PREDICTIVE ANALYSIS OF RE-HOSPITALIZATION IN DIABETIC PATIENTS

ALY 6020 FINAL REPORT - **GROUP 7**

Predictive Analytics Fall 2020

Submitted to –

Prof. Ajit Appari

Northeastern University, CPS Analytics

Boston, MA

Submitted by –

SIDDHARTH SAXENA

DONGLING JIANG

Submitted on –

December 11th, 2020

PROJECT OVERVIEW

As of this week, work has been completed:

(1) Identify the data set to be used for the analysis.

(2) Determine the problem to be solved by using predictive analysis.

(3) The first method/technique -- KNN is used to solve the problem raised.

(4) Use the second method/technique to solve the problem raised.

(5) Compare the accuracy of the two methods and analyze the reasons for the differences.

(6) Future work, use other classifiers to solve classification problems, such as decision tree, naive Bayes classifier.

EXECUTIVE SUMMARY

Diabetes can be considered to be a chronic condition that is prevalent in over 100 million people across the world. In the case of Diabetic patients, the chances of readmission are quite high. Despite major advances in science and technology, diabetes continues to be a chronic disease, with a thirty-day readmission rate of around 20%, as compared to an average of 12% for the rest of the diseases. Additionally, readmissions cost hospitals a large amount of money in terms of financial penalties and etc., so the end goal from a hospital's point of view is to be

able to identify and reduce the possibility of a re-admission in patients. Prevention of patient readmission has been given a greater importance due to large amounts of re-admission costs that are involved.

So, we consider the “diabetic_data” to build our prediction models. The dataset comprises of records from 1999 to 2008. This data was collected from 130 different US hospitals and integrated delivery networks. The dataset comprises of 50 features that provide with specific details about the patient based on the patient number.

The objective of this project is to build models which will predict the chances of re-admissions in patients whether a patient will be readmitted to the hospital or not. For the sake of our predictive analyses, we have built KNN model and also a logistic regression model.

To identify the best and the most significant features, we make use of Principle Component Analysis (PCA) technique and correlation matrix. Furthermore, we convert and pre-process the data by removing null values and changing categorical variables into dummy variables which can be fed into our logistic model as an input. Some of the categorical variables like “diag_1”, “diag_2”, “diag_3” although contain numerical values, however in reality represent the diagnosis code and converting them accurately would require some domain specific knowledge.

With our EDA we get insights into the age distribution, gender distribution as well as the readmission rate. In conclusion, we provide an overall comparison of methods based on their accuracy, sensitivity and specificity score.

Based on our predictive analysis, we obtain a higher accuracy from our KNN model which comes out to be around ~80%, higher than logistic regression accuracy which is around ~61%. Thus, we recommend the KNN model to the business users for consideration.

INTRODUCTION

The data set used in this project is a record of information about the inclusion and treatment of diabetics in the medical system. The information in the data set can be divided into three main categories. The first category is the patient's personal information. For examples: patient number, race, sex, weight, age. The second type of information is mainly about admissions. For example: type of admission, source of patient, number of days in hospital, type of discharge, number of outpatients, emergency, or hospital visits within a year after treatment, whether to be hospitalized again. The third type of information is about the patient's treatment. This includes the number of tests performed after admission, the primary diagnosis, secondary diagnosis, the number of diagnoses, whether prescription drugs have been used, whether there is a change in the type of drug, changes in the dose of the drug, and whether 24 important drugs have been used for treatment. The data set has character-type attributes and numeral attributes, so KNN can be used as the first method.

Overview of problem

Based on the data set described above, the project mainly addressed the question of whether the patient would be re-hospitalized based on the data of the diabetic in the medical system.

Overview of technique

K-nearest neighbor (KNN) is a widely used machine learning method. It is mainly used to solve the classification problem. This approach works by categorizing existing data and calculating the distance between new data and the surrounding data as new data enters the model. The new data belongs to the type of data that is closer together. When using the KNN method, there are three methods for calculating distance. Euclidean Distance, Manhattan Distance and Minkowski Distance. Their formula for calculating distance is shown in the figure below:

Euclidean distance

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan distance

$$\sum_{i=1}^{k} \left| x_i - y_i \right|$$

Minkowski distance

$$\left(\sum_{i=1}^{k} \left(\left| x_i - y_i \right| \right)^q \right)^{1/q}$$

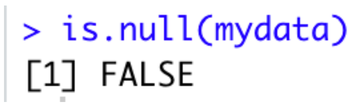
<u>KNN IMPLEMENTATION</u>

First, read the data from the local computer. This step can be done using the function read.csv(). Because the KNN method calculates the distance, it can only retain the numerical attributes of the data set. Subset out the data with function subset (). The code is as follows:

mydata<-subset(mydata, select = c(time_in_hospital, num_lab_procedures, num_procedures, number_outpatient, number_emergency, number_inpatient, diag_1, diag_2, diag_3, number_diagnoses, readmitted))

The last column ‘readadmitted’ is not a numerical variable, but it will be the dependent variable of the model, so it will also be extracted.

