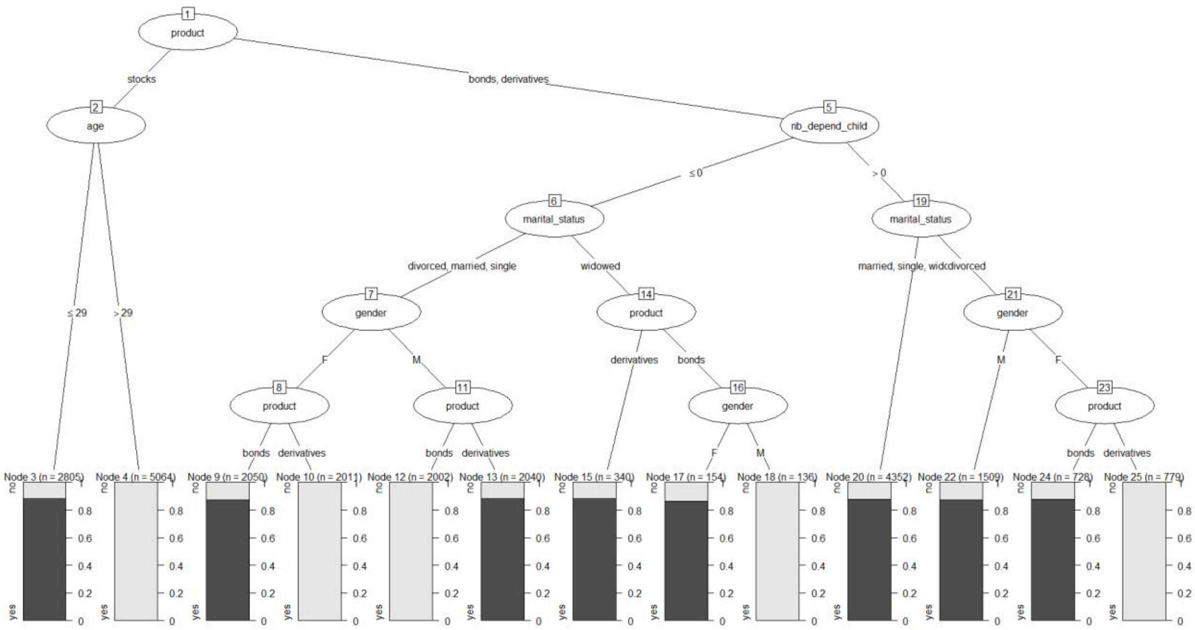
Decision tree is basically much advanced version of ID3 and it has become as the major standard for corporate world to use it as the primary tree type to do any machine learning regarding decision tree. It uses entropy and information gain as the major factor to work out to a solution.

So, to implement a decision tree 5.0, we use the following code snippet:

```
############################################## Decision Tree C5.0 ######################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train1 <- dat1[ind == 1,]
test1 <- dat1[ind == 2,]
tree1 <- C5.0(purchase ~., data = train1, cp = 0.0025)
plot(tree1)

p5 <- predict(tree1, test1, type = 'class')
confusionMatrix(p5, test1$purchase)

p15 <- predict(tree1, test1, type = 'prob')
p15 <- p15[,2]
r5 <- multiclass.roc(test1$purchase, p15, percent = TRUE)
roc5 <- r5[['rocs']]
r15 <- roc5[[1]]
plot.roc(r15,
     print.auc=TRUE,
     auc.polygon=TRUE,
     grid=c(0.1, 0.2),
     grid.col=c("green", "red"),
     max.auc.polygon=TRUE,
     auc.polygon.col="lightblue",
     print.thres=TRUE,
     main= 'ROC Curve')

test2 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
           spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
           employ_status = factor(employ_status))
test2$purchase <- as.factor(test2$purchase)
str(test2)
test2 <- select(test2, -purchase)
purchase <- predict(tree1, test2, type = 'class')
test2 <- cbind(test2,purchase)
str(test2)
test2$purchase
reg3_tst <- C5.0(purchase ~ .,test2)
summary(reg3_tst)
plot(reg3_tst)

write.csv(test2,"D://UCC STUFF//IS6052//Assignment//Market_pred_result2.csv")
```

So, after executing the model creation code of decision tree 5.0, we get the following as the output:

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  2491    0
       yes  435 3104

               Accuracy : 0.9279
                 95% CI : (0.921, 0.9343)
    No Information Rate : 0.5148
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.855

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8513
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.8771
             Prevalence : 0.4852
         Detection Rate : 0.4131
   Detection Prevalence : 0.4131
      Balanced Accuracy : 0.9257

