# POLICY RECOMMENDATION SYSTEM

Submitted in partial fulfilment

Of

Project in Bachelor of Technology


Submitted by

1. **GSV Vagdevi**

2. **C Cara Evangeline**

3. **M Sandhya**

Under the Supervision of

Mr.BKSP Kumar Raju Alluri


## NATIONAL INSTITUTE OF TECHNOLOGY,

## Andhra Pradesh.

# INDEX

# SUMMARY


- C5.0, Logistic Regression, Bayesian Network, Support Vector Machine, Neural Networks are the algorithms used to implement the base paper to predict if a customer accepts a term deposit on receiving a call. Where SVM gives the highest accuracy in predicting the class labels.

- Hierarchical clustering technique is used to predict the telemarketing skills of an employee beforehand.

- Some of the customers who took the term deposit in the previous telemarketing campaign may not be able to take the term deposit in the present Campaign. Association Rules are extracted considering this situation to ensure the reasons behind this.

- Weka is used to know which attribute is the most deciding factor for a person to accept term deposit.

- Classification techniques are used to predict the class label for new customer where class label gives the level of performance of customer in repaying the loans.

- Hierarchical Clustering is used to recommend the most probable bank policies to a customer through telemarketing by banks.

- K means clustering algorithm is used to recommend the bank policies to a customer along with profit percentage and risk percentage for the respective policy. Even classification techniques such as knn,C5.0 are used to predict the most probable policy for a customer.

- Association mining using Apriori template matching decides which type of customers to be chosen for telemarketing to recommend a policy as calling every customer who previously approached bank would not be fruitful.

| Module | Objective | Challenges | Methodology | Outcome |
|---|---|---|---|---|
| 1A | To predict whether a customer will be able to repay loan or not. | Dataset from internet doesn't have deciding attributes | Considering various datasets we took 29 useful attributes and constructed our own dataset by applying various conditions | We can predict whether a new customer will be able to repay or not |
| 1B | To predict whether a customer will accept term deposit or not | Interesting questions cannot be answered because of less attributes in the standard dataset. | Applied association mining | Various questions can be answered based on antecedent and consequence |
| 2 | To suggest 1 most probable policy for a customer | Every policy has so many deciding attributes which leads to spurious database table | We reduced the number of attributes and policies and considered only few common deciding attributes | we can predict which policy most likely a customer will go for |
| 5B | To calculate new employee efficiency | Speech recognition couldn't be done due to time limitation. | Generated random dataset | Efficiency of an employee can be calculated |
| 5A | In bank marketing: previous=yes and y=no | Factors deciding the change of mind of a customer from yes to no. No classification algorithm can perform this. | We used template matching | 11 rules were generated and based on the confidence ,lift value the most interesting rule is obtained |
| 6 | Customer called to inform about the policies | All the customer should not be informed about all policies | We performed Clustering along with sorting | Recommendation of policy in order based on customer's profile |

# INTRODUCTION

With the latest news showing clients of large banks fleeing to smaller credit unions and local banks and as banking competition becomes more and more global and intense, banks have to fight more creatively and proactively to gain or even maintain market shares. Data mining is becoming strategically important area for banking sector. It is a process of analyzing the data from various perspectives and summarizing it into valuable information.

## MOTTO OF DOING THE PROJECT:

Data mining assists the banks to look for hidden pattern in a group and discover unknown relationship in the data.

## Techniques which are being emphasized

***Association Mining:***

This technique was used to unearth unsuspected data dependencies. In other words, it tried to detect data items that are associated or connected or correlated with each other which are not obvious previously. More formally, the task was to uncover hidden associations from a large database. The idea was to derive a set of strong association rules in the form of “A1∧A2∧ … Am∧ B1∧ B2∧ … Bn” where Aj (for i∈{1… m}) and Bj (for j∈{1… n}) are set of attribute-values from the relevant data sets in a database. Association rules are employed in application areas including web usage mining, intrusion detection and bioinformatics. Typically all association rules are not interesting. From a large data set, a very large and a high proportion of the rules mined will be usually of little value. An associative relationship is considered to be useful if it satisfies a predefined support and confidence values . Hence, a rule is discarded if it does not satisfy this minimum support threshold and minimum confidence threshold. All these discovered strong association rules may not be interesting enough to present. Additional analysis need to be performed to uncover interesting statistical correlations between associated attribute-value pairs .

***Classification:***

It is employed when the classes of data in the population are known. For example, in the case of detecting fraudulent banking transactions from a bank’s transactions database, there can only be two classes, namely fraudulent and non-fraudulent. It constructs a model from the sample data items with known class labels and use this model to predict the class of objects in the population whose classes are not known. Each tuple from the database contains one or more predicting attributes which determines the predicted class label of the tuple according to the constructed model. In the banking scene, classification technique is employed for Fraud detection (both corporate and credit fraud). These models are constructed usually using a decision tree model or a neural network model.

A decision tree is a flow chart like recursive structure to express classification rules where each node specifies a test on an attribute value, each branch specifies a mutually exclusive outcome of the test together with a subsidiary decision tree for each outcome and tree leaves represent classes or class distributions.

***Clustering:***

Clustering is similar to classification. But subtle difference is that classes are not known before. Clustering is used to generate class labels. The objects are classified or grouped based on the principle of maximizing the similarity within a class based on the observed pattern. A regularly used and the simplest of clustering algorithms is K-means algorithm. Concept formation is a closely related process to clustering and is used to learn summaries from data. This process integrates learning and classification tasks to identify summaries and organize learned summaries into a hierarchy. In banking area, clustering and concept formation can be employed for classifying customers with same kind of transactions or queries or profiles or subscribe to similar policies or has similar risk aptitude. Knowledge about these classes will help banks to design products to each class of customers and can embark on targeted and more effective marketing campaigns.

# ABOUT DATASET

## Module 1A:

loan.arff(400 tuples) and loantestm.arff(200 tuples)   [Number of attributes:29]

loannumeric.arff and loan_testnumeric.arff (nominal to numeric converted files)

- id :numeric
- sex nominal {female,male}
- education:nominal {basic,high_school,university_degree,professional_course,illiterate}
- marital_status :nominal {single,divorced,married}
- age: numeric
- no_of_children :nominal {0,1,2,3,4}
- no_of_employees_in_family: nominal {0,1,2,3}
- net_income_of_family :numeric
- no_of_dependence :nominal {0,1,2,3}
- income :numeric
- region :nominal {inner_city,town,rural,sub_urban} ->region of stay
- car :nominal {yes,no} ->whether car loan is taken or not
- mortgage: nominal {yes,no} ->loan in which property or real estate used as collateral
- loan_adjustment: nominal {adjusted,not_adjusted}
- class_label nominal: {excellent,very_good,good,average,bad} savings_account: nominal {yes,no}
- credit_account :nominal {yes,no}
- housing :nominal {rent,own,inherited}
- loan :nominal {yes,no} ->present loan we are working on
- job :nominal {services,technician, etc..}
  dr =>debit cr=>credit
- min_dr_amt: nominal {lessequal25,lessequal50,lessequal75,above75}
- max_dr_amt :nominal{lessequal25,lessequal50,lessequal75,above75}
- total_dr_amt:nominal {lessequal100,lessequal200,lessequal300,lessequal400,lessequal500,above 500}
- total_dr_vocher: nominal {lessequal3,lessequal6,lessequal10,above10}
- min_cr_amt :nominal{lessequal25,lessequal50,lessequal75,above75}
- max_cr_amt :nominal{lessequal25,lessequal50,lessequal75,above75}
- total_cr_amt:nominal{lessequal100,lessequal200,lessequal300,lessequal400 ,lessequal500,above500}
- total_cr_vocher :nominal{lessequal3,lessequal6,lessequal10,above10}
- principal_amt :nominal {exceed_limit,lessequal50,above50}

**Links**

1. Base Paper:

➔ 201O International Conference on Computer Application and System Modeling (ICCASM 2010)
Data Mining Application in Banking-Customer Relationship Management

➔ International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.5, No.2, March 2015
A DATA MINING APPROACH TO PREDICT PROSPECTIVE BUSINESS SECTORS FOR LENDING IN RETAIL BANKING USING DECISION TREE

2. http://facweb.cs.depaul.edu/mobasher/classes/ect584/WEKA/classify.html

## Module 1B,5A,5B:

bank_ful1.csv(45211 tuples) [Number of attributes:17]

- age (numeric)
- job : type of job (categorical: 'admin.', 'bluecollar', 'entrepreneur', 'housemaid','management','retired','selfemployed','services','student', 'technician','unemployed','unknown)
- marital : marital status(categorical:'divorced','married','single','unknown';
- education(categorical:'basic.4y','basic.6y','basic.9y','high.school','illiterate','professional.course','university.degree','unknown')
- default: has credit in default? (categorical: 'no','yes','unknown')
- housing: has housing loan? (categorical: 'no','yes','unknown')
- loan: has personal loan? (categorical: 'no','yes','unknown')
  # related with the last contact of the current campaign:
- contact: contact communication type (categorical: 'cellular','telephone')
- month: last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
- day: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')
- duration: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.
  # other attributes:
- campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)

- pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
- previous: number of contacts performed before this campaign and for this client (numeric)
- poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')
  # social and economic context attributes
- balance(numeric;average yearly balance,in euros)
- y - has the client subscribed a term deposit? (binary: 'yes','no')

***Links***

1. Base Paper: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
2. https://archive.ics.uci.edu/ml/machine-learning-databases/00222/

## Module 2,3:

policy_final.csv(1308 tuples) [Number of attributes:36]

- id numeric
- sex nominal {female,male}
- education:nominal {basic,high_school,university_degree,professional_course,illiterate}
- marital_status nominal {single,divorced,married}
- age numeric
- no_of_children nominal {0,1,2,3,4}
- no_of_employees_in_family nominal {0,1,2,3}
- net_income_of_family numeric
- no_of_dependence nominal {0,1,2,3}
- income numeric
- region nominal {inner_city,town,rural,sub_urban} ->region of stay
- car nominal {yes,no} ->whether car loan is taken or not
- mortgage nominal {yes,no} ->loan in which property or real estate used as collateral
- savings_account nominal {yes,no}
- credit_account nominal {yes,no}
- housing nominal {rent,own,inherited}
- loan nominal {yes,no} ->present loan we are working on
- job nominal {services,technician, etc..}
  dr =>debit cr=>credit
- min_dr_amt nominal {lessequal25,lessequal50,lessequal75,above75}
- max_dr_amt nominal{lessequal25,lessequal50,lessequal75,above75}