The next step is to clean the data. First, use the function is.null() to check if the data has a NULL value. The return result is shown in the figure.

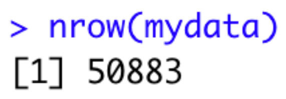


```
> is.null(mydata)
[1] FALSE
```

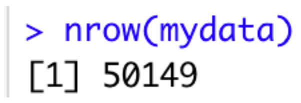
This means that there are no NULL values in the dataset. Then, the missing values in the data are deleted. The function used is na.omit(). There are no NULL values or missing values in the data set, but when browse through the data, there are many ‘?’ value that cannot be modeled and need to be removed. Use a for loop to iterate through each row of data, if there is a ‘?’ deletes the row. The code is as follows:

```
i<-1
for(i in 1:10){
  mydata<-mydata[!(mydata[,i]=='?'),]
}
```

The following two pictures, the first is the number of rows in the previous data. The second is the number of lines after cleaning up. It can be saw that the number of rows is going down.



```
> nrow(mydata)
[1] 50883
```



```
> nrow(mydata)
[1] 50149
```

What's more, diag_1, diag_2, diag_3 are factor variables which are needed to be transform into binary variables. According to the directory, code 250 represents diabetes. Thus, first round up the data for these three columns. And then I write a loop. Mark 1 when the data is 250 and 0 otherwise. The code is like following:

```
for(i in 1:1043){
  if(mydata[i,19]==250){
    mydata[i,19]<-1
  }else{
    mydata[i,19]<-0
  }
}
```

Export the cleaned data to the local computer for direct use next time. This step can be done with the function write.csv(). Next, the data set is divided into train data and test data. In this project, 70 percent of the data is train data and 30 percent is test data. The code is as follows:

```
train_data<-mydata%>%sample_frac(.7)
test_data<-mydata%>%sample_frac(.3)
```

The data used to build the KNN model must be between 0 and 1. To meet this requirement, define a function that normalizes the data form. The code of the function is as follows:

```
data_norm<-function(x) {
  ((x - min(x))/ (max(x)- min(x)))
}
```