       'Positive' Class : no
```



From the above pictures, we can infer that:

Accuracy = 92.79% ; kappa = 85.5%

It shows that this decision tree is giving us pretty good results, that means it can be considered as one of the best solutions for the problem, but we can't give it a confirm result, but we have still other algos to check and also, we have to check how well it is going with our prediction dataset.

Also, if we see the tree, we can see that it is using product as the first factor, which shows that it is working on the right direction as per information-gain we checked at first.

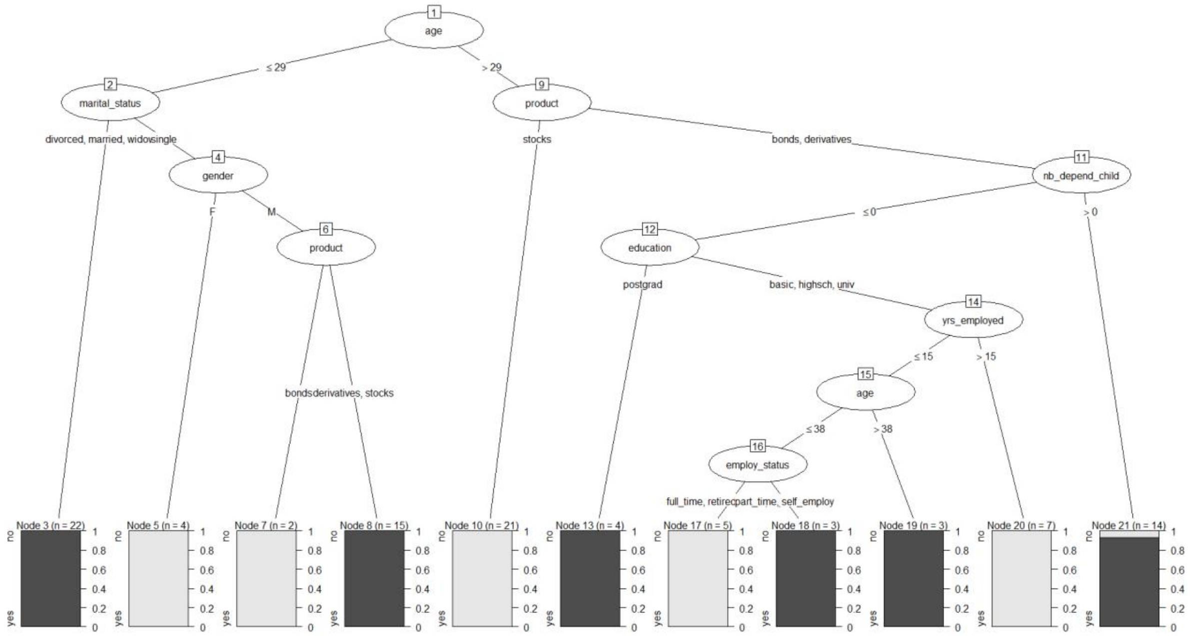
If we test with the prediction dataset, we get the following answer:

```
> test2$purchase
  [1] no  no  yes yes yes no  yes no  no  no  no  no  yes yes no  no  yes no  yes yes yes yes no  yes no  no  yes yes yes yes yes
 [32] no  yes yes yes yes no  no  yes yes no  yes yes no  no  yes yes no  no  yes yes no  yes no  no  yes no  yes yes no  no  yes
 [63] yes no  yes no  yes yes no  yes yes no  yes yes no  yes yes no  yes yes yes no  no  yes yes yes yes yes no  yes yes yes no
 [94] yes yes no  yes yes yes no
Levels: no yes
```

C

```
Decision tree:

age <= 29:
:...marital_status in {divorced,married,widowed}: yes (22)
:   marital_status = single:
:   :...gender = F: no (4)
:       gender = M:
:       :...product = bonds: no (2)
:           product in {derivatives,stocks}: yes (15)
age > 29:
:...product = stocks: no (21)
    product in {bonds,derivatives}:
    :...nb_depend_child > 0: yes (14/1)
        nb_depend_child <= 0:
        :...education = postgrad: yes (4)
            education in {basic,highsch,univ}:
            :...yrs_employed > 15: no (7)
                yrs_employed <= 15:
                :...age > 38: yes (3)
                    age <= 38:
                    :...employ_status in {full_time,retired,
                        :                 unemployed}: no (5)
                        employ_status in {part_time,self_employ}: yes (3)


Evaluation on training data (100 cases):

	    Decision Tree
	  ----------------
	  Size      Errors

	    11    1( 1.0%)   <<


	   (a)   (b)    <-classified as
	  ----  ----
	    39     1    (a): class no
	          60    (b): class yes


	Attribute usage:

	100.00% age
	 74.00% product
	 43.00% marital_status
	 36.00% nb_depend_child
	 22.00% education
	 21.00% gender
	 18.00% yrs_employed
	  8.00% employ_status
```

It is showing us that on prediction dataset, we can see that even this is giving us 99 of the solution correct but in this, we are seeing a change in the values inside confusion matrix, in which 39 are correctly guessed as no, 60 correctly guessed as yes and 1 wrong, which shows that the dataset is containing 40 no and 60 yes as the solution, but we can’t say it yet that it is the solution for our dataset. So, we need to test other algorithms.



Also, we see the plot, we can say that it can be the optimal solution for our problem, but we need to look for something better that can be used as an optimal solution.

5. Support Vector Machines (SVM):

Support Vector Machine are type of algorithm that can be used for both regression as well as classification but it is mostly used for classification. It is based on non-linear classification that can’t be used for unsupervised learning.

So, to test in our case, we will use the following code snippet:

