Prabin Karki

Final Project

Applied Analytics

25th May 2020

In this project, we will be developing a model to predict if a client will purchase a term deposit or not after a marketing campaign based on phone calls by a Portuguese banking institution.

**Summary Statistics**

In the dataset, we have a total of 15 predictor variables and a categorial response variable. Among the predictor variables, 10 are categorical predictors whereas 5 of them are numeric predictors.

# Data import

*bank <- read.csv("D:\\Spring 2020\\AppliedAnalytics223\\FinalProject\\Data\\bank- additional.csv")*

*summary(bank)*

Age is the first numeric predictor we have in the dataset. It ranges from 18 years to 88 years with mean age of 40.11 years. Duration of the call is another numeric variable with calls as short as 0 seconds and as long as 3643 seconds which must be an outlier as the mean sits around 256.8 seconds. Similarly, we have campaign variable, which is the number of contacts performed during the campaign for a client. So, the minimum times contacted is once and the maximum is 35 for some client which again looks like an exception as its mean is around 2.5.

pdays and previous are other numeric variables in the dataset. Variable pdays is measuring the number of days that passed by after the client was last contacted from a previous campaign which would be 999 if the client was not previously contacted, so the minimum for it is 0 days and the mean is 960.4, the 1st quartile, median, 3rd quartile and the max is 999 days. The previous variable has the count of contacts performed before the campaign. It ranges from 0 to 6. When I looked at table() output for these variables, I think treating them as qualitative variables makes more sense when developing the model.

For the categorial predictors, I used table() function to calculate the summary. I did not notice any thing out of ordinary. After the campaign, 3668 clients rejected the offer and 451 clients accepted it.

The summary statistics of job of the clients is:  
> table(bank$job)

admin. blue-collar entrepreneur housemaid management retired

1012 884 148 110 324 166

self-employed services student technician unemployed unknown

159 393 82 691 111 39

The table below shows the stats for marital status of the clients.

> table(bank$marital)

divorced married single unknown

446 2509 1153 11

The table below summarizes education of the clients.

> table(bank$education)

basic.4y basic.6y basic.9y high.school

429 228 574 921

illiterate professional.course university.degree unknown

1 535 1264 167

For default variable:

> table(bank$default)

no unknown yes

3315 803 1

For housing loan variable:

> table(bank$housing)

no unknown yes

1839 105 2175

For personal loan variable:

> table(bank$loan)

no unknown yes

3349 105 665

For contact method variable:

> table(bank$contact)

cellular telephone

2652 1467

For month of last contact variable:

> table(bank$month)

apr aug dec jul jun mar may nov oct sep

215 636 22 711 530 48 1378 446 69 64

For last contact day of the week variable:

> table(bank$day\_of\_week)

fri mon thu tue wed

768 855 860 841 795

And last but not the least, for poutcome i.e. outcome of previous campaign variable:

> table(bank$poutcome)

failure nonexistent success

454 3523 142

**Logistic Regression Model**

***Data Preparation***

The values for the y i.e. response variable is assigned as 1 for “yes” (the client subscribed the term deposit) and 0 for the contrary.

*bank$y = ifelse(bank$y == "yes", 1, 0)*

I used the following command to separate my training dataset and my testing dataset. I went with approximate split of 80% of the data for training and 20% for the testing.

*train\_sample = sample(4119, 3300)*

*train\_data = bank[train\_sample, ]*

*test\_data = bank[-train\_sample, ]*

As said, the variable duration is not known before a call is performed and after the end of the call, y is known, I will not be using this variable in my prediction model. If duration is used to predict the outcome, this feature might be overpowered and shadowing the other important predictors.

***Building the Model***

The first model I built had all the predictor variables that we selected to keep in our model from the dataset.