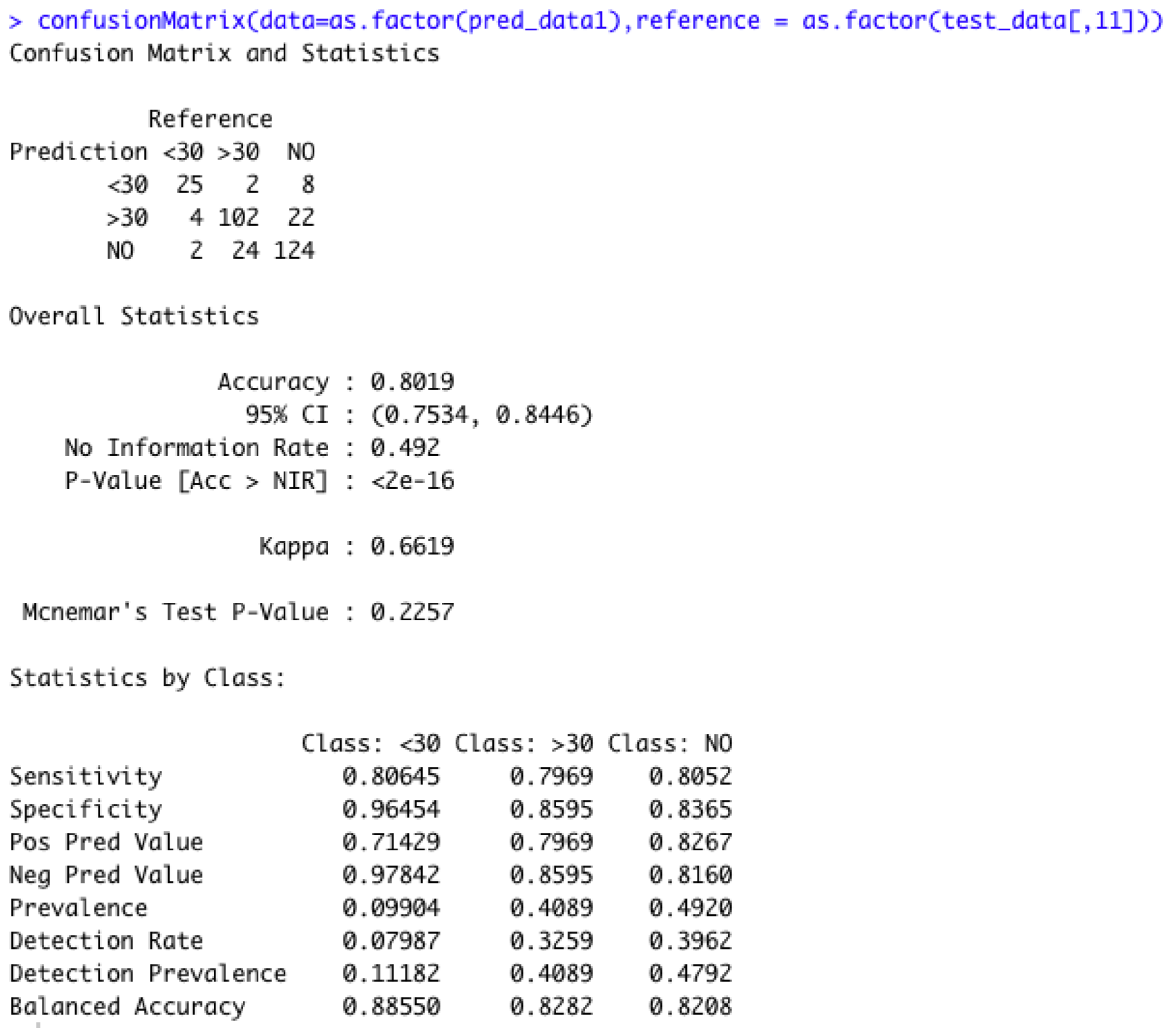
Then, the train data and test data are normalized respectively. Because the last column of the dataset, column 11, is the dependent variable y, it is not normalized. Next, the KNN model is established. This step requires the use of the package 'class'. When building the KNN model, two k values are selected. They are 1 and 3, respectively. Firstly, the prediction model is established with the train data, and then the independent variable of the test data is substituted into the model to generate the predicted value. This step is accomplished using the KNN () function. The code is as follows:

```
pred_data1<-knn(train_data_norm, test_data_norm, train_data[,11], k=1)
pred_data2<-knn(train_data_norm, test_data_norm, train_data[,11], k=3)
```

Next, use the function confusionMatrix () to calculate the confusion matrix for these two models. And the accuracy of the two models was calculated, that is, the proportion of the correct predicted value among all the predicted values.

Finally, in order to find the appropriate k value, the for loop is used to calculate the accuracy of the model when k value is between 1 and 28. For a more intuitive look, use the function plot() to plot the results.