```
################################################################ SVM ######################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train2 <- dat1[ind == 1,]
test2 <- dat1[ind == 2,]
svmtest <- svm(purchase ~., data = train2)

summary(svmtest)
print(svmtest)
svmpred <- predict(svmtest,test2, type = 'class')
confusionMatrix(svmpred, test2$purchase)
plot(svmpred,test2$purchase)

psvm <- predict(svmtest, test2, type = 'prob')
plot(psvm)
test3 <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
test3$purchase <- as.factor(test3$purchase)
str(test3)
test3 <- select(test3, -purchase)
purchase <- predict(svmtest, test3, type = 'class')
test3 <- cbind(test3,purchase)
str(test3)
test3$purchase
reg4_tst <- C5.0(purchase ~ .,test3)
summary(reg4_tst)
plot(reg4_tst)

write.csv(test3,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_svm.csv")
```

After executing codes for model creation, we get the following outcomes

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  2458   73
       yes  468 3031

               Accuracy : 0.9103
                 95% CI : (0.9028, 0.9174)
    No Information Rate : 0.5148
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8197

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8401
            Specificity : 0.9765
         Pos Pred Value : 0.9712
         Neg Pred Value : 0.8662
             Prevalence : 0.4852
         Detection Rate : 0.4076
   Detection Prevalence : 0.4197
      Balanced Accuracy : 0.9083

       'Positive' Class : no
```

We get the following inference from the data mentioned above:

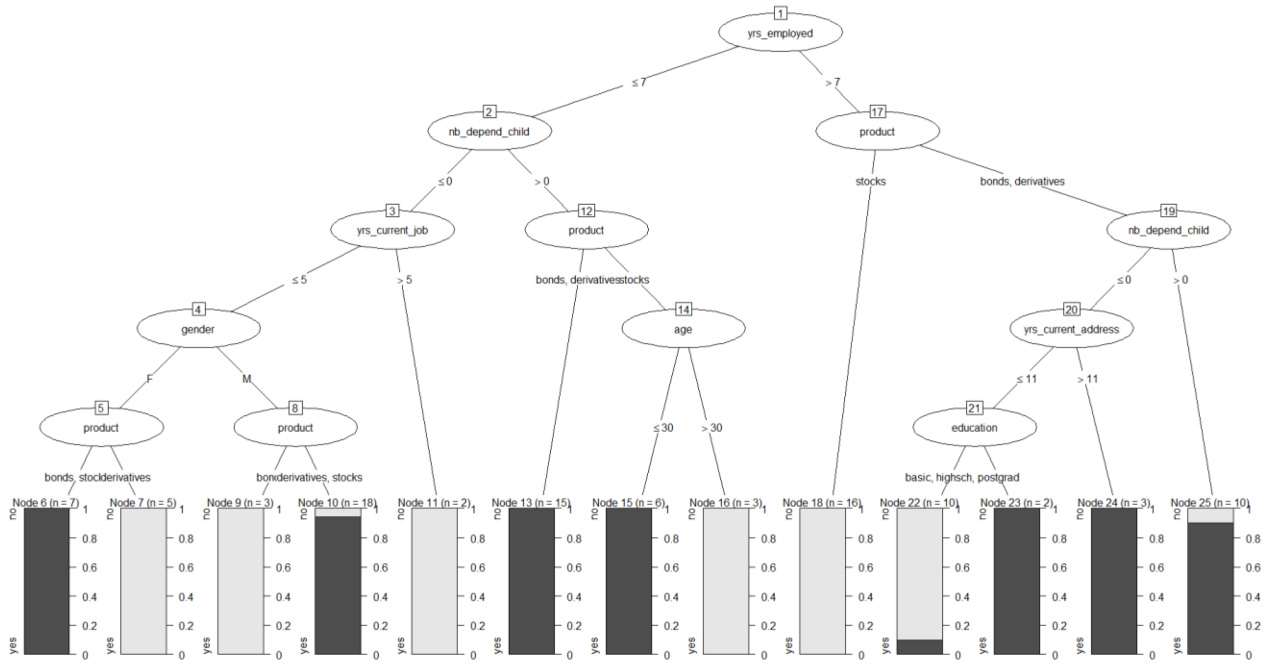
Accuracy = 91.03% ; kappa = 81.97%

It shows that it is also a good model as the accuracy is approx. 91%, it can also be considered as one of the solutions for the problem. But we still need to check for more algorithms for the solution. But before we move to next solution, we need to check for the performance on the prediction dataset.