*model1 = glm(data = train\_data, y ~ age + job + marital + education + default + housing + loan + contact + month + day\_of\_week + campaign + pdays + previous + poutcome, family = binomial)*

*summary(model1)  
anova(model1, test = "Chisq")*

The value for AIC of this model is 1955.7 whereas the residual deviance is 1861.7. There are a lot of insignificant predictors in the model as seen from the output of the summary command. First, I decided to check the significance of the numeric predictors i.e. age, campaign, pdays and previous. All of them were significant except pdays with pvalue of 0.881892. But before taking any action on the omission of the variables, I ran anova command on the model.

The insignificant variables according to the anova command were, education, housing, loan, and day\_of\_week with p-values 0.2496, 0.78444, 0.6279 and 0.7793, respectively. Rather than removing the least significant among them, I decided to remove the day of week variable as it is unlikely that it has any major impact on the decision of the clients. So, I ran model2 with all the variables except the day\_of\_week.

*model2 = glm(data = train\_data, y ~ age + job + marital + education + default + housing + loan + contact + month+ campaign + pdays + previous + poutcome, family = binomial)  
summary(model2)  
anova(model2, test = "Chisq")*

The AIC value for this model was 1950, improvement over the previous model with residual deviance of 1864. Again, after running the Chisq anova test, the insignificant variables were still highly insignificant with similar pvalues. Another model was built, this time eliminating the housing loan variable that yielded AIC of 1948.2. Again, I rectified the model by removing the education variable although I thought that it had very high significance in the model, but the data were not just supporting the idea. The model improved as the new AIC value was only 1936.8. But again, the presence of the insignificant loan variable was holding the model to be even better. So, at last, I developed a model with the variables, age, job, marital, default, contact, month, campaign, pdays, previous and poutcome and got a AIC of 1934.9 and residual deviance of 1868.9, the most well fitted model yet.

Although the anova command was showing that the pdays variable was statistically significant to stay in the model, when I looked at the summary, I noticed that it is pvalue is 0.9329 and the coefficient was positive, though small. As it did not make sense to me, the greater number of days from the past contact, the more likely is the person to accept the offer and there exists the previous variable which is keeping track of the number of contacts previously made, I decided to omit it from the model. The command that I used to build my final model is:

*model5 = glm(data = train\_data, y ~ age + job + marital + default + contact + month + campaign + previous + poutcome, family = binomial)*

*summary(model5)*

model5 gave me AIC value of 1933 and residual deviance of 1869, so I decided to settle on this model for my final model. I also tried to regress by treating pdays as factors because the 999 as a lot of values were not looking good for my model but it made the model worse, so I dropped the idea.

***Testing the Model***

To test the model, I used the test dataset that we had separated from the training set earlier.

To produce test statistics for the testing, I used ConfusionMatrix() function from the caret package and CrossTable() from the gmodels package.

After using the following code to predict the probabilities of a client accepting or rejecting the offer, I used 0.5 threshold probability to start the testing.

*predicted = predict(model5, newdata = test\_data, type = "response")*

*classify = ifelse(predicted>.5, 1, 0)*

The CrossTable(test\_data$y, classify) function gave me the following confusion matrix.

| classify

test\_data$y | 0 | 1 | Row Total |

-------------|-----------|-----------|-----------|

0 | 728 | 9 | 737 |

| 0.191 | 6.611 | |

| 0.988 | 0.012 | 0.900 |

| 0.915 | 0.391 | |

| 0.889 | 0.011 | |

-------------|-----------|-----------|-----------|

1 | 68 | 14 | 82 |

| 1.717 | 59.416 | |

| 0.829 | 0.171 | 0.100 |

| 0.085 | 0.609 | |

| 0.083 | 0.017 | |

-------------|-----------|-----------|-----------|

Column Total | 796 | 23 | 819 |

| 0.972 | 0.028 | |

-------------|-----------|-----------|-----------|

Below is the output of the function confusionMatrix from the package caret.