- total_dr_amt:nominal {lessequal100,lessequal200,lessequal300,lessequal400,lessequal500,above 500}
- total_dr_vocher nominal {lessequal3,lessequal6,lessequal10,above10}
- min_cr_amt nominal{lessequal25,lessequal50,lessequal75,above75}
- max_cr_amt nominal{lessequal25,lessequal50,lessequal75,above75}
- total_cr_amt nominal{lessequal100,lessequal200,lessequal300,lessequal400,lessequal500,above500}
- total_cr_vocher nominal{lessequal3,lessequal6,lessequal10,above10}
- principal_amt nominal {exceed_limit,lessequal50,above50}
- loan_adjustment nominal {adjusted,not_adjusted}
- pid numeric
- policy name: steing
- entry_age: numeric
- gender: nominal{both,female}- who are eligible
- min_annual_premium:numeric
- max_annual_premium:numeric
- risk: numeric
- profit
- call_yes:nominal{yes,no}- whether a call can be made

Generation Procedure:

Policy table(12 tuples)

| pid | name | entry_age | gender | min_annual_premium | max_annual_premium |
|---|---|---|---|---|---|
| 1 | SMART WEALTH BUILDER | 7 | both | 30000 | 300000 |
| 3 | SMART WEALTH ASSURE | 8 | both | 50000 | NULL |
| 4 | SMART PRIVILAGE | 8 | both | 600000 | NULL |
| 6 | SARAL MAHA ANAND | 18 | both | 15000 | 29000 |
| 8 | SMART POWER INSURANCE | 18 | both | 15000 | NULL |
| 11 | SMART ELITE | 18 | both | 150000 | NULL |
| 21 | SHUBH NIVESH | 18 | both | 6000 | NULL |
| 28 | SMART WOMEN ADVANTAGE | 18 | Female | 15000 | NULL |
| 33 | SMART BACHAT | 8 | both | 5100 | NULL |
| 37 | SMART GUARANTEED SAVINGS PLAN | 18 | both | 15000 | 75000 |
| 44 | FLEXI SMART PLUS | 18 | both | 50000 | NULL |
| 50 | EWEALTH INSURANCE | 18 | both | 10000 | 100000 |

Fig:1,Policy database

➔ My SQL Query : (loan:400 tuples, policy:12 tuples)

Select * from loan l,policy p where l.income-5000>p.min_annual_premium and l.age-5>f.entry_age and (l.gender=male and f.gender<>female)

➔ Then Risk and Profit are randomly generated and concatenated with the resulting table.

| risk | profit | id | sex | education | marital_status | age | no_of_children | no_of_employees_in_family | net_income_of_family | no_of_dependent |
|---|---|---|---|---|---|---|---|---|---|---|
| 22 | 59 | 1 | Male | tertiary | single | 45 | 0 | 0 | 0 | |
| 15 | 32 | 1 | Male | tertiary | single | 45 | 0 | 0 | 0 | |

| no_of_dependents | income | region | car | mortgage | savings_account | credit_account | housing | loan | job | min_dr_amount | max_d |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 27000 | inner_city | no | no | no | no | own | no | programmer | lessequal75 | above7 |
| 2 | 27000 | inner_city | no | no | no | no | own | no | programmer | lessequal75 | above7 |

| max_dr_amount | total_dr_amount | total_dr_vocher | min_cr_amount | max_cr_amount | total_cr_amount | total_cr_vocher | principal_amt | loan_a |
|---|---|---|---|---|---|---|---|---|
| above75 | lessequal100 | above10 | lessequal25 | lessequal75 | lessequal400 | lessequal3 | lessequal50 | not_ad |
| above75 | lessequal100 | above10 | lessequal25 | lessequal75 | lessequal400 | lessequal3 | lessequal50 | not_ad |

| loan_adjustment | pid | name | entry_age | gender | min_annual_premium | max_annual_premium |
|---|---|---|---|---|---|---|
| not_adjusted | 6 | SARAL MAHA ANAND | 18 | both | 15000 | 29000 |
| not_adjusted | 8 | SMART POWER INSURANCE | 18 | both | 15000 | NULL |

Fig:2,Policy_full table

➔ Then the call_yes attribute was added by joining this table with bank_full1.csv table.

In bank_full1.csv values of the attributes education and marital status for the records where at least one of these values housing ,loan, poutcome, y are yes are considered.

In policy_final.csv for records having these values for marital status and education and income>2000 , we generated the attribute value as yes for call_yes.

## Module 4:

Policy_final_call3(25 tuples) and policy_final_call4(26 tuples)

➔ Policy_final_call3

Sample of policy_final dataset with generalised class hierarchy of job attribute is used.

➔ Policy_final_call4
A new test tuple without having values for the attributes(pid to call_yes) is added to policy_final_call3 and named as Policy_final_call4.

New attributes are:

- Job:nominal {farming, business, academics, maintenance_mans, government_jobs, entrepreneur}

- The above high level of job hierarchy is derived from the low level of job hierarchy whose domain is taken from policy_final.csv which is mentioned above in module 3,4 dataset explanation
    - Low level domain {housemaid,services,management,self_employed,clerk,receptionist,**dean,principal,lecturer,**programmer,plumber,maintenance_man,farmer,lawyer,police}
    - For example dean, principle, lecturer comes under academics in higher level.

## Module 6:

Employee.csv(21 tuples- 1 testing tuple) [No of attributes:11]

- Eid :Numeric
- No_of_customers: numeric - no of customers contacted per month
- Total_calls: numeric- total no of call in a month
- Avg_call_per_customer: numeric-per month
- Avg_call_duration: numeric- per month per customer
- Avg_customer_interaction: numeric per month per customer
- Avg_e_pitch: numeric - employee pitch per month per customer
- Avg_c_pitch: numeric - customer pitch per month per call
- Avg_e_emotion: numeric- employee emotion per month per customer
- Avg_c_emotion: numeric- customer emotion per month per call
- No_of_policies: numeric – policies conformed by an employee per month

Data generation:

➔ For a total of 20 employees separate table consisting data per month have been generated.

➔ The domain values for the generation of values are set as follows:
- no of calls per customer(1-5)
- avg call duration per call(30-600 sec)
- avg customer interaction(5% to 65% of total call duration)
- e pitch(0-5)(2=>good)
- c pitch(0-5)(2=>good)
- e-emotion(1,2) :1 happy 2:sad
- c-emotion(1,2,3) :1-happy 2:sad 3:angry (very less max 10%)

Here e-emotion=3 not considered as if at all an employee shows anger on customer he is immediately removed from job.

| cid | calls_per_customer | avg_call_duration | avg_customer_interaction | e_pitch | c_pitch | e_emotion | c_emotion |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 55 | 19 | 0 | 4 | 2 | 1 |
| 2 | 4 | 38 | 13 | 3 | 1 | 2 | 2 |
| 3 | 2 | 417 | 166 | 3 | 0 | 2 | 1 |
| 4 | 5 | 59 | 17 | 2 | 3 | 1 | 1 |
| 5 | 4 | 50 | 30 | 3 | 0 | 2 | 1 |
| 6 | 3 | 55 | 24 | 3 | 5 | 1 | 2 |
| 7 | 3 | 600 | 180 | 5 | 1 | 2 | 1 |
| 8 | 4 | 250 | 87 | 1 | 3 | 2 | 1 |
| 9 | 4 | 417 | 187 | 2 | 5 | 1 | 1 |
| 10 | 3 | 428 | 214 | 4 | 4 | 1 | 1 |
| 11 | 2 | 82 | 4 | 0 | 2 | 1 | 2 |
| 12 | 2 | 228 | 148 | 4 | 0 | 2 | 2 |
| ver=1&db=telephone&table=e1&pos=0 | | 557 | 278 | 0 | 4 | 2 | 1 |

Fig 3:Employee table

- Sum or average per month have been calculated per respective attributes and the results of each employee are stored as single record in employee.csv. Thus resulting in 20 tuples for 20 employees.
- Note that for the new tuple , no_of_customers=1 and calls_per customer=1 . Therefore the averages are for 1 customer only.

## Objective:

Major objective is to find out the best technique to classify the customer and to recommend the customer with the best suitable bank products.

## Use Case Diagram:

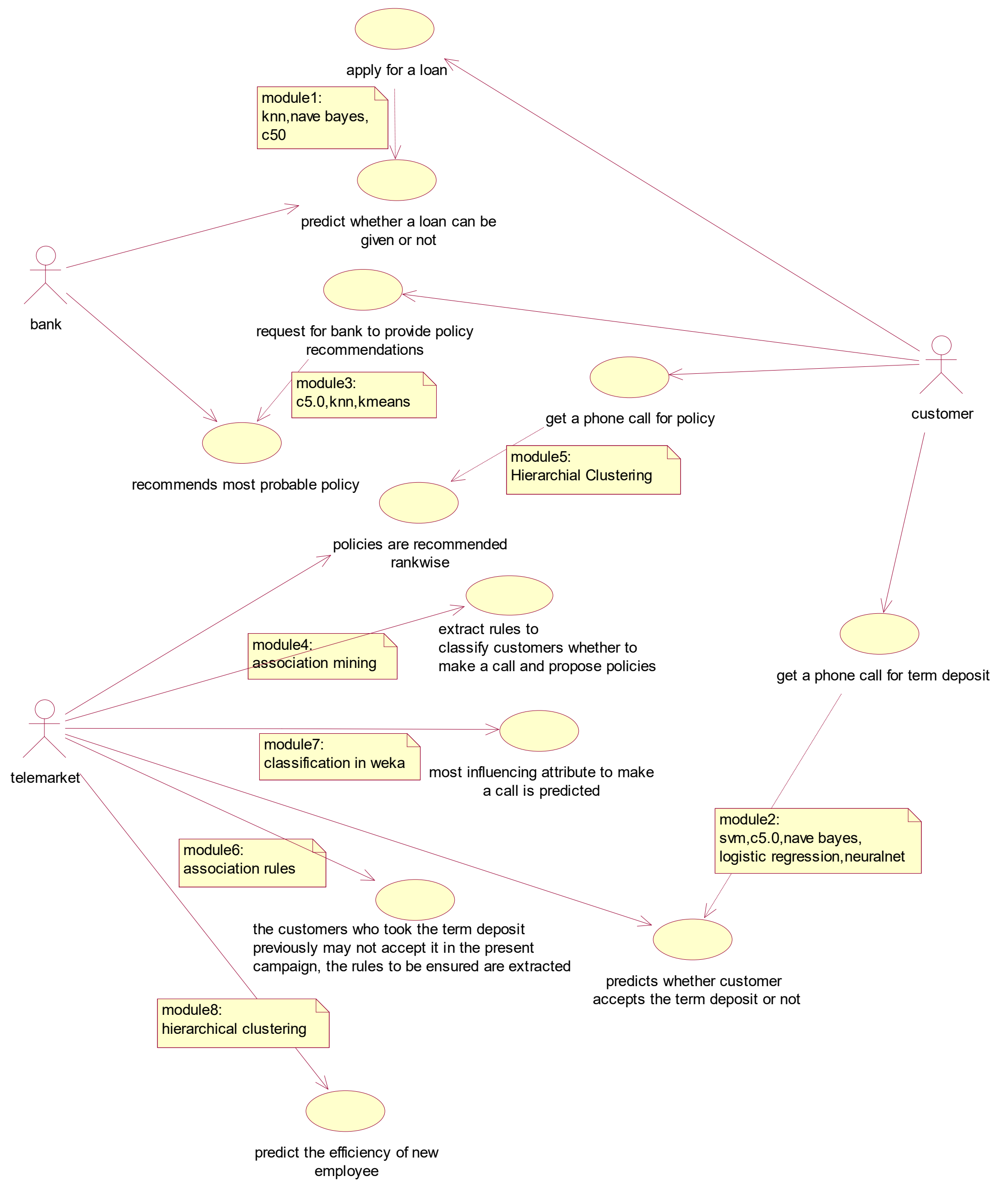



Fig 4:Use Case Diagram

# MODULES EXPLANATION

## Module 1:

**PART A:**

**Objective:**

Classification techniques are used to predict the class label for new customer where class label gives the level of performance of customer in repaying the loans.