So, we get these results after we run the codes on prediction dataset:

```
> test3$purchase
  [1] no  no  no  yes yes yes yes no  no  no  no  no  yes yes no  no  yes no  yes yes yes yes no  yes no  no  no  yes yes yes yes
 [32] no  yes yes yes yes no  no  yes yes no  yes yes no  no  yes yes no  no  yes yes no  yes no  no  yes no  yes yes no  yes yes
 [63] yes no  yes no  yes yes no  yes yes no  yes yes no  yes yes no  yes yes yes no  no  yes yes yes yes yes no  yes yes yes no
 [94] yes yes no  yes yes yes no
Levels: no yes
```

```
Evaluation on training data (100 cases):

	    Decision Tree
	  ----------------
	  Size      Errors

	    13    3( 3.0%)   <<


	   (a)   (b)    <-classified as
	  ----  ----
	    38     2    (a): class no
	     1    59    (b): class yes


	Attribute usage:

	100.00% yrs_employed
	 98.00% product
	 84.00% nb_depend_child
	 35.00% yrs_current_job
	 33.00% gender
	 15.00% yrs_current_address
	 12.00% education
	  9.00% age


Time: 0.0 secs
```



From the above pictures, we can say that the model was able to guess 97 out of 100 correct, out of which 59 was guessed as yes and 38 as no and as per the confusion matrix, the no. of yes is 60 and no is 40.

Also, it is taking years employed as the main node, which was decided as per the predictions made by the model.

6. Naïve Bayes

It is one of the classification algorithms that work on the basis of Bayes theorem. It is totally based on probability and it calculates the probability of occurrence of the event.

So, to find how suitable it is for our data, we will make model and test it and to do that, we used this code snippet:

```
############################################################ naivebayes ##################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train3 <- dat1[ind == 1,]
test3 <- dat1[ind == 2,]
naive_test <- naive_bayes(purchase ~., data = train3)

summary(naive_test)
print(naive_test)
plot(naive_test)
naivepred <- predict(naive_test,test2, type = 'class')
confusionMatrix(naivepred, test2$purchase)
plot(naivepred,test2$purchase)
```

After running the code, we get this as output:

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1640 1096
       yes 1286 2008

               Accuracy : 0.605
                 95% CI : (0.5925, 0.6173)
    No Information Rate : 0.5148
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2078

 Mcnemar's Test P-Value : 0.0001077

            Sensitivity : 0.5605
            Specificity : 0.6469
         Pos Pred Value : 0.5994
         Neg Pred Value : 0.6096
             Prevalence : 0.4852
         Detection Rate : 0.2720
   Detection Prevalence : 0.4537
      Balanced Accuracy : 0.6037

       'Positive' Class : no
```

From this data, we can infer that the accuracy is 60.5% and the kappa value for the data is 20.78%, which is already less than the algorithms that we applied before, so it will be useless for us to even test it with the prediction dataset as we already know that there are models which are giving better results.

So, we will move on to the next algorithm.

7. Decision Tree (ID3)

It is a type of decision tree which is used for classification. In this, we calculate each and every node in the tree on the basis of Information Gain and Entropy.

So, we used this code snippet for ID3:

```
################################################# Decision Tree ID3 ########################################################################
set.seed(1234)
ind <- sample(2, nrow(dat1), replace = T, prob = c(0.8, 0.2))
train <- dat1[ind == 1,]
test <- dat1[ind == 2,]
treex <- ctree(purchase ~., data = train)
print(treex)
plot(treex)

p <- predict(tree, test, type = 'response')
confusionMatrix(p, test$purchase)

test1x <- mutate(data_test,gender = factor(gender), marital_status = factor(marital_status), education = factor(education),
        spouse_work = factor(spouse_work), residential_status = factor(residential_status), product = factor(product),
        employ_status = factor(employ_status))
test1x$purchase <- as.factor(test1x$purchase)
str(test1x)
test1x <- select(test1x, -purchase)
purchase <- predict(treex, test1x, type = 'response')
test1x <- cbind(test1x,purchase)
str(test1x)
test1x$purchase
reg2_tstx <- C5.0(purchase ~ .,test1x)
summary(reg2_tstx)
plot(reg2_tstx)

write.csv(test1x,"D://UCC STUFF//IS6052//Assignment//Market_pred_result_ID3.csv")
```

So, after running the code up to prediction, we get these as output

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  2609  872
       yes  317 2232

               Accuracy : 0.8028
                 95% CI : (0.7925, 0.8128)
    No Information Rate : 0.5148
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6074

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8917
            Specificity : 0.7191
         Pos Pred Value : 0.7495
         Neg Pred Value : 0.8756
             Prevalence : 0.4852
         Detection Rate : 0.4327
   Detection Prevalence : 0.5773
      Balanced Accuracy : 0.8054

       'Positive' Class : no
```

As we can see from the picture before, we can infer that accuracy is 80.28% and kappa is 60.74%. Just like before, it doesn't make sense to go for further steps as we already have a better solution for the same. So, we won't test it on the prediction dataset.

6. Comparative analysis of the results

So, after working on 7 algorithms, we can now compare the results based on their accuracy and the kappa value

Accuracy is defined as the ratio of true value to total values whereas kappa value is defined as the factor which shows how well that particular set of parameters are corelated to the decision i.e., how well is the purchase is able to corelate with all the other variables.