> confusionMatrix(table(classify, test\_data$y))

Confusion Matrix and Statistics

classify 0 1

0 728 68

1 9 14

Accuracy : 0.906

95% CI : (0.8839, 0.9251)

No Information Rate : 0.8999

P-Value [Acc > NIR] : 0.3042

Kappa : 0.233

Mcnemar's Test P-Value : 3.851e-11

Sensitivity : 0.9878

Specificity : 0.1707

Pos Pred Value : 0.9146

Neg Pred Value : 0.6087

Prevalence : 0.8999

Detection Rate : 0.8889

Detection Prevalence : 0.9719

Balanced Accuracy : 0.5793

'Positive' Class : 0

The accuracy of the model was 90.6% with 9 false positive and 68 false negatives. The sensitivity is 0.9878 and the specificity is 0.1707, whereas the accuracy after considering the false reports is 57.93. Since, campaign of this sorts is a big thing for banks, if banks were supposed to use this model, they might end up not carrying out the campaign as the it falsely reported 68 possible clients who might have subscribed the plan i.e. it is not reported in true positives. So, to reduce the number of false reports, I tried different threshold probabilities and finally decided to settle on 0.17. The confusion matrix for 0.17 threshold is below:

*predicted1 = predict(model5, newdata = test\_data, type = "response")*

*classify1 = ifelse(predicted>.17, 1, 0)*

*CrossTable(test\_data$y, classify1)*

Total Observations in Table: 819

| classify1

test\_data$y | 0 | 1 | Row Total |

-------------|-----------|-----------|-----------|

0 | 695 | 42 | 737 |

| 1.271 | 11.904 | |

| 0.943 | 0.057 | 0.900 |

| 0.939 | 0.532 | |

| 0.849 | 0.051 | |

-------------|-----------|-----------|-----------|

1 | 45 | 37 | 82 |

| 11.422 | 106.989 | |

| 0.549 | 0.451 | 0.100 |

| 0.061 | 0.468 | |

| 0.055 | 0.045 | |

-------------|-----------|-----------|-----------|

Column Total | 740 | 79 | 819 |

| 0.904 | 0.096 | |

-------------|-----------|-----------|-----------|

Again, the output from the confusionMatrix command is:

> confusionMatrix(table(classify1, test\_data$y))

Confusion Matrix and Statistics

classify1 0 1

0 695 45

1 42 37

Accuracy : 0.8938

95% CI : (0.8706, 0.914)