**Code:** “loanknn.R”,”loannb.R”,”loan.R”

**DataSet:** “loannumeric.arff”,”loan_testnumeric.arff”,”loan.arff”,”loan_testm.arff”

**Major Techniques Used:** knn,Naïve bayes,c50

**Packages Included**: foreign,gmodels,dplyr,class,caret,c50,navebayes

**Output:** Test data class labels are predicted

## Module Explanation:

Step1:

Training data is extracted from “loannumeric.arff” and test data is extracted from “loan_testnumeric.arff”. Class label is a nominal attribute with 5 distinct values showing 5 levels of performance of customer ,Excellent ,Very good,good,average,bad.

Step2:

Knn is used to predict the class labels in dataset “loan_testnumeric.arff”.

```
train<-data1[,c(-1,-110)]
> test<-data2[,c(-1,-110)]
> train_label<-data1[,110]
> test_label<-data2[,110]
> pred<-knn(train,test,cl = train_label, k=10)
```

Fig 5:Knn Code

Step3:

Initially a cross table is calculated and out of it the predicted class labels are verified with previously existing class labels in “loan_testnumeric.arff” forming confusion matrix to calculate the accuracy of classification algorithm chosen.

```
>CrossTable(x=test_label,y=pred,prop.chisq=FALSE)


  Cell Contents
|-------------------------|
```

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


Total Observations in Table:  200


             | pred
  test_label |   average |       bad | excellent |      good | very_good | Row Total |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
     average |         6 |        10 |         4 |         3 |        11 |        34 |
             |     0.176 |     0.294 |     0.118 |     0.088 |     0.324 |     0.170 |
             |     0.273 |     0.159 |     0.138 |     0.188 |     0.157 |           |
             |     0.030 |     0.050 |     0.020 |     0.015 |     0.055 |           |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
         bad |         3 |        26 |         4 |         3 |        12 |        48 |
             |     0.062 |     0.542 |     0.083 |     0.062 |     0.250 |     0.240 |
             |     0.136 |     0.413 |     0.138 |     0.188 |     0.171 |           |
             |     0.015 |     0.130 |     0.020 |     0.015 |     0.060 |           |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
   excellent |         3 |        10 |        13 |         2 |        11 |        39 |
             |     0.077 |     0.256 |     0.333 |     0.051 |     0.282 |     0.195 |
             |     0.136 |     0.159 |     0.448 |     0.125 |     0.157 |           |
             |     0.015 |     0.050 |     0.065 |     0.010 |     0.055 |           |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
        good |         7 |         7 |         4 |         4 |         6 |        28 |
             |     0.250 |     0.250 |     0.143 |     0.143 |     0.214 |     0.140 |
             |     0.318 |     0.111 |     0.138 |     0.250 |     0.086 |           |
             |     0.035 |     0.035 |     0.020 |     0.020 |     0.030 |           |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
   very_good |         3 |        10 |         4 |         4 |        30 |        51 |
             |     0.059 |     0.196 |     0.078 |     0.078 |     0.588 |     0.255 |
             |     0.136 |     0.159 |     0.138 |     0.250 |     0.429 |           |
             |     0.015 |     0.050 |     0.020 |     0.020 |     0.150 |           |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
Column Total |        22 |        63 |        29 |        16 |        70 |       200 |
             |     0.110 |     0.315 |     0.145 |     0.080 |     0.350 |           |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|
```

Fig 6:Cross Table

Step4:

From the following statistics it can be clearly observed that knn is predicting the class label with an accuracy of 0.395.

```
a<-table(pred,test_label)
> confusionMatrix(a)

Confusion Matrix and Statistics

           test_label
pred        average bad excellent good very_good
  average         6   3         3    7         3
  bad            10  26        10    7        10
  excellent       4   4        13    4         4
  good            3   3         2    4         4
  very_good      11  12        11    6        30

Overall Statistics

               Accuracy : 0.395
                 95% CI : (0.3268, 0.4664)
    No Information Rate : 0.255
    P-Value [Acc > NIR] : 9.821e-06

                  Kappa : 0.2213
 Mcnemar's Test P-Value : 0.04328

Statistics by Class:

                     Class: average Class: bad Class: excellent Class: good
Sensitivity                  0.1765     0.5417           0.3333      0.1429
Specificity                  0.9036     0.7566           0.9006      0.9302
Pos Pred Value               0.2727     0.4127           0.4483      0.2500
Neg Pred Value               0.8427     0.8394           0.8480      0.8696
Prevalence                   0.1700     0.2400           0.1950      0.1400
Detection Rate               0.0300     0.1300           0.0650      0.0200
Detection Prevalence         0.1100     0.3150           0.1450      0.0800
```

Fig 7:Confusion Matrix of KNN

Step5:

Naive bayes algorithm is also used to predict the class label and providing the training data and testing data is same as the knn and from the confusion matrix an accuracy of 0.14 is calculated.

```
> confusionMatrix(a)
Confusion Matrix and Statistics

           test_label
```

```
p           average bad excellent good very_good
  average         0   0         1    0         0
  bad             0   0         0    0         0
  excellent       0   0         0    0         0
  good           34  48        38   28        51
  very_good       0   0         0    0         0

Overall Statistics

              Accuracy : 0.14
                95% CI : (0.0951, 0.1959)
    No Information Rate : 0.255
    P-Value [Acc > NIR] : 1

                 Kappa : -2e-04
 Mcnemar's Test P-Value : NA

 Statistics by Class:

                     Class: average Class: bad Class: excellent Class: good
Sensitivity                  0.0000       0.00            0.000    1.000000
Specificity                  0.9940       1.00            1.000    0.005814
Pos Pred Value               0.0000        NaN              NaN    0.140704
Neg Pred Value               0.8291       0.76            0.805    1.000000
Prevalence                   0.1700       0.24            0.195    0.140000
Detection Rate               0.0000       0.00            0.000    0.140000
Detection Prevalence         0.0050       0.00            0.000    0.995000
Balanced Accuracy            0.4970       0.50            0.500    0.502907
                     Class: very_good
Sensitivity                     0.000
Specificity                     1.000
Pos Pred Value                    NaN
Neg Pred Value                  0.745
Prevalence                      0.255
Detection Rate                  0.000
Detection Prevalence            0.000
Balanced Accuracy               0.500
```

Fig 8:Confusion Matrix of Naïve Bayes

Step6:

C50 algorithm is the next algorithm which is being followed to predict the class label on the data set loan.arff and loan_testm.arff

```
Evaluation on training data (400 cases):
```

```
Trial        Decision Tree
-----       ----------------
            Size      Errors

   0         123    5( 1.3%)
   1          80   85(21.3%)
   2          96   70(17.5%)
   3          86  106(26.5%)
   4          95   80(20.0%)
   5          99   71(17.8%)
   6         108   55(13.8%)
   7          90   90(22.5%)
   8          93   91(22.8%)
   9         110   47(11.8%)
boost              0( 0.0%)   <<


             (a)   (b)   (c)   (d)   (e)    <-classified as
            ----  ----  ----  ----  ----
              72                            (a): class average
                    93                      (b): class bad
                          75                (c): class excellent
                                58          (d): class good
                                     102    (e): class very_good


        Attribute usage:

        100.00%        id
        100.00%        job
         26.25%        marital_status
         23.00%        sex
         22.50%        age
         22.50%        no_of_dependents
         20.25%        net_income_of_family
         18.50%        housing
         18.25%        max_cr_amt
         18.25%        principal_amt
         17.00%        income
         16.50%        region
         14.50%        total_cr_vocher
```

```
13.75%            education
13.75%            loan
13.00%            savings_account
12.50%            total_cr_amt
12.00%            no_of_children
11.00%            mortgage
10.00%            loan_adjustment
 8.75%no_of_employees_in_family
 8.75%total_dr_vocher
 8.25%credit_account
 7.75%car
 4.75%min_cr_amt
 4.50%min_dr_amt
 4.25%total_dr_amt
 3.75%max_dr_amt


Time: 0.1 secs
```

Fig 9:Trained Model output

```
> sum(p==t2)/length(p)
[1] 0.98
```

Fig10:Test model output

**PART B**

**Objective:** Base paper implementation to predict if a customer accepts a policy(class_label) on receiving a call.

**Code**: market.R

**Data Set:** bank_full1.csv

**Major Techniques Used:** C5.0, Logistic Regression, Bayesian Network, Support Vector Machine, Neural Networks

**Packages Included:** c50,navebayes,Amelia,caret,neuralnet,e1071

**Output:** We were able to predict the class_label using SVM which gave maximum accuracy.

## Module Explanation:

Step1: Logistic Regression was applied on the dataset the results are as follows

LOGISTIC REGRESSION

```
> class(t)
[1] "table"
> accuracy<-(t[1,1]+t[2,2])/(t[1,1]+t[1,2]+t[2,1]+t[2,2])
> accuracy
[1] 0.9018281179
> a
   No   Yes
14412   959
```

Fig 11:Output for Logistic Regression

Step2:

C5.0