So, we can make a comparison table for that:

Algorithm	Accuracy	Kappa
KNN	67.84	---
Logistic	64.58	29.93
DTree(CART)	79.78	59.54
DTree(C5.0)	92.79	85.5
SVM	91.03	51.48
Naïve Bayes	60.5	20.78
DTree(ID3)	80.28	60.74

So, after comparing these two, we can say that the best algorithm for us would be using for this data would be Decision Tree 5.0

So, if we move forward with decision tree 5.0 and start to look for solution, we can see a data file outcome

```
> test2$purchase
  [1] no  no  yes yes yes no  yes no  no  no  no  no  yes yes no  no  yes no  yes yes yes yes no  yes no  no  yes yes yes yes yes
 [32] no  yes yes yes yes no  no  yes yes no  yes yes no  no  yes yes no  no  yes yes no  yes no  no  yes no  yes yes no  no  yes
 [63] yes no  yes no  yes yes no  yes yes no  yes yes no  yes yes no  yes yes yes no  no  yes yes yes yes yes no  yes yes yes no
 [94] yes yes no  yes yes yes no
Levels: no yes
```

And if we go with this outcome, we can say that we have a perfect target audience whom we can select and go for selling the product.

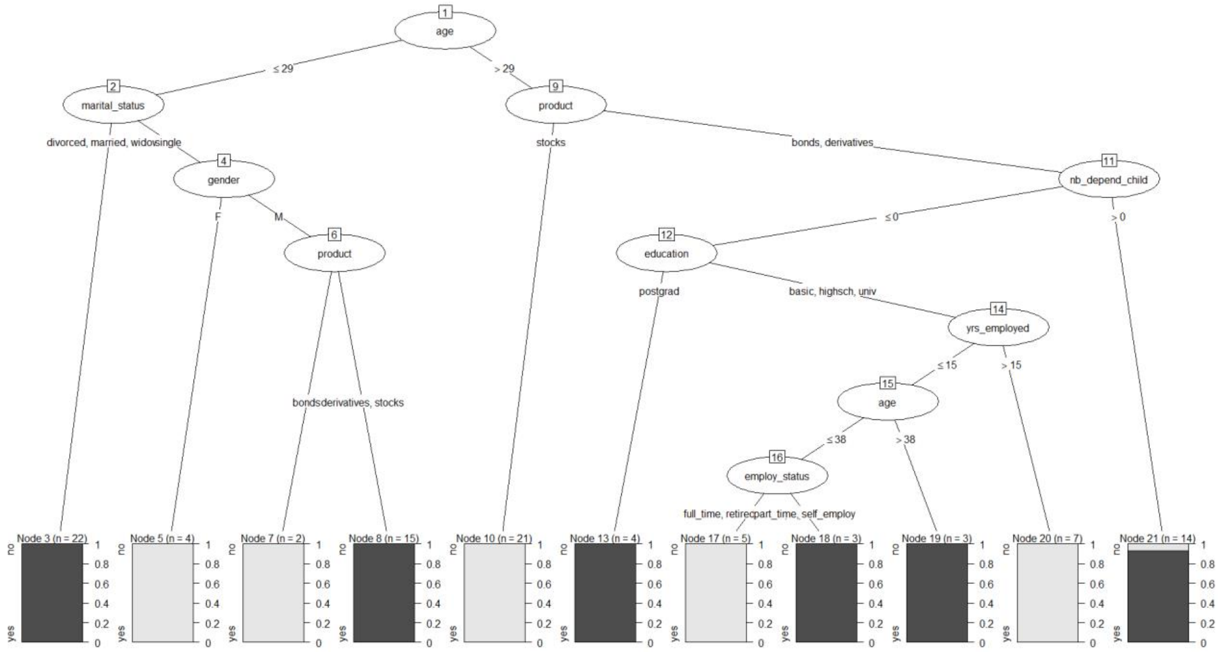
Also, we can say with this that it can be used to give a final solution for ABSA's market problem.