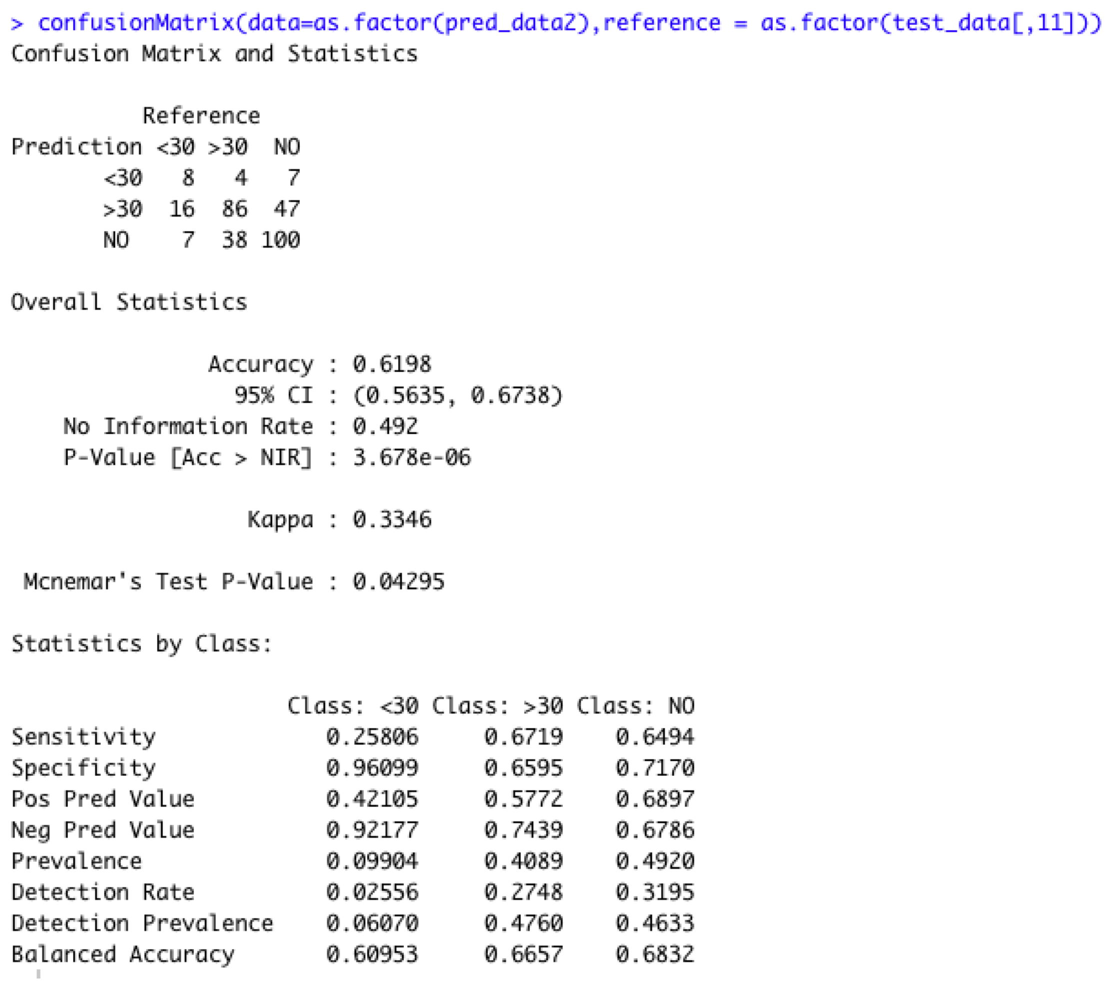
KNN - DATA ANALYSIS



```
> confusionMatrix(data=as.factor(pred_data1),reference = as.factor(test_data[,11]))
Confusion Matrix and Statistics

          Reference
Prediction <30 >30  NO
       <30  25   2   8
       >30   4 102  22
       NO    2  24 124

Overall Statistics

               Accuracy : 0.8019
                 95% CI : (0.7534, 0.8446)
    No Information Rate : 0.492
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.6619

 Mcnemar's Test P-Value : 0.2257

Statistics by Class:

                     Class: <30 Class: >30 Class: NO
Sensitivity             0.80645     0.7969    0.8052
Specificity             0.96454     0.8595    0.8365
Pos Pred Value          0.71429     0.7969    0.8267
Neg Pred Value          0.97842     0.8595    0.8160
Prevalence              0.09904     0.4089    0.4920
Detection Rate          0.07987     0.3259    0.3962
Detection Prevalence    0.11182     0.4089    0.4792
Balanced Accuracy       0.88550     0.8282    0.8208
```

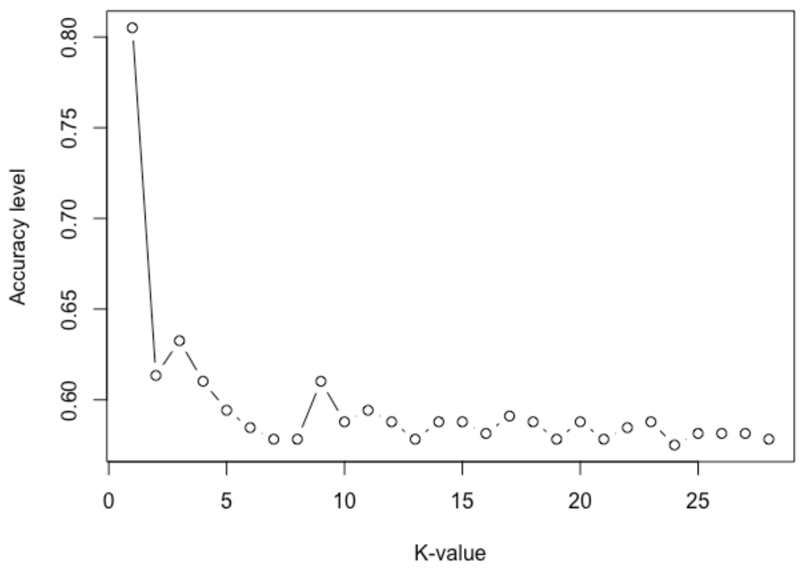
The picture above shows the confusion matrix of the model when k=1. The first line represents each unique value of the dependent variable in the test data. The first column represents all the unique values in the predicted result. '<30' represents a readmission with a stay of less than 30 days. '>30' means re-hospitalization and stay longer than 30 days. 'No' means No re-hospitalization. The prediction is correct only if the value of the dependent variable in the test data is equal to the predicted result. In other words, the numbers on the diagonal of the table represent the correct number of predictions. The number 2 in the second row and third column indicates that there are 2 cases where the value should have been '>30' but were predicted to be '<30'.