No Information Rate : 0.8999

P-Value [Acc > NIR] : 0.7420

Kappa : 0.4007

Mcnemar's Test P-Value : 0.8302

Sensitivity : 0.9430

Specificity : 0.4512

Pos Pred Value : 0.9392

Neg Pred Value : 0.4684

Prevalence : 0.8999

Detection Rate : 0.8486

Detection Prevalence : 0.9035

Balanced Accuracy : 0.6971

'Positive' Class : 0

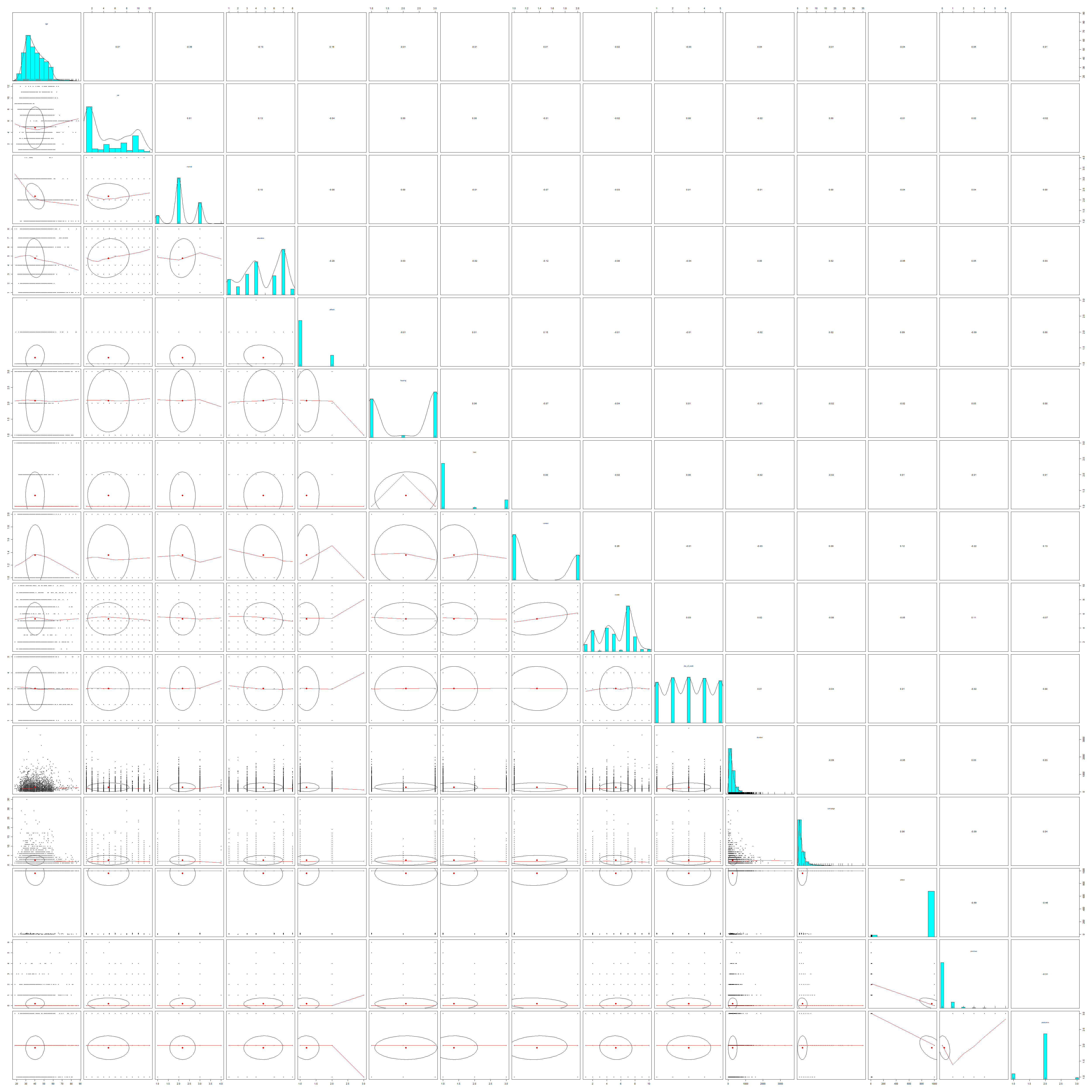
The accuracy of this model is 89.4% which is less than the prediction model above. But in this model, the rate of true positives is higher, and the value of specificity improved dramatically. The balanced accuracy of 69.71 explains the betterment of this model over the last one. Notice that the rate of false negative also decreased but the rate of false positives increased. The main reason why I decided to settle on this model as my final prediction model is because the number of false negatives and the number of false positives are equal. So, if we were to increase the accuracy and the rate of true positives/ true negatives, the bank personnel’s might be deceived by the underwhelming number of the less possible clients that would subscribe.

Banks invest a lot of manpower and resources in such campaigns, so a model which might predict the true negatives and true positives better but had high values for false positives, then banks might end up investing the manpower and resources but not getting the outcome as predicted. And if the model had high values for false negatives, the bank might not even decide to carry out the campaign due to disappointing success rates. By making the model report equal number of false negatives and false positives while still maintain the high values for sensitivity and specificity, banks will be getting the numbers of clients who accept to offer as predicted from the model which directly although might have false numbers but then the false’s cancel out and yielding the predicted numbers after the campaign.