```
> confusionMatrix(table(p,test_label))
Confusion Matrix and Statistics

     test_label
p        no   yes
  no  12938   913
  yes   513   847

               Accuracy : 0.9062521
                 95% CI : (0.9015091, 0.910839)
    No Information Rate : 0.8842943
    P-Value [Acc > NIR] : < 0.00000000000000022204
```

Fig 12:Output for C5.0

Step 3:

Neural Network

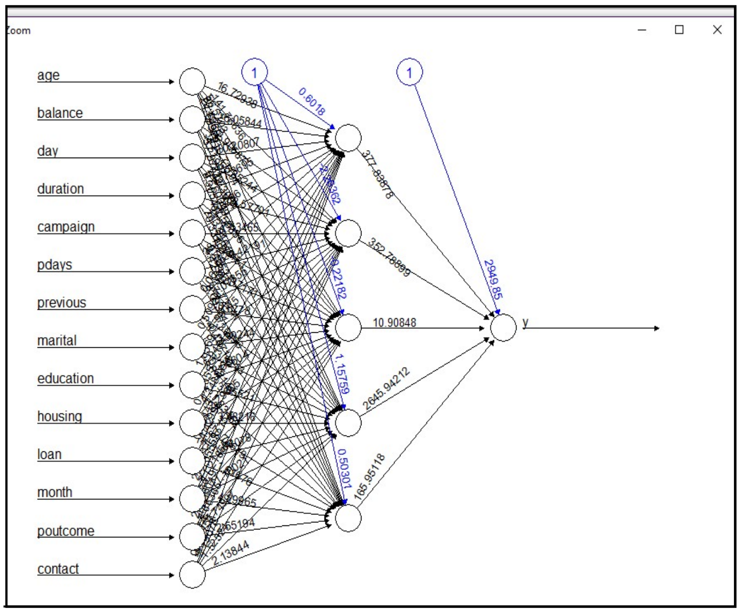



Fig 13:Neural Network

Step 4:

SVM

```
> test.svm<-predict(train.svm,test)
> confusionMatrix(table(predict=test.svm,truth=data[r,17]))
Confusion Matrix and Statistics

       truth
predict    no   yes
    no  12922  1029
    yes   499   761

               Accuracy : 0.8995464
                 95% CI : (0.8946597, 0.9042796)
    No Information Rate : 0.882322
    P-Value [Acc > NIR] : 0.000000000008762049
```

Fig 14:Output for Support vector Machine

Step 5:

Naïve bayes

```
> confusionMatrix(table(p,test_label))
Confusion Matrix and Statistics

    test_label
p        no   yes
  no  12436   826
  yes  1023   926

               Accuracy : 0.8784432
                 95% CI : (0.8731449, 0.8835963)
    No Information Rate : 0.8848202
    P-Value [Acc > NIR] : 0.9930541
```

Fig 15:Output for Naïve bayes

## Module 2:

**Objective:**

To recommend the bank policies to a customer along with profit percentage and risk percentage.

**Code** : policy_clustering_recommendations_including_job.R

**DataSet :** policy_final_binary_correct.csv, policy_final.csv,policy_test.csv

**Major Techniques Used:** knn, C5.0, K means Clustering.

**Packages Included:** carets, class, foreign, gmodels, dplyr,stats, cluster, factoextra,

**Output :** List of policies along with their profit percentage and risk percentage feasible for the particular customer.

## Module Explanation:

Step1:

The dataset containing the policies taken by the customers , profit factor, Risk factor along with some other customer details is prepared , named as "policy_final.csv"

Step2:

k-means clustering technique is being applied to the dataset considering attributes job(nominal),income(numeric) to calculate dissimilarity among the tuples .

But k-means can't be applied for nominal attribute like job so all the nominal attributes in the data set have to be converted into binary variables except policy_name because it is going to be the class label after applying k-means clustering algorithm.

Step3:

Weka software can be used to convert nominal values to binary values .

- Choose file ->"policy_final.csv"
- NominalToBinary is selected by the following the below path
- Weka -> filters -> unsupervised -> attribute -> NominalToBinary
- NominalToBinary
    - attributeIndices = 30
    - InvertSelection = True
- It is saved as "policy_final_binary_correct.csv"

Step4:

By using Elbow method, a plot is visualised to calculate the optimal number of clusters

From the following plot optimal number of clusters =3

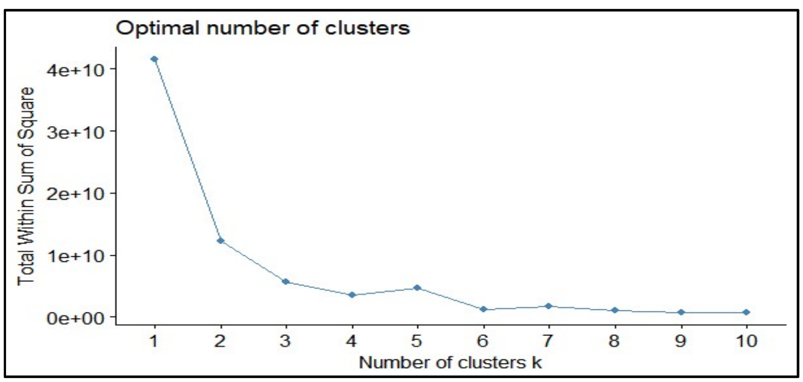



Fig 16: Elbow Method

Step 5:

K means Clustering algorithm is now applied with k value as 3

Step 6:

The total profit value and the total risk value for each policy in each cluster is calculated

| |
|---|
| profit_table |

```
      SARAL  MAHA  ANAND  SMART  POWER  INSURANCE  SHUBH  NIVESH
SMART BACHAT
cluster1           4800              4429       4840       4931
cluster2              0                 0      5727       6059
cluster3           4317              4316       4623       5006
      SMART  GUARANTEED  SAVINGS  PLAN  EWEALTH  INSURANCE  SMART
WOMEN ADVANTAGE
cluster1                    5323          5027             2332
cluster2                       0        3203                0
cluster3                    3971          4882             2670
> risk_table
      SARAL  MAHA  ANAND  SMART  POWER  INSURANCE  SHUBH  NIVESH
SMART BACHAT
cluster1           1628              1620       1510       1558
cluster2              0                 0      1986       2186
cluster3           1409              1479       1530       1484
      SMART  GUARANTEED  SAVINGS  PLAN  EWEALTH  INSURANCE  SMART
WOMEN ADVANTAGE
cluster1                    1494          1534              803
cluster2                       0        1049                0
cluster3                    1288          1583              917
```

Fig 17:Clustering based on Policy

Now the average profit percentage and the average risk percentage of a policy for a particular cluster is obtained .

As it can be seen in the code those percentages are stored as matrices in "profit_percentage" and "risk_percentage".

```
profit_percentage
      SARAL  MAHA  ANAND  SMART  POWER  INSURANCE  SHUBH  NIVESH
SMART BACHAT
cluster1        58.53659           54.01220    59.02439    60.13415
cluster2             NaN                NaN    59.04124    54.58559
cluster3        59.13699           59.12329    55.03571    59.59524
      SMART  GUARANTEED  SAVINGS  PLAN  EWEALTH  INSURANCE  SMART
WOMEN ADVANTAGE
cluster1                 64.91463        61.30488            58.30000
cluster2                      NaN       61.59615                 NaN
cluster3                 54.39726        58.11905            62.09302
> risk_percentage
      SARAL  MAHA  ANAND  SMART  POWER  INSURANCE  SHUBH  NIVESH
SMART BACHAT
cluster1        19.85366           19.75610    18.41463    19.00000
cluster2             NaN                NaN    20.47423    19.69369
cluster3        19.30137           20.26027    18.21429    17.66667
      SMART  GUARANTEED  SAVINGS  PLAN  EWEALTH  INSURANCE  SMART
WOMEN ADVANTAGE
```

| | | | |
|---|---|---|---|
| cluster1 | 18.21951 | 18.70732 | 20.07500 |
| cluster2 | NaN | 20.17308 | NaN |
| cluster3 | 17.64384 | 18.84524 | 21.32558 |

Fig 18:policy vs profit and risk percentage

Step 7:

A new tuple will now be added to the data set and k-means clustering have to be applied

Step 8:

The cluster in which the new tuple falls is found out.

Step 9:

The Policies of that cluster are displayed along with the profit percentage and risk percentage.

```
print("Average profit percentage")
[1] "Average profit percentage"
> sort(profit_percentage[cluster_new,],decreasing=TRUE)
SMART GUARANTEED SAVINGS PLAN            EWEALTH INSURANCE
                 64.91463                   61.30488
              SMART BACHAT                 SHUBH NIVESH
                 60.13415                   59.02439
           SARAL MAHA ANAND         SMART WOMEN ADVANTAGE
                 58.53659                   58.30000
       SMART POWER INSURANCE
                 54.01220
> print("Average risk percentage")
[1] "Average risk percentage"
> sort(risk_percentage[cluster_new,],decreasing=FALSE)
SMART GUARANTEED SAVINGS PLAN                SHUBH NIVESH
                 18.21951                   18.41463
          EWEALTH INSURANCE                 SMART BACHAT
                 18.70732                   19.00000
       SMART POWER INSURANCE              SARAL MAHA ANAND
                 19.75610                   19.85366
       SMART WOMEN ADVANTAGE
                 20.07500
```

Fig 19:Recommending policy along with profit and risk

*Classification steps:*

Considering the above mentioned data set policy_final_binary_correct.csv classification techniques such as C5.0 and knn are applied on it. The context of output which we get from classification and clustering are slightly different. Through classification we recommend the customer with the policy which was taken by similar customer previously.(similar customers means the dissimilarity value between those customers is minimum). Whereas in clustering a list of ordered policies with the profit and risk percentage for the policy in the cluster to which the customer belongs is displayed.

C5.0: on policy_final.csv

Train model:

```
 (a)   (b)   (c)   (d)   (e)    <-classified as
----  ----  ----  ----  ----
  72                           (a): class average
        93                     (b): class bad
              75               (c): class excellent
                    58         (d): class good
                         102   (e): class very_good


Attribute usage:

100.00%     id
100.00%     job
 26.25%     marital_status
 23.00%     sex
 22.50%     age
 22.50%     no_of_dependents
 20.25%     net_income_of_family
 18.50%     housing
 18.25%     max_cr_amt
 18.25%     principal_amt
 17.00%     income
 16.50%     region
 14.50%     total_cr_vocher
 13.75%     education
 13.75%     loan
 13.00%     savings_account
 12.50%     total_cr_amt
 12.00%     no_of_children
 11.00%     mortgage
 10.00%     loan_adjustment
  8.75%no_of_employees_in_family
  8.75%total_dr_vocher
  8.25%credit_account
```