When k=3, the confusion matrix of the model is shown in the figure below:



```
> confusionMatrix(data=as.factor(pred_data2),reference = as.factor(test_data[,11]))
Confusion Matrix and Statistics

          Reference
Prediction <30 >30  NO
       <30   8   4   7
       >30  16  86  47
       NO    7  38 100

Overall Statistics

               Accuracy : 0.6198
                 95% CI : (0.5635, 0.6738)
    No Information Rate : 0.492
    P-Value [Acc > NIR] : 3.678e-06

                  Kappa : 0.3346

 Mcnemar's Test P-Value : 0.04295

Statistics by Class:

                     Class: <30 Class: >30 Class: NO
Sensitivity             0.25806     0.6719    0.6494
Specificity             0.96099     0.6595    0.7170
Pos Pred Value          0.42105     0.5772    0.6897
Neg Pred Value          0.92177     0.7439    0.6786
Prevalence              0.09904     0.4089    0.4920
Detection Rate          0.02556     0.2748    0.3195
Detection Prevalence    0.06070     0.4760    0.4633
Balanced Accuracy       0.60953     0.6657    0.6832
```

As can be seen from the two pictures above, the accuracy rate of the first model is 80.19%. The accuracy rate of the second model is 61.98%. In other words, the accuracy of the first model is higher than that of the second model, so the value of k should be 1. However, the accuracy of testing only two different k-value models does not necessarily lead to the most suitable k-value. Thus let's expand k to 1 to 28. Moreover, the trend of the accuracy of the model changing with the k value is shown more intuitively through the chart. The result is as follows:



As can be seen from the figure above, when k=1, the accuracy of the model is the highest. After that, the accuracy decreases dramatically. When k>3, the accuracy of the model fluctuates in a small scale with the increase of k value. In conclusion, 1 is the most appropriate value for k. When k=1, the accuracy of the model is 80%.

DISCUSSION

In this data set, whether or not hospitalization will occur is well suited as the dependent variable y in the classification problem. And the other characteristics are not independent of y. For example, the result of diagnosis, dosage of medication, composition and so on may affect the dependent variable. At the same time, the independent variables are independent of each other. This is very helpful for modeling and predictive analysis. In addition, there are many digital features in the data set. These columns provide conditions for using the KNN method.

The prediction model established by KNN method accurately predicted 80% of the data. In other words, it is feasible to use this model to predict whether a diabetic person will be hospitalized again.

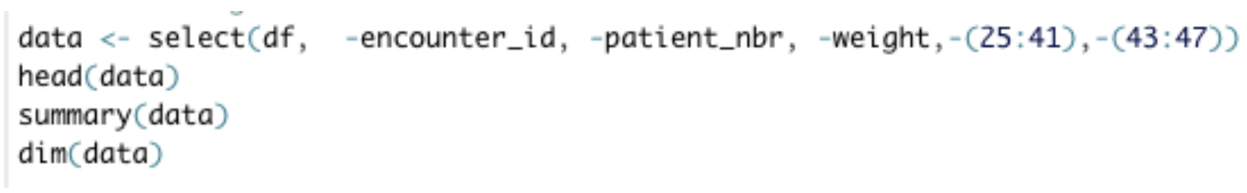
However, due to the limitations of KNN's algorithm and its applicability to digital variables, the accuracy of the model cannot be higher. In fact, character variables in the dataset can also have an impact on dependent variables. Inclusion of these variables should improve the accuracy of the model. Therefore, logistic regression models will be used for predictive analysis.