7. Final recommendation to ABSA on which customers the sales force should choose to target

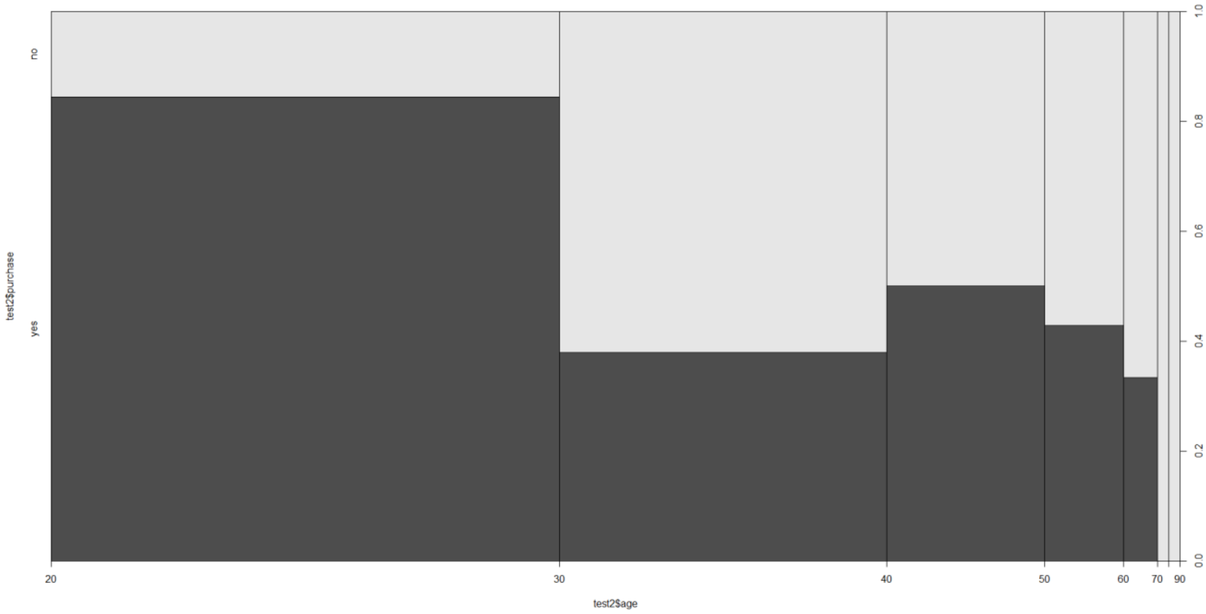
So, after seeing all the possible solutions that are given by the classification algorithm, we can say that there are few things that ABSA must understand so that they can understand which customers they should target.

So, as per our presented algorithms and models, we selected Decision Tree 5.0 and when we do analysis on that we get these pointers:

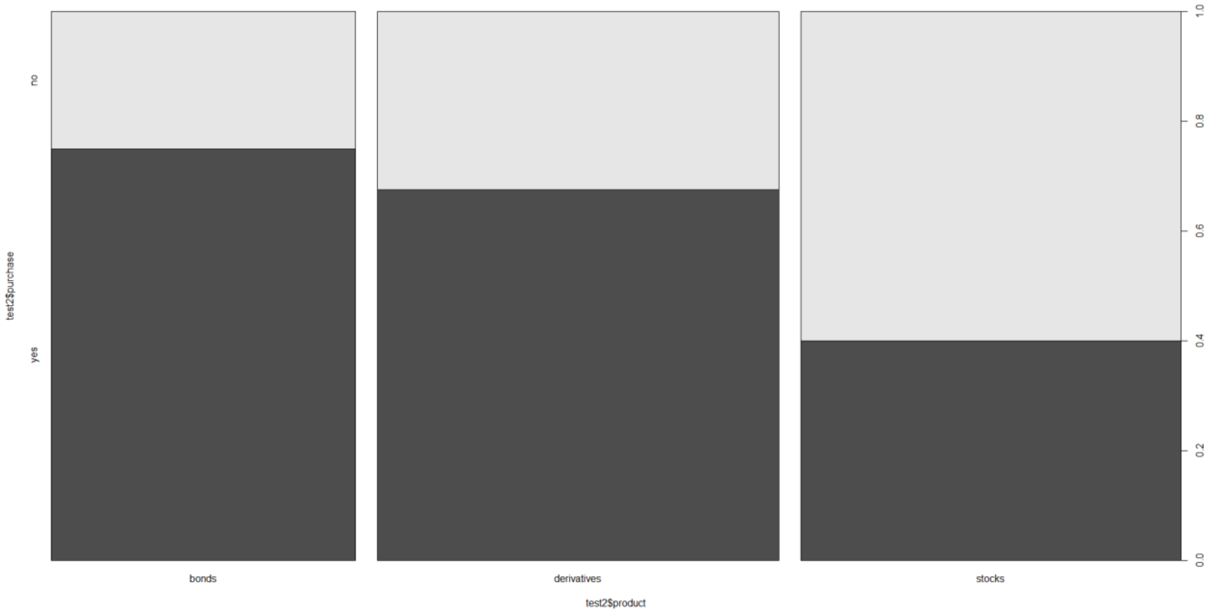
```
Decision tree:

age <= 29:
:...marital_status in {divorced,married,widowed}: yes (22)
:   marital_status = single:
:   :...gender = F: no (4)
:       gender = M:
:       :...product = bonds: no (2)
:           product in {derivatives,stocks}: yes (15)
age > 29:
:...product = stocks: no (21)
    product in {bonds,derivatives}:
    :...nb_depend_child > 0: yes (14/1)
        nb_depend_child <= 0:
        :...education = postgrad: yes (4)
            education in {basic,highsch,univ}:
            :...yrs_employed > 15: no (7)
                yrs_employed <= 15:
                :...age > 38: yes (3)
                    age <= 38:
                    :...employ_status in {full_time,retired,
                        :                 unemployed}: no (5)
                        employ_status in {part_time,self_employ}: yes (3)


Evaluation on training data (100 cases):

            Decision Tree
          ----------------
          Size      Errors

            11    1( 1.0%)   <<


           (a)   (b)    <-classified as
          ----  ----
            39     1    (a): class no
                  60    (b): class yes


        Attribute usage:

        100.00% age
         74.00% product
         43.00% marital_status
         36.00% nb_depend_child
         22.00% education
         21.00% gender
         18.00% yrs_employed
          8.00% employ_status
```