**Bayesian Classification Model**

***Data Preparation***

Since Naïve Bayes classification’s performance degrades if the dataset contains highly correlated features, I produce the following correlation pairs plot to check and rectify the data.



From the plots above, we can see that the variables pdays, previous and poutcome is somewhat correlated with correlation coefficient of -0.59 between pdays and previous and -0.46 between pdays and poutcome. So, I will be removing the variable previous and keeping the variables pdays and poutcome as I think both play a significant role in the prediction of the client’s response. Also, the variable duration will not be used in the model for previously mentioned reasons. Correlation coefficient is -0.39 for marital and age but since both variables were significant in the regression model, I will keep both of those.

Furthermore, I will be eliminating variable day of week which was insignificant in the regression model. As we can say that the probability of a client investing in bank term deposit is less likely either if the person is already engaged in housing loan or personal loan, although they are the measure of similar kinds and were insignificant in the regression model, I will keep both of them in this Bayesian model.

***Building the Model***

So, the variables that are selected to be included in the model are: age, job, marital, education, default, housing, loan, contact, month, campaign, pdays and poutcome. With all this, the following command is used to build the first model based on Naïve Bayes.

*bayesmodel2 = naiveBayes(data = train\_data,y ~ age + job + marital + default + housing + loan + contact + month+ campaign + previous + poutcome, laplace = 0)*

The conditional probabilities are for each variable were:

**job**

Y admin. blue-collar entrepreneur housemaid management retired self-employed 0 0.2347 0.22654 0.03582 0.0276800 0.0796233 0.0354870 0.0402297

1 0.29269 0.135555 0.01626 0.0216802 0.07060705 0.092921 0.0352303

Y services student technician unemployed unknown

0 0.099624701 0.017059024 0.168543159 0.024223814 0.009894234 1 0.070460705 0.037940379 0.176151762 0.043360434 0.008130081

**marital**

Y divorced married single unknown

0 0.110201296 0.624701467 0.262026612 0.003070624

1 0.084010840 0.563685637 0.352303523 0.000000000

**education**

Y basic.4y basic.6y basic.9y high.school illiterate professional.course

0 0.10951 0.0539065 0.1453428 0.223473217 0.0003415 0.12964858

1 0.0894308 0.0352303 0.0894308943 0.205962 0.000000 0.1490514905

Y university.degree unknown

0 0.3012623678 0.0365063118

1 0.3739837398 0.0569105691

**default**

Y no unknown yes

0 0.7905151825 0.2091436370 0.0003411805

1 0.8861788618 0.1138211382 0.0000000000

**housing**

Y no unknown yes

0 0.44694643 0.02729444 0.52575913

1 0.43089431 0.01897019 0.55013550

**loan**

Y no unknown yes

0 0.81030365 0.02729444 0.16240191

1 0.83197832 0.01897019 0.14905149

**contact**

Y cellular telephone

0 0.6141249 0.3858751

1 0.8319783 0.1680217

**month**

Y apr aug dec jul jun mar may 0 0.0481644 0.1552312 0.00341180 0.17468440 0.12487205 0.0058000 0.358239509

1 0.0867267 0.1511518 0.02810298 0.13001301 0.14363436 0.0670678 0.200542005

Y nov oct sep

0 0.106789492 0.012623678 0.010235415

1 0.078590786 0.054200542 0.056910569

**campaign**

Y [,1] [,2]

0 2.559536 2.579544

1 1.962060 1.415624

**pdays**

Y [,1] [,2]