LOGISTIC REGRESSION

Hospital readmissions are among the challenges that hospitals are facing as of today. However, most of the readmissions are believed to be preventable. So we can use predictive models to assess the chance of readmission of a patient within a given period. With readmissions costing hospitals over 100 million dollars a year in the form of penalties, excessive usage of hospital resources etc. this part of the project aims at developing a logistic regression model that classifies whether the patient will be readmitted or not. With this knowledge on patients beforehand, hospitals can provide extended care to the patients so that they are less likely to be readmitted thus in turn save up on a lot of money.

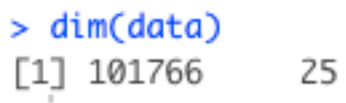
IMPLEMENTATION

We begin with data pre-processing, where we clean our dataset, perform feature reduction to make use of only those columns that are having an impact on our analysis and we remove all the columns with medicine dependency. Converting the large amount of non-numerical values into numerical values for the sake of further analyses.



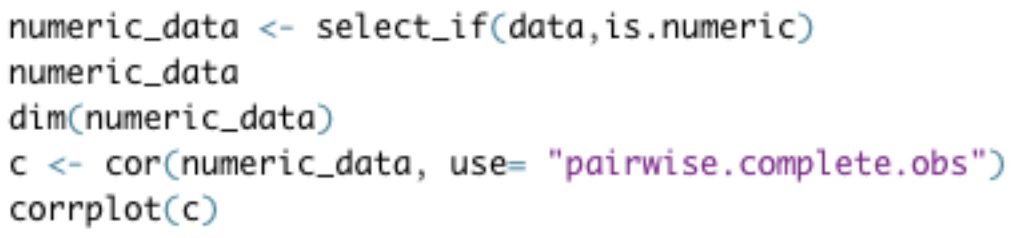
```
data <- select(df,  -encounter_id, -patient_nbr, -weight,-(25:41),-(43:47))
head(data)
summary(data)
dim(data)
```

As seen above, we exclude the columns which we won't be considering for our analysis and include only the relevant columns.



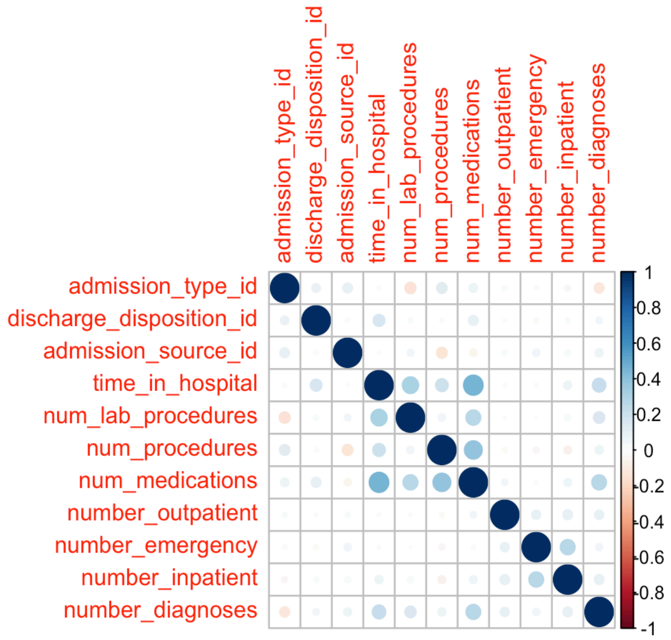
```
> dim(data)
[1] 101766     25
```

We also convert the categorical variables into numeric using as.numeric() function to obtain the co-relation between all the numeric features in our dataset.

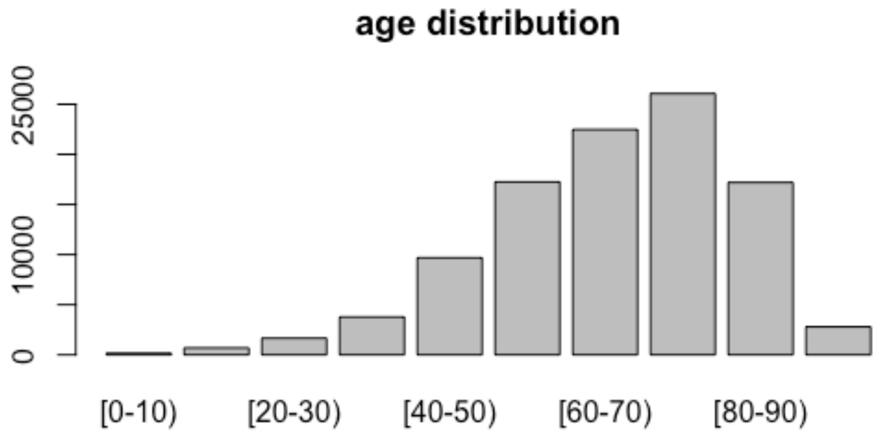


```
numeric_data <- select_if(data,is.numeric)
numeric_data
dim(numeric_data)
c <- cor(numeric_data, use= "pairwise.complete.obs")
corrplot(c)
```