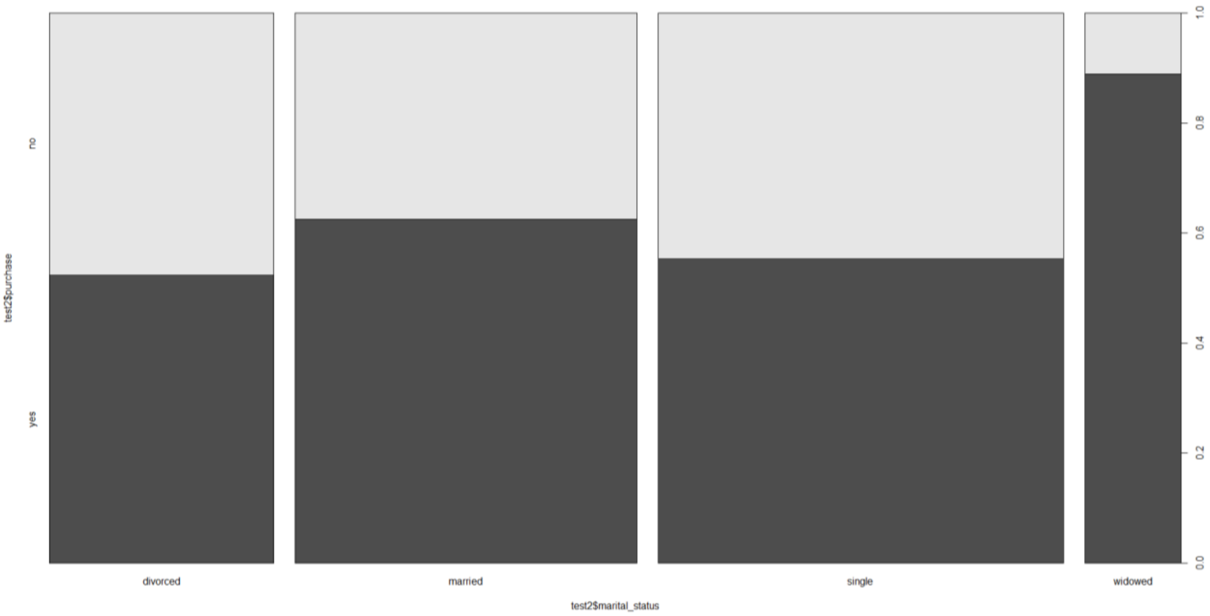
age
≤ 29
> 29
marital_status
divorced, married, widowsingle
gender
F
M
product
bondsderivatives, stocks
product
stocks
bonds, derivatives
nb_depend_child
≤ 0
> 0
education
postgrad
basic, highsch, univ
yrs_employed
≤ 15
> 15
age
≤ 38
> 38
employ_status
full_time, retirecpart_time, self_employ
Node 3 (n = 22)
Node 5 (n = 4)
Node 7 (n = 2)
Node 8 (n = 15)
Node 10 (n = 21)
Node 13 (n = 4)
Node 17 (n = 5)
Node 18 (n = 3)
Node 19 (n = 3)
Node 20 (n = 7)
Node 21 (n = 14)



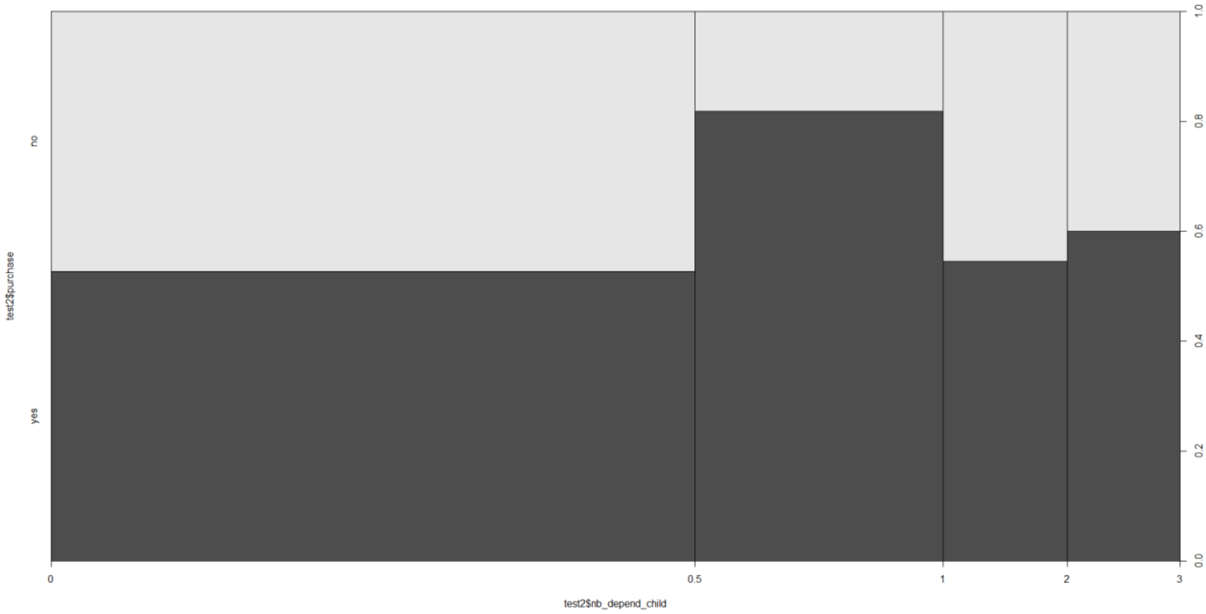
test2$purchase
no
yes
test2$age
20
30
40
50
60
70
90