```
        7.75%car
        4.75%min_cr_amt
        4.50%min_dr_amt
        4.25%total_dr_amt
        3.75%max_dr_amt


Time: 0.1 secs
```

Fig 20:C5.0 Train model

Test output:

```
p         ?
  average   34
  bad       47
  excellent 39
  good      27
  very_good 53
```

Fig 21:C5.0 Test Output

Knn:

```
confusionMatrix(a)
Confusion Matrix and Statistics

                              test_label
pred                           'EWEALTH INSURANCE' 'SARAL MAHA ANAND'
  'EWEALTH INSURANCE'                           54                 23
  'SARAL MAHA ANAND'                            22                 28
  'SHUBH NIVESH'                                50                 25
  'SMART BACHAT'                                29                 19
  'SMART GUARANTEED SAVINGS PLAN'               21                 28
  'SMART POWER INSURANCE'                       29                 23
  'SMART WOMEN ADVANTAGE'                       14                  9
                              test_label
pred                           'SHUBH NIVESH' 'SMART BACHAT'
  'EWEALTH INSURANCE'                      43             43
  'SARAL MAHA ANAND'                       20             29
  'SHUBH NIVESH'                           74             72
  'SMART BACHAT'                           73             73
  'SMART GUARANTEED SAVINGS PLAN'          19             26
  'SMART POWER INSURANCE'                  20             22
  'SMART WOMEN ADVANTAGE'                  14             12
                              test_label
pred                           'SMART GUARANTEED SAVINGS PLAN'
  'EWEALTH INSURANCE'                                       22
  'SARAL MAHA ANAND'                                        22
```

'SHUBH NIVESH' 22
'SMART BACHAT' 22
'SMART GUARANTEED SAVINGS PLAN' 24
'SMART POWER INSURANCE' 23
'SMART WOMEN ADVANTAGE' 20

| | test_label | |
|---|---|---|
| pred | 'SMART POWER INSURANCE' | 'SMART WOMEN ADVANTAGE' |
| 'EWEALTH INSURANCE' | 29 | 11 |
| 'SARAL MAHA ANAND' | 25 | 13 |
| 'SHUBH NIVESH' | 27 | 15 |
| 'SMART BACHAT' | 24 | 6 |
| 'SMART GUARANTEED SAVINGS PLAN' | 28 | 12 |
| 'SMART POWER INSURANCE' | 17 | 14 |
| 'SMART WOMEN ADVANTAGE' | 5 | 12 |

Overall Statistics

Accuracy : 0.2158
95% CI : (0.1937, 0.2391)
No Information Rate : 0.2119
P-Value [Acc > NIR] : 0.3781

Kappa : 0.0675
Mcnemar's Test P-Value : 0.6299

Statistics by Class:

| | Class: 'EWEALTH INSURANCE' | Class: 'SARAL MAHA ANAND' |
|---|---|---|
| Sensitivity | 0.24658 | 0.18065 |
| Specificity | 0.84283 | 0.88628 |
| Pos Pred Value | 0.24000 | 0.17610 |
| Neg Pred Value | 0.84750 | 0.88937 |
| Prevalence | 0.16756 | 0.11859 |
| Detection Rate | 0.04132 | 0.02142 |
| Detection Prevalence | 0.17215 | 0.12165 |
| Balanced Accuracy | 0.54470 | 0.53346 |

| | Class: 'SHUBH NIVESH' | Class: 'SMART BACHAT' |
|---|---|---|
| Sensitivity | 0.28137 | 0.26354 |
| Specificity | 0.79789 | 0.83204 |
| Pos Pred Value | 0.25965 | 0.29675 |
| Neg Pred Value | 0.81507 | 0.80773 |
| Prevalence | 0.20122 | 0.21194 |
| Detection Rate | 0.05662 | 0.05585 |
| Detection Prevalence | 0.21806 | 0.18822 |
| Balanced Accuracy | 0.53963 | 0.54779 |

| | Class: 'SMART GUARANTEED SAVINGS PLAN' |
|---|---|
| Sensitivity | 0.15484 |
| Specificity | 0.88368 |

```
Pos Pred Value                    0.15190
Neg Pred Value                    0.88599
Prevalence                        0.11859
Detection Rate                    0.01836
Detection Prevalence              0.12089
Balanced Accuracy                 0.51926
                Class: 'SMART POWER INSURANCE' Class: 'SMART WOMEN
ADVANTAGE'
Sensitivity                    0.10968              0.144578
Specificity                    0.88628              0.939542
Pos Pred Value                 0.11486              0.139535
Neg Pred Value                 0.88093              0.941851
Prevalence                     0.11859              0.063504
Detection Rate                 0.01301              0.009181
Detection Prevalence           0.11324              0.065800
Balanced Accuracy              0.49798              0.542060
```

Fig 22:Confusion Matrix of KNN

```
library(caret)
library(stats)
library(cluster)
library(factoextra)
data1<-read.csv(file.choose())
#policy_final_binary_correct.csv

set.seed(123)
k.max <- 10
fviz_nbclust(data1[,c(16,28:59)],kmeans,method="wss")
#number of clusters=3

m3<-kmeans(data1[,c(16,28:59)],3,nstart=3,algorithm="Lloyd")
table(m3$cluster, data1[,101])
cluster_col <- m3$cluster
cluster_col
data1$cluster_col <- cluster_col
data1
policy_code<-c("SARAL MAHA ANAND","SMART POWER INSURANCE","SHUBH
NIVESH",
          "SMART BACHAT","SMART GUARANTEED SAVINGS PLAN","EWEALTH
INSURANCE",
          "SMART WOMEN ADVANTAGE")

policy_id <-c(6,8,21,33,37,50,28)

policy_table <- matrix(rep(0,times=21),nrow=3,byrow=TRUE)
profit_table <- matrix(rep(0,times=21),nrow=3,byrow=TRUE)
```

```
risk_table <- matrix(rep(0,times=21),nrow=3,byrow=TRUE)
profit_percentage <- matrix(rep(0,times=21),nrow=3,byrow=TRUE)
risk_percentage <- matrix(rep(0,times=21),nrow=3,byrow=TRUE)

policy_table
profit_table
risk_table
profit_percentage
risk_percentage

colnames(policy_table) <- policy_code
rownames(policy_table) <- c("cluster1","cluster2","cluster3")
colnames(profit_table) <- policy_code
rownames(profit_table) <- c("cluster1","cluster2","cluster3")
colnames(risk_table) <- policy_code
rownames(risk_table) <- c("cluster1","cluster2","cluster3")
colnames(profit_percentage) <- policy_code
rownames(profit_percentage) <- c("cluster1","cluster2","cluster3")
colnames(risk_percentage) <- policy_code
rownames(risk_percentage) <- c("cluster1","cluster2","cluster3")


i <- 1
j <- 1
k <- 1
for(a in 0:2)
{
  j <- 1
  for(b in 0:6)
  {
    k <- 1
    for(c in 0:1305)
    {
      if((data1[k,100]==policy_id[j])&&(data1[k,109]==i))
      {
        policy_table[i,j] <- policy_table[i,j]+1

      }
      k <- k+1
    }
    j <- j+1
  }
  i <- i+1
}

i <- 1
j <- 1
k <- 1
for(a in 0:2)
```

```
{
  j <- 1
  for(b in 0:6)
  {
    k <- 1
    for(c in 0:1306)
    {
      if((data1[k,100]==policy_id[j])&&(data1[k,109]==i))
      {
        profit_table[i,j] <- profit_table[i,j]+data1[k,107]

      }
      k <- k+1
    }
    j <- j+1
  }
  i <- i+1
}

i <- 1
j <- 1
for(a in 0:2)
{
  j <- 1
  for(b in 0:6)
  {
    profit_percentage[i,j]=profit_table[i,j]/policy_table[i,j]
    j <- j+1
  }
  i <- i+1
}

i <- 1
j <- 1
k <- 1
for(a in 0:2)
{
  j <- 1
  for(b in 0:6)
  {
    k <- 1
    for(c in 0:1306)
    {
      if((data1[k,100]==policy_id[j])&&(data1[k,109]==i))
      {
        risk_table[i,j] <- risk_table[i,j]+data1[k,106]

      }
      k <- k+1
```

```
    }
    j <- j+1
  }
  i <- i+1
}


i <- 1
j <- 1
for(a in 0:2)
{
  j <- 1
  for(b in 0:6)
  {
    risk_percentage[i,j]=risk_table[i,j]/policy_table[i,j]
    j <- j+1
  }
  i <- i+1
}