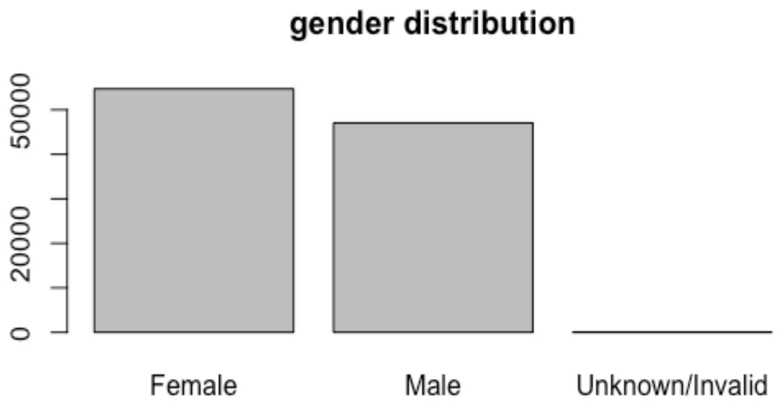
Correlation matrix results in 11 all numeric independent features which were significant and also their correlation with each other. From the matrix we can see that the number of lab procedures and number of medications had a decent correlation with the time spent by a patient in the hospital.



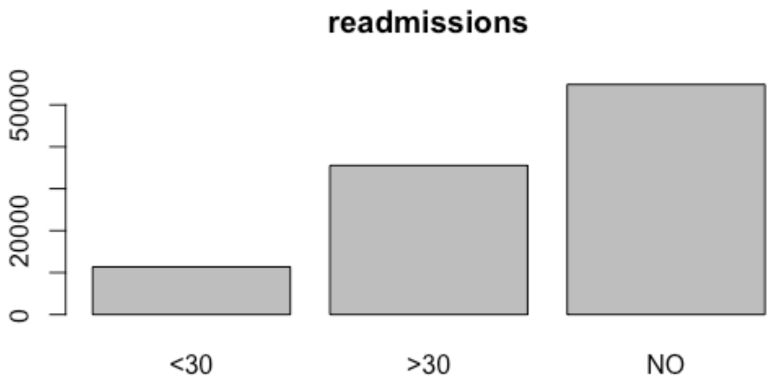
Then we come to EDA part to firstly obtain the **Age Distribution** as seen in the below bar chart. From the Age Distribution graph, we can conclude that the maximum number of patients admitted belonged to the 70-80 age group followed by the 60-70 age group.



Based on the **gender distribution** graph as seen below, we can see from the dataset that females accounted for ~55% of the hospital admissions as compared to males who accounted for a little less than 45%.



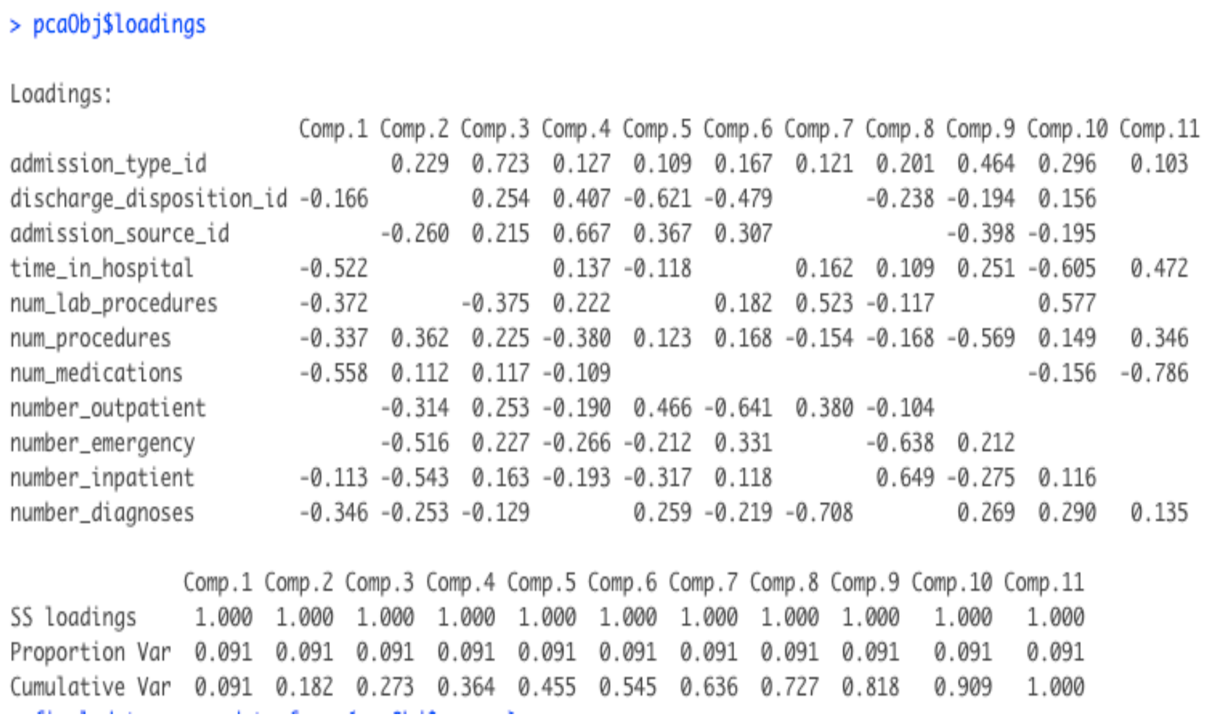
Based on our EDA, we can conclude that around 11% of the population is re-hospitalized and that too within 30 days.