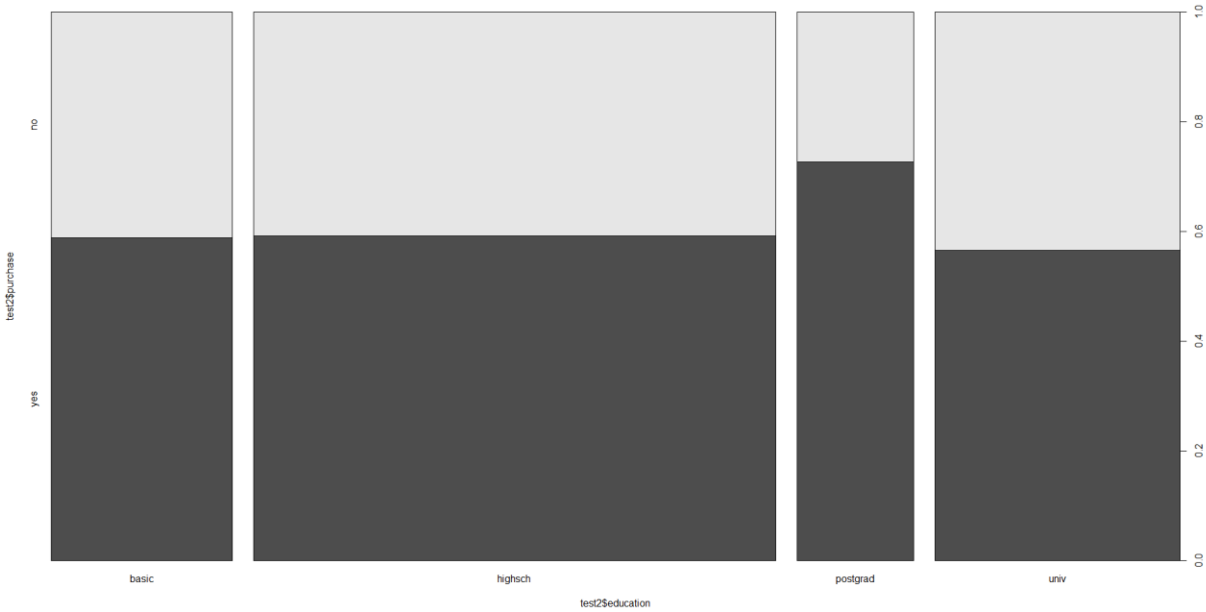


no
test2$purchase
yes
bonds
derivatives
stocks
test2$product
1.0
0.8
0.6
0.4
0.2
0.0



no
test2$purchase
yes
divorced
married
single
widowed
test2$marital_status
1.0
0.8
0.6
0.4
0.2
0.0





From these, we can devise some pointers that can be considered to make final recommendation to ABSA

1. They should consider that age is most important factor that should be taken at utmost priority
2. Then they should consider the type of investment product the customers are buying.
3. Then next factor they should look into their marital status followed by no. of dependable child/children.
4. Then they should consider education level and the gender of the customer
5. Then they should look into years employed and status of employment of the customer.

So, after seeing all the points, ABSA should target audiences who are:

1. Age less than 30 and they are either divorced, married or widowed
2. Age less than 30, who are single, who are Male and who have bought derivatives or stocks
3. Age greater than 29, invested in bonds or derivatives, who has 0 dependable child under them and the education level is post-graduation
4. Age greater than 29, invested in bonds or derivatives, who has at least 1 dependable child under them.
5. Age greater than 29, invested in bonds or derivatives, who has 0 dependable child under them, the education level is not post-graduation, employment years is less than 15 and age is greater than 38
6. Age greater than 29, invested in bonds or derivatives, who has 0 dependable child under them, the education level is not post-graduation, employment years is less than 15, age is less or equal to 38 and they are doing either part time job or being self employed

8.Additional data that would be useful to make better predictions in the future

So, after seeing all the data and analysis, there can be new points that can help in detecting more customer interest.

1. Checking the credit score of the customer. As credit score is directly linked with amount of transaction done by the customer, it can help the bank to understand the customer better, which will give better idea about where the customer will invest in future.
2. Checking the amount of taxes filed by the customer. As taxes are also one of the major aspects of financial status, it could help the bank to understand the investment portfolio of the customer in the near future.