0 981.7308 129.7961

1 772.8537 417.1236

**poutcome**

Y failure nonexistent success

0 0.10406005 0.88092801 0.01501194

1 0.16260163 0.63143631 0.20596206

There are certain features from the above table that stand out. From the job table, the clients whose job were unknown had the least probability of subscribing to the service than any other jobs. And retired people were 3 times more likely to subscribe the offer than not subscribing. The people with jobs such as admin, blue-collar and technician were more likely to accept the offer. Also, another interesting data is that education wise, given that a client was illiterate, there is 0 probability that he/she accepts the offer whereas the clients with university degree were most likely to accept the offer. Also, when a client had credit default, there is again 0 probability that he/she will subscribe the bank term service. People with no loans are extremely likely to subscribe and interestingly, those who were contacted by cellular were 7 times more likely to accept the offer that who were contacted by the telephone. And, for those, whose poutcome were nonexistent or simply were not contacted by the bank once in the past were mot likely to reject the offer with a probability of 0.88. For the numeric variables age, campaign and pdays, the table is showing the mean and the standard deviation as [,1] for mean and [,2] for standard deviation.

***Testing and Improving the Model***

As I used the same training dataset from the logistic regression to build the model, I will be using the same test dataset to test the model as well so that we can compare and contrast the two methods. I used the threshold probability of 0.5 to start testing the model.

*predicted3 = predict(bayesmodel1, test\_data, type = "raw")*

*classify3 = ifelse(predicted3[,2]>=.5, 1, 0)*

*CrossTable(test\_data$y, classify3)*

The output of the crosstable() function is:

Total Observations in Table: 819

| classify3

test\_data$y | 0 | 1 | Row Total |

-------------|-----------|-----------|-----------|

0 | 721 | 16 | 737 |

| 0.425 | 8.984 | |

| 0.978 | 0.022 | 0.900 |

| 0.922 | 0.432 | |

| 0.880 | 0.020 | |

-------------|-----------|-----------|-----------|

1 | 61 | 21 | 82 |

| 3.821 | 80.748 | |

| 0.744 | 0.256 | 0.100 |

| 0.078 | 0.568 | |

| 0.074 | 0.026 | |

-------------|-----------|-----------|-----------|

Column Total | 782 | 37 | 819 |

| 0.955 | 0.045 | |

-------------|-----------|-----------|-----------|

Here, the total accuracy of the model in predicting is 90.6% with 16 false positives and 61 false negatives. We have 721 true negatives and 37 true positives. The sensitivity of the model is 0.97 and specificity is 0.26 with balanced accuracy of 0.6172 which can be verified by the output of the counfustionMatrix() function below.

> confusionMatrix(table(classify3, test\_data$y))

Confusion Matrix and Statistics

classify3 0 1

0 721 61

1 16 21

Accuracy : 0.906

95% CI : (0.8839, 0.9251)

No Information Rate : 0.8999

P-Value [Acc > NIR] : 0.3042

Kappa : 0.31

Mcnemar's Test P-Value : 5.324e-07

Sensitivity : 0.9783

Specificity : 0.2561

Pos Pred Value : 0.9220

Neg Pred Value : 0.5676

Prevalence : 0.8999

Detection Rate : 0.8803

Detection Prevalence : 0.9548

Balanced Accuracy : 0.6172

'Positive' Class : 0

Before I moved further in improving the model, I wanted to try pdays as factors as I think it will have great effect in this classification though it made the model worse in the logistic regression. The output of the crossTable and ConfusionMatrix for pdays when treated as factors is below:

> CrossTable(test\_data$y, classify3)

Total Observations in Table: 819

| classify3

test\_data$y | 0 | 1 | Row Total |

-------------|-----------|-----------|-----------|

0 | 710 | 27 | 737 |

| 0.736 | 10.223 | |

| 0.963 | 0.037 | 0.900 |

| 0.929 | 0.491 | |

| 0.867 | 0.033 | |

-------------|-----------|-----------|-----------|

1 | 54 | 28 | 82 |

| 6.614 | 91.878 | |

| 0.659 | 0.341 | 0.100 |

| 0.071 | 0.509 | |

| 0.066 | 0.034 | |

-------------|-----------|-----------|-----------|

Column Total | 764 | 55 | 819 |

| 0.933 | 0.067 | |

-------------|-----------|-----------|-----------|

> confusionMatrix(table(classify3, test\_data$y))