data2<-read.csv(file.choose())
#policy_final.csv
m4<-kmeans(data2[,c(16,28:59)],3,nstart=3,algorithm="Lloyd")
table(m4$cluster, data2[,101])
cluster_col <- m4$cluster
cluster_new <- tail(cluster_col,n=1)
print("Average profit percentage")
sort(profit_percentage[cluster_new,],decreasing=TRUE)
print("Average risk percentage")
sort(risk_percentage[cluster_new,],decreasing=FALSE)
```

Fig 23: Code code for recommending policies along with profit and risk percentage

## Module 3

**Objective:** To decide which type of customers to be chosen for telemarketing as calling every customer who previously approached bank would not be fruitful

**Code:** Policy_template.R

**Data Set:** policy_final.csv

**Major Techniques Used:** Association mining using template matching

**Packages Included:** apriori

**Output:** Certain attributes and their values which define whether to call a customer or not.

## Module Explanation:

Step1:

All numeric attributes are converted into factor variables and stored back into data set.

Step 2:

Apriori with template matching is used to fix the class label as yes and association rules are generated.

Step 3:

These rules are stored in an excel file. It is observed that although the confidence value lies about 0.5 lift value is very high , thus making the rules interesting.

| 1 | rules | support | confidenc | lift | count |
|---|---|---|---|---|---|
| 2 | {sex=Female,savings_account=no,total_cr_vocher=lessequal10} => {call_yes=yes} | 0.037462 | 0.576471 | 5.200162 | 49 |
| 3 | {credit_account=yes,min_dr_amt=lessequal25,total_dr_vocher=above10} => {call_yes=yes} | 0.030581 | 0.512821 | 4.625995 | 40 |
| 4 | {min_dr_amt=lessequal25,total_dr_vocher=above10,entry_age=18} => {call_yes=yes} | 0.030581 | 0.5 | 4.510345 | 40 |
| 5 | {sex=Female,savings_account=no,total_cr_vocher=lessequal10,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.03211 | 0.65625 | 5.919828 | 42 |
| 6 | {sex=Female,savings_account=no,total_cr_vocher=lessequal10,entry_age=18} => {call_yes=yes} | 0.03211 | 0.617647 | 5.571602 | 42 |
| 7 | {sex=Female,savings_account=no,total_cr_vocher=lessequal10,gender=both} => {call_yes=yes} | 0.03211 | 0.56 | 5.051586 | 42 |
| 8 | {mortgage=yes,savings_account=no,credit_account=yes,max_cr_amt=lessequal25} => {call_yes=yes} | 0.030581 | 0.547945 | 4.942844 | 40 |
| 9 | {mortgage=yes,savings_account=no,housing=inherited,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.031346 | 0.518987 | 4.681624 | 41 |
| 10 | {mortgage=yes,savings_account=no,credit_account=yes,housing=inherited} => {call_yes=yes} | 0.031346 | 0.577465 | 5.209131 | 41 |
| 11 | {car=yes,savings_account=no,housing=inherited,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.03211 | 0.5 | 4.510345 | 42 |
| 12 | {car=yes,mortgage=yes,housing=inherited,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.031346 | 0.546667 | 4.93131 | 41 |
| 13 | {mortgage=yes,savings_account=no,credit_account=yes,housing=inherited,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.031346 | 0.640625 | 5.778879 | 41 |
| 14 | {sex=Female,car=yes,savings_account=no,housing=inherited,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.03211 | 0.583333 | 5.262069 | 42 |
| 15 | {car=yes,mortgage=yes,credit_account=yes,housing=inherited,loan_adjustment=not_adjusted} => {call_yes=yes} | 0.031346 | 0.561644 | 5.066415 | 41 |
| 16 | | | | | |

Fig 24:Association Rules using Template matching

## Module 4 :

**Objective**:

To recommend the bank policies to a customer through telemarketing by banks.

**Code :** recommend_phone_calls.R

**DataSet :** policy_final_call3.csv, policy_final_call4.csv

**Major Techniques Used**: Hierarchical Clustering.

**Packages Included**: carets, stats, cluster

**Output :** Array of Policies in an Sorted Order (policy_sort) which can be recommended for the customer is obtained.

## Module Explanation:

Step1:

Data about the customers who accepted the bank policies through telemarketing in the past has been prepared. Which is saved as “policy_final_call3.csv”.

Step2:

Dissimalirity matrix of the dataset is prepared by considering the attributes job(nominal) and income(numeric).

Step3:

All the tupples in the dataset fall into 4 Clusters by applying hierarchical clustering considering the calculated dissimilarity matrix and the policy_name as class label.

Here number of clusters is set to 4 by observing the plot Cluster Dendogram.

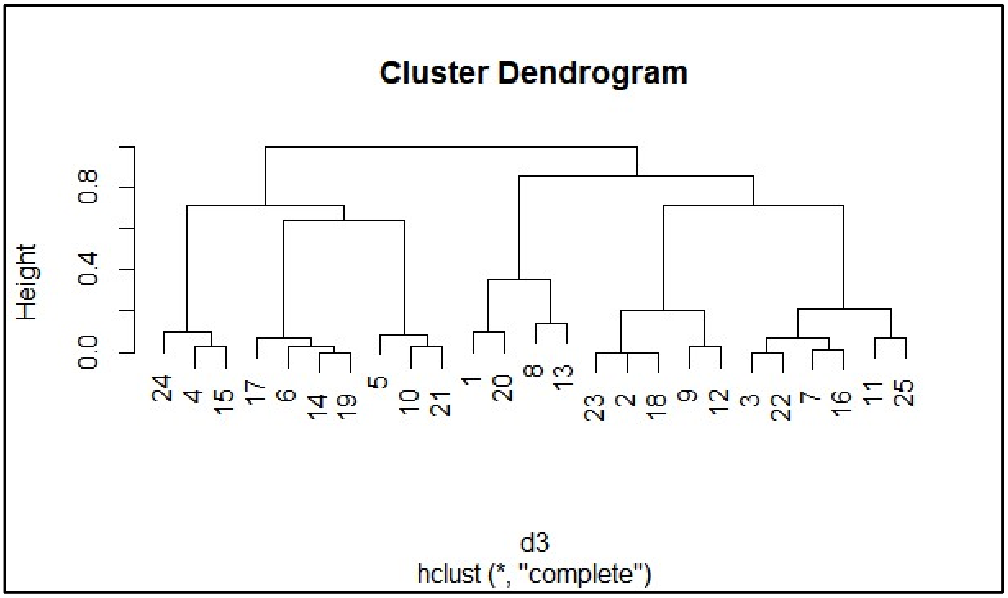



Fig 25:Cluster Dendrogram

Step4:

Now Add a new tupple of customer information to whom the Customer Care have to make a call. That is added to “policy_final_call3.csv” which is named as “policy_final_call4.csv”.

Step5:

Calculation of Dissimilarity Matrix and Clustering the Data set is done same as Step2 and Step3

Now it is observed that new tupple falls in one of the 4 Clusters.

Step6:

It can be clearly seen in the given code that calculations are made to find the cluster in which the new Tupple falls.

Step7:

The Policies in that Particular Cluster are taken into an Array along with their Frequencies which is named as “policy”.

Step 8:

“Policy” Array is Sorted in Descending Order of Frequency and the Policies are displayed according to their Ranks. Sorted Array is stored in “policy_sort”.

```
policy
          EWEALTH INSURANCE             SARAL MAHA ANAND
                        0                        1
              SHUBH NIVESH                SMART BACHAT
                        4                        2
SMART GUARANTEED SAVINGS PLAN        SMART POWER INSURANCE
                        1                        2
       SMART WOMEN ADVANTAGE
                        0
```

Fig 26:Policy array

Step 9:

“policy_sort” is viewed by the Customer Care employees and they can explain the customers about their policies according to the order obtained which is the best way to improve telemarketing.

The output which we got is

```
policy_sort
              SHUBH NIVESH                SMART BACHAT
                        4                        2
       SMART POWER INSURANCE            SARAL MAHA ANAND
                        2                        1
SMART GUARANTEED SAVINGS PLAN            EWEALTH INSURANCE
                        1                        0
       SMART WOMEN ADVANTAGE
                        0
names(policy_sort)
[1] "SHUBH NIVESH"                  "SMART BACHAT"
[3] "SMART POWER INSURANCE"         "SARAL MAHA ANAND"
[5] "SMART GUARANTEED SAVINGS PLAN" "EWEALTH INSURANCE"
[7] "SMART WOMEN ADVANTAGE"
```

Fig 27:Policy Sorted array

Code:

```
library(caret)
library(stats)
library(cluster)
data3<-read.csv(file.choose())
#policy_final_call3.csv
d3<-daisy(data3[,c(10,18)],metric="gower" ,stand = FALSE)
cluster3<-hclust(d3,method="complete")
```

```
plot(cluster3)
table(cutree(cluster3,4),data3[,30])
table1 <- table(cutree(cluster3,4),data3[,30])


data4<-read.csv(file.choose())
#policy_final_call4.csv
d4<-daisy(data4[,c(10,18)],metric="gower" ,stand = FALSE)
cluster4<-hclust(d4,method="complete")
plot(cluster4)
table(cutree(cluster4,4),data4[,30])
table2 <- table(cutree(cluster4,4),data4[,30])
table3 <- table2-table1


x <- -1
y <- -1
i <- 1
j <- 1
c<- 0
for(a in 0:3)
{
  for(b in 0:7)
  {
    c <- table2[i,j]-table1[i,j]
    j <- j+1
    if(c==1)
    {
      x <- i
      y <- j-1
    }
  }
  i <- i+1
  j <- 1
}
x
y
policy <- table1[x,c(-8)]
policy
policy_sort <- (sort(policy,decreasing = TRUE))
policy_sort
names(policy_sort)
```

Fig 28: Code for recommending Policy

## Module 5:

**PART A:**

**Objective:**

Some of the customers who took the term deposit in the previous telemarketing campaign might not take the term deposit in the present Campaign. Rules are extracted considering this situation to ensure the reasons behind this.

**Code:** telephone_template.R

**DataSet:** bank_full1.csv

**Major Techniques Used:** Association rules from apriori

**Packages Included:** arules

**Output:** Rules obtained after template matching

## Module Explanation:

Step1:

To apply apriori algorithm and extract association rules we need nominal attributes. So we initially convert the numerical attributes in the dataset “bank_full1.csv” to factors.

Step2:

Template matching is done by using Apriori .Antecedent is fixed in such a way that association rules are extracted only from the tupples of the customer who took the term deposit in the previous campaign but not in the present campaign.