Confusion Matrix and Statistics

classify3 0 1

0 710 54

1 27 28

Accuracy : 0.9011

95% CI : (0.8786, 0.9207)

No Information Rate : 0.8999

P-Value [Acc > NIR] : 0.482971

Kappa : 0.3571

Mcnemar's Test P-Value : 0.003866

Sensitivity : 0.9634

Specificity : 0.3415

Pos Pred Value : 0.9293

Neg Pred Value : 0.5091

Prevalence : 0.8999

Detection Rate : 0.8669

Detection Prevalence : 0.9328

Balanced Accuracy : 0.6524

'Positive' Class : 0

We can see a good amount of increase in specificity from 0.256 to 0.341. As with my idea of making false negatives and false positives equal for the overall betterment of the model for banks seems to be favored as false positives went up to 27 and false negatives dropped to 54 as well as the true positive went up to 28. All the improvement is reflected in the balance accuracy of 65.2% from 61.7% of the previous model with pdays treated as numeric variables.

After playing around for a threshold probability, I finally decided to settle on the value 0.38 as it gave the best result for my needs.

The output for 0.38 as threshold probability is below:

| classify3

test\_data$y | 0 | 1 | Row Total |

-------------|-----------|-----------|-----------|

0 | 689 | 48 | 737 |

| 0.801 | 7.500 | |

| 0.935 | 0.065 | 0.900 |

| 0.931 | 0.608 | |

| 0.841 | 0.059 | |

-------------|-----------|-----------|-----------|

1 | 51 | 31 | 82 |

| 7.196 | 67.407 | |

| 0.622 | 0.378 | 0.100 |

| 0.069 | 0.392 | |

| 0.062 | 0.038 | |

-------------|-----------|-----------|-----------|

Column Total | 740 | 79 | 819 |

| 0.904 | 0.096 | |

-------------|-----------|-----------|-----------|

> confusionMatrix(table(classify3, test\_data$y))

Confusion Matrix and Statistics

classify3 0 1

0 689 51

1 48 31

Accuracy : 0.8791

95% CI : (0.8548, 0.9007)

No Information Rate : 0.8999

P-Value [Acc > NIR] : 0.9769

Kappa : 0.3181

Mcnemar's Test P-Value : 0.8407

Sensitivity : 0.9349

Specificity : 0.3780

Pos Pred Value : 0.9311

Neg Pred Value : 0.3924

Prevalence : 0.8999

Detection Rate : 0.8413

Detection Prevalence : 0.9035

Balanced Accuracy : 0.6565

'Positive' Class : 0

Here, we have almost equal false negatives and false positives with 48 and 51 misclassifications, respectively. Balanced accuracy is 65.6%. We managed to boost the true positive to 31 and specificity to 0.3780. The overall accuracy of the model is 87.9%.

***Comparing Logistic Regression and Naïve Bayes Classification***

According to my observation, between the final selected models for Logistic Regression and Naïve Bayes Classification, Logistic Regression performed better. The accuracy of the logistic model was 89.4% whereas the accuracy of the Bayesian model was 87.91%, again these are the numbers when the model was fitted to best fit my idea for the use of bank’s purposes. The balanced accuracy of the Logistic model was about 69.7% whereas the Bayesian had about 65.6%.

Talking about the exact classifications, out of 819 observations, logistic regression gave me 695 true negatives, 37 true positives, 42 false positives and 45 false negatives. The sensitivity was 0.94 and the specificity was 0.45. On the other hand, Bayesian Classification gave me 689 true negatives, 31 true positives, 48 false positives and 51 false negatives. The sensitivity was 0.93 and the specificity was about 0.38. In terms of every statistics, the logistic regression fitted my requirements better.