```
ap<apriori(data,parameter=list(supp=0.0036,conf=0.001),appearance=list(lhs=c("p
outcome=success", "y=no"),default="rhs"),control=list(verbose=F))
```

Fig 29:Function for Template matching

Step 3:

The rules obtained by template matching are

| rules | support | confidenc | lift | count |
|---|---|---|---|---|
| {poutcome=success,y=no} => {previous=1} | 0.003804 | 0.322702 | 5.263227 | 172 |
| {poutcome=success,y=no} => {marital=single} | 0.004092 | 0.347092 | 1.226925 | 185 |
| {poutcome=success,y=no} => {education=tertiary} | 0.004711 | 0.399625 | 1.358352 | 213 |
| {poutcome=success,y=no} => {campaign=1} | 0.006193 | 0.525328 | 1.353774 | 280 |
| {poutcome=success,y=no} => {housing=no} | 0.006879 | 0.58349 | 1.313687 | 311 |
| {poutcome=success,y=no} => {education=secondary} | 0.005353 | 0.454034 | 0.884722 | 242 |
| {poutcome=success,y=no} => {housing=yes} | 0.00491 | 0.41651 | 0.749337 | 222 |
| {poutcome=success,y=no} => {marital=married} | 0.006392 | 0.542214 | 0.900788 | 289 |
| {poutcome=success,y=no} => {contact=cellular} | 0.010727 | 0.909944 | 1.404796 | 485 |

Fig 30:Rules output

These rules are stored in “output.csv”

**PART B:**

**Objective:**

For the dataset bank_full1.csv various interesting questions can be answered, one of that is being addressed here.

Which attribute is deciding factor for a person to accept term deposit?

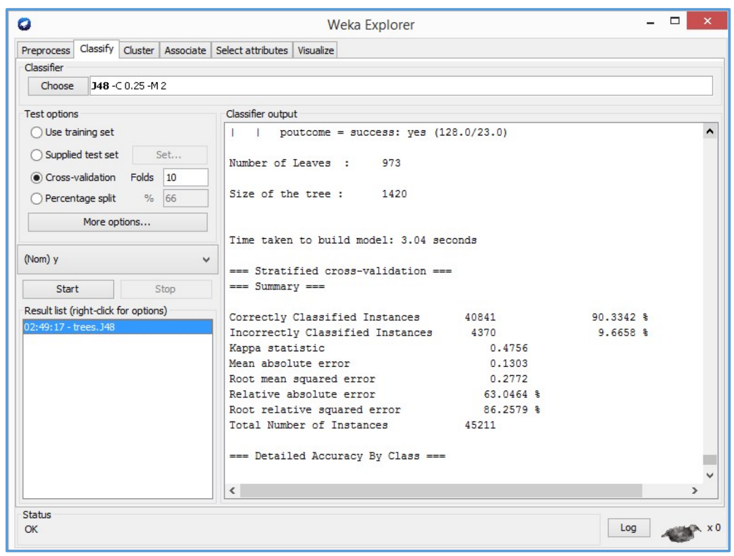
**Dataset**: bank_full1.csv

## Module explanation:

To answer this question we removed one attribute at a time and applied J48 decision tree and we got the accuracy. While removing the decisive attribute, J48 decision tree will give the least accuracy while compared to removing other attributes.

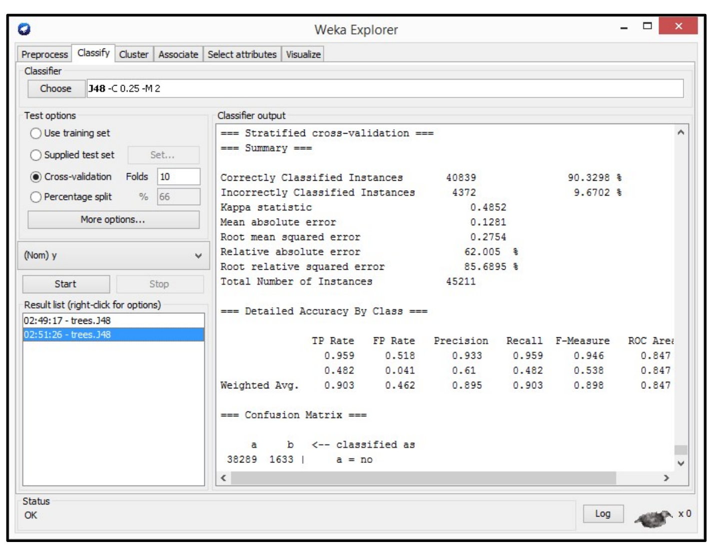
Contact is removed

(Accuracy=90.3342%)



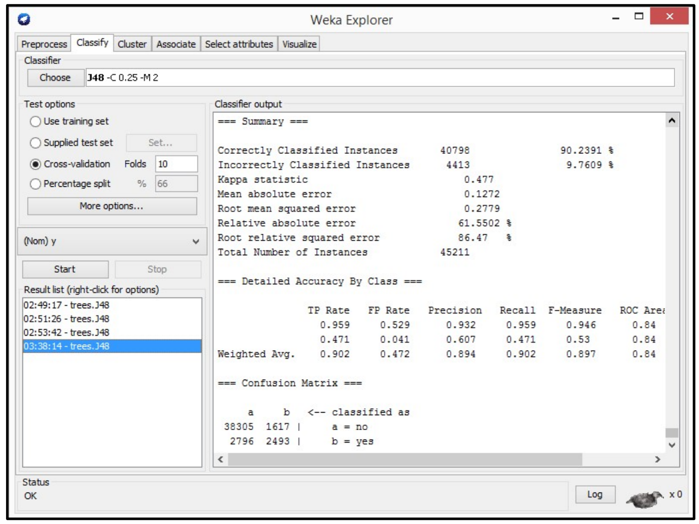

Age is removed

(Accuracy=90.3298%)



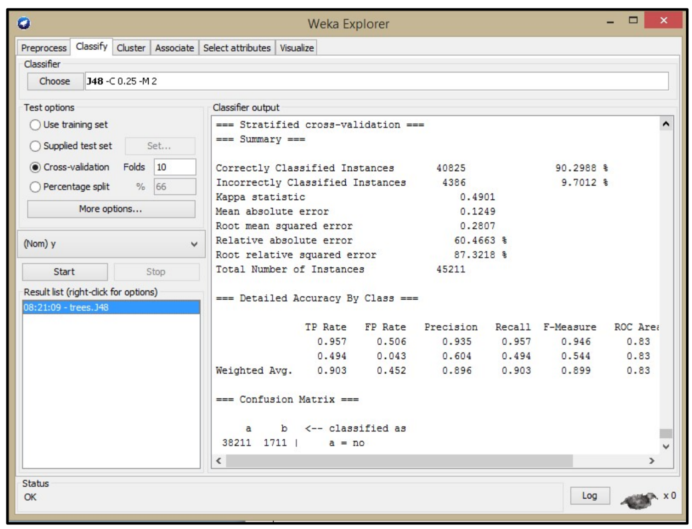
Marital is removed

(Accuracy=90.2392%)



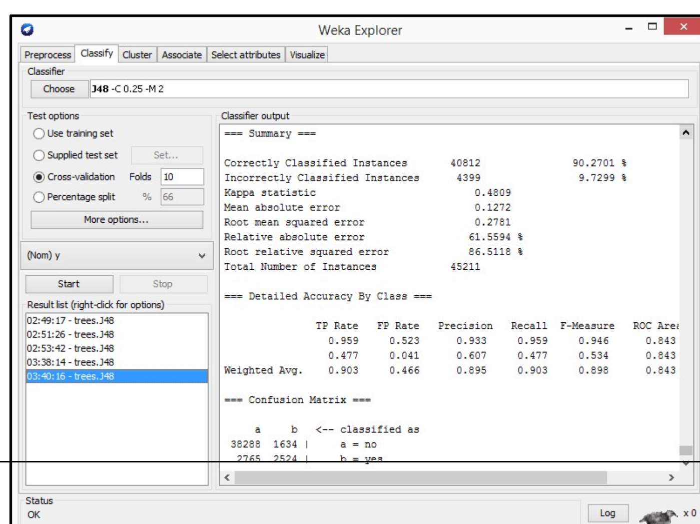

Job is removed

(Accuracy=90.2988%)



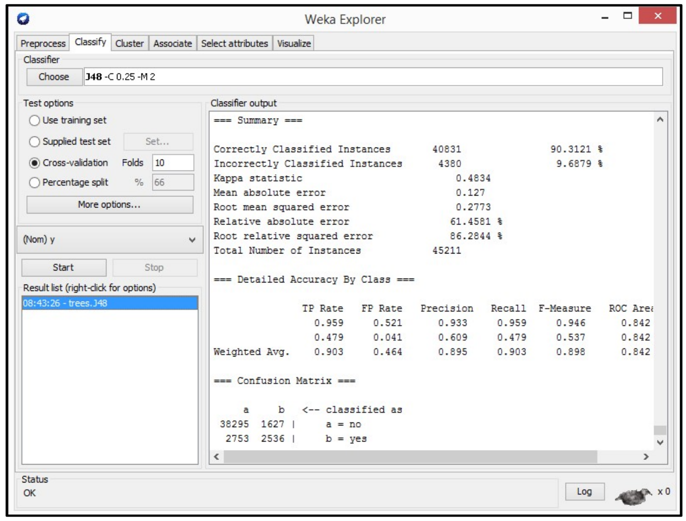

Education is removed

(Accuracy=90.2701%)



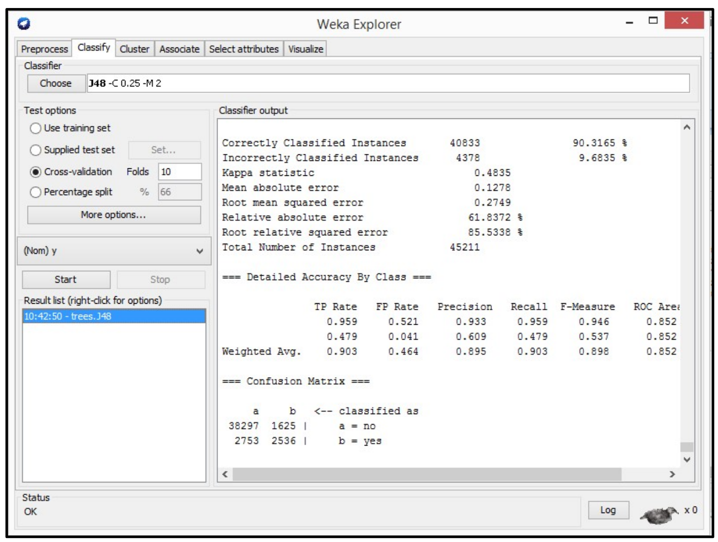
Default is removed

(Accuracy=90.3121%)



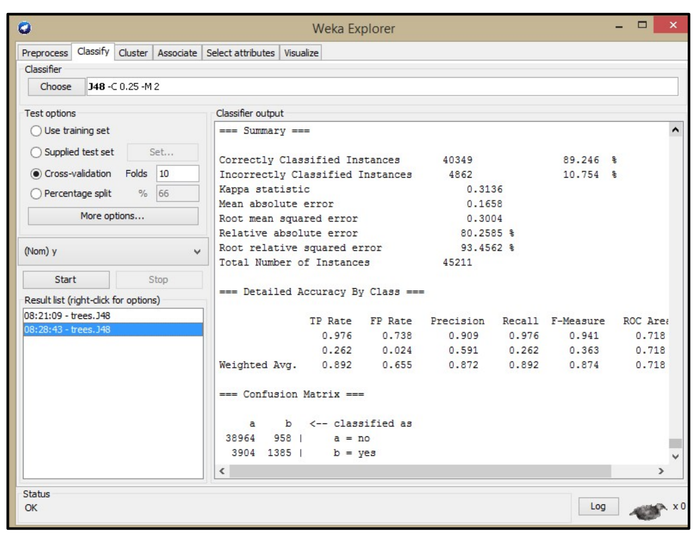

Balance is removed

(Accuracy=90.3165%)




Duration is removed

(Accuracy=89.246%)



Similarly,

- Housing is removed(Accuracy=90.3674%)
- Loan is removed(Accuracy=90.332%)
- Day is removed(Accuracy=90.0776%)
- Month is removed(Accuracy=89.7765%)
- Campaign is removed(Accuracy=90.374%)
- Pdays is removed(Accuracy=90.2612%)
- Previous is removed(Accuracy=90.3497%)
- Poutcome is removed (Accuracy=89.7038%)

Hence it is known that **Duration i**s the decisive attribute

## Module 6:

**Objective**: To predict the telemarketing skills of an employee before hand.

**Code**: telephone_cluster.R

**Data Set**: employee.csv

**Major Techniques Used**: hierarchical clustering, elbow method

**Packages Included**: factoextra, stats

**Output**: Predicting number of policies that the employee can conform within a month.

## Module Explanation:

Step 1:

Telemarketing Data of 20 employees over a month has been generated and considered as input data set.

Step 2:

Elbow method using hcut is applied to know the optimal number of clusters to be generated.

It resulted to be 3

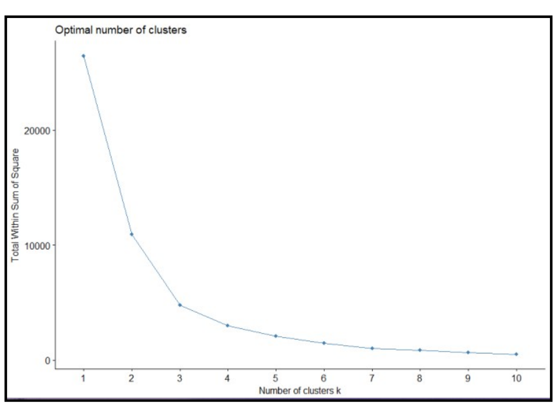


Fig 31:Elbow Method

Step 3:

Then a distance matrix is constructed from the input data which is then converted to matrix and is stored in a variable 'a' .

Step 4:

Clusters are formed using complete method in hclust based on the attributes avg_call duration, avg_customer_interaction, avg_e_pitch, avg_c_pitch, avg_e_emotion, avg_c_emotion. The integer vector containing cluster numbers for each tuple is stored back as a feature in the input table.

Step 5:

Now a new employee record is added into the dataset without having the class label(no_of_policies).Again clustering is applied and the cluster number of this new record is taken into a variable 'no'.

```
1 2 1 2 1 2 2 3 1 3 3 3 1 1 3 1 2 1 3 1 2
```

Fig 32:Cluster array

Step 6:

Using , for loop and distance matrix ,the nearest employee within the particular cluster of the new record are known. This employee's no_of_policies is the predicted range of the new employee.

```
> t2<-table(cl,data1$no_of_policies)
> t2

cl    100-150 110-160 125-175 130-180 25-75 30-80 55-105 70-120 75-125 90-
140
  1 0      2      0      1      1    0     1      1      1      1      1
  2 1      0      0      0      0    1     1      0      1      0      2
  3 0      2      1      0      0    1     0      0      0      0      0

cl  94-145 95-145
  1      0      0
  2      0      0
  3      1      1
> print("cluster no's of tuples")
[1] "cluster no's of tuples"
> no
 [1] 1 2 1 2 1 2 2 3 1 3 3 3 1 1 3 1 2 1 3 1 2
> print("cluster of new tuple")
```

```
[1] "cluster of new tuple"
> num
[1] 2
> print("most nearest tuple")
[1] "most nearest tuple"
> min_index
[1] 2
> print("most probable range of policies he can take are:")
[1] "most probable range of policies he can take are:"
> data1[min_index,11]
[1] 70-120
13 Levels:  100-150 110-160 125-175 130-180 25-75 30-80 55-105 70-120 ...
95-145
```

Fig 33:Cluster vs Policy ranges

Code:

```
library(factoextra)
data1<-read.csv("C://Users//Admin//Desktop//employee.csv",header=TRUE)
fviz_nbclust(data1[,c(-1,-2,-3,-4,-11)],hcut, method = "wss")

d<-data1[,c(-1,-2,-3,-4,-11)]
d1<-dist(d)
d1
class(d1)
a<-as.matrix(d1)
a#distance matrix
class(a)
a[2][1]
cl<-cutree(hclust(d1,method="complete"),3)
class(cl)
cl
no<-cl#cluster no matrix
class(no)
data1$no<-no
data1
t2<-table(cl,data1$no_of_policies)
t2
class(t1)
num<-tail(cl,n=1)
num#cluster no of new point

min<-1000
m<-1
min_index<-0
```

```
for(i in 1:20)
{if(no[m]==num)
{
  #if(num>m){
  if(min>a[num,m])
  {min<-a[num,m]
   min_index<-m}
#}
  }
  m<-m+1
}
```

Fig 34:Code for Hierarchical Clustering

**CONCLUSION:**

So finally we are able to get satisfiable results for the idea we have started with , using different data mining techniques which were applied on different data sets of banking sector.

**Future Work:**

1. Speech recognition techniques have to be included to get the attributes such as pitch, amplitude, frequency which will be able to explain emotions of the customer care employee and the customer. This can be a major improvement in the field of telemarketing.

2. In present banking system we have many bank applications to fulfil the needs of people, however some inconvenience still persists. One such inconvenience is , to change the address of the customer in passbook either he has to approach the bank and submit a letter along with required proofs or can mail the same in case of some private banks. Automating this by verifying the Adhar Id and finger print of customer Online would make the process more convenient.

3. The Policy dataset and Loan dataset which is used in this project is generated randomly. But better works can be done on the standard bank dataset ,to find the hidden relationships between the customer attributes and the policies or term deposits provided by banks. Number of attributes can be improved. For instance: the policy dataset we generated considered only few attributes to prepare training set ,thus resulted in single policy per customer which is not a real scenario.

4. Product Segmentation:

   This is used to identify the needs of the customers and develop the products (such as schemes, policies, term deposits) according to their needs. Here the attributes of the customer will be the deciding factors to develop a new product.

5. Customer targeting:

   Let us suppose banks are providing some new policies and they want to apply customer targeting to improve the profits of bank. The attributes of policies will be the deciding factors along with the historical data of the customers. Data mining can be done to select the customers to notify about the policy. For instance, for old and reliable customers time to time notifications can be sent pre calculating their requirements using customer data. This reduces the cost of marketing by targeting subset of customers as well as strengths the Customer-Bank relationships.

## TIMELINE:

| No of days spent | Implementation activities(process objectives) |
|---|---|
| 7 | Initially we got 6 base papers on applications of data mining on banks. Out of them 2 were found to be useful for the first module (based on loans) and remaining 4 were regarding credit card fraud detection. |
| 4 | Data set for credit card fraud detection was unavailable because it's confidential. So we generated a random data set. Then we applied various classification algorithms and got 98% accuracy in classifying the transactions. But later we discarded this module because in the real world existing system is more efficient and secure. |
| 5 | Even for the first module the data set is not available. So we took reference about the data set from the references given in the next section and generated the data set accordingly. We applied various classification algorithms to identify the category of payment of loans by the customer and identified the technique with best accuracy. |
| 4 | Now we went on with a novelty of recommendation of policies to the customers for better customer retention management and the standard dataset was not available and we couldn't generate them because it leads to spurious database table |
| 3 | Since standard datasets were not available for any of the previous modules we searched for a standard data set and finally we got bank telemarketing dataset from UCI repository and we found the base paper related to this dataset. We performed the classification algorithms given in the dataset and found the best algorithm to identify whether a person will accept the term deposit. |
| 2 | In the base paper considered above they considered 69 attributes and answered various interesting questions. But the present dataset consists of only 17 attributes so we couldn't answer all the questions. So to get interesting rules out of the limited data we did association mining with less support and confidence value. |
| 4 | While going through the dataset we found that some people who accepted for the term deposit previously didn't accept the term deposit in the current campaign. To know the reason for the change we went with template matching and got the reasons. Nothing more useful can be done with the attributes given in the data set so with this we concluded this module. To know which is the more deciding attribute for accepting term deposit we used weka j4.8 algorithm where we removed each attribute and calculated the accuracy for classification. We found that duration_of_call is the more deciding attribute. |

| | |
|---|---|
| 8 | We thought of another novelty which requires policy dataset. So we generated policy dataset with less number of attributes and few policies avoiding the presence of spurious tuples. We mapped the policy dataset with the loan dataset providing some constraints using MYSQL. The attributes profit and risk percentage were added to this dataset. Now the new customer will be recommended with a policy which is obtained by training the policy dataset with classification algorithms. |
| 4 | Instead of recommending the most probable policy for a customer we thought of recommending the customer with an ordered policy list based on profit and risk percentage. We used clustering to achieve this. |
| 4 | We got an idea from the telemarketing module that we can extract rules whether to call a customer to explain him about the product. We retrieved income and job attributes values of the customers who accepted the term deposits from the telemarketing dataset and using MYSQL we were able to add the class label (call =yes/no) to policy dataset. Now we generated rules whether to call a person using template matching |
| 2 | One more idea for telemarketing: Generally bank customer care employees will call the customers and explain about the policies. The drawback which the bank may face is, all the customers will not have much patience to listen to all policies. So we thought of generating a rank wised list of policies which will be more related for the customer. Classification and clustering techniques can be applied here. |
| 2 | To reduce the manual work in hiring an employee for telemarketing we thought of an idea where attributes such as emotion of the employee, pitch of the voice, customer interaction time and some other attributes are considered. Clustering techniques are implemented to estimate the telemarketing skills of new employee |

Total number of days spent : 49 days

# REFERENCES:


- http://facweb.cs.depaul.edu/mobasher/classes/ect584/WEKA/classify.html
- http://ieeexplore.ieee.org/document/5619002/
- http://ieeexplore.ieee.org/document/7093731/
- http://ieeexplore.ieee.org/document/8076775/
- http://www.ijetae.com/files/Volume3Issue8/IJETAE_0813_35.pdf
- https://archive.ics.uci.edu/ml/datasets/bank+marketing
- http://shodhganga.inflibnet.ac.in/bitstream/10603/25027/12/chapter-8.pdf
- http://research.ijcaonline.org/volume85/number7/pxc3893218.pdf
- A Novel Framework for Stock Trading Analysis Using Casual Relationship Mining
  Harchana Bhoopathi ,B.Rama
- 201O International Conference on Computer Application and System Modeling (ICCASM 2010)
  Data Mining Application in Banking-Customer Relationship Management
  Zhao Li Ping[l] Shu Qi Liang[2]
- 8th International Conference on Advanced Computational Intelligence. Chiang Mai, Thailand; February 14-16, 2016 Decision Support System for Investing in Stock Market by using OAA-Neural Network Sabaithip Boonpeng and Piyasak Jeatrakul
- Proceedings of the 2015 International Conference on Industrial Engineering and Operations Management. Dubai, United Arab Emirates (UAE), March 3 – 5, 2015.Data mining application in banking sector with clustering and classification methods.Aslı Çaliş,Ahmet Boyaci,Kocaeli University