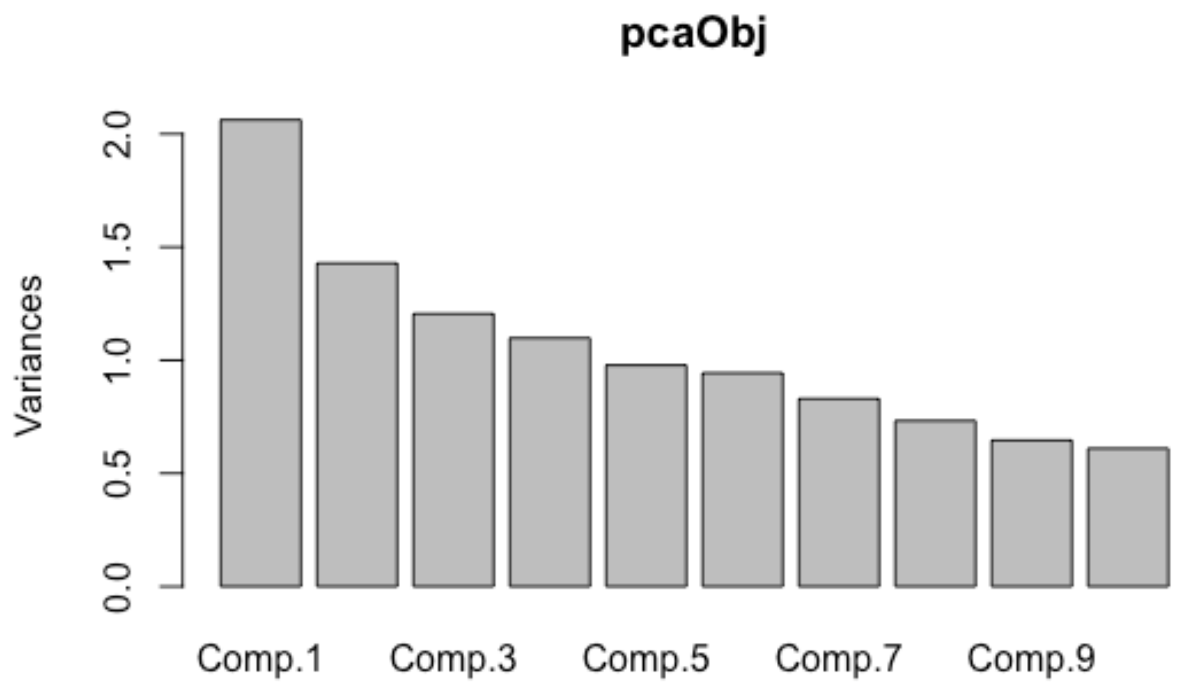
```r
#### Altering categorical variables
num_data <- data
num_data$diag_1 <- as.numeric(levels(num_data$diag_1)[num_data$diag_1])
num_data$diag_2 <- as.numeric(levels(num_data$diag_2)[num_data$diag_2])
num_data$diag_3 <- as.numeric(levels(num_data$diag_3)[num_data$diag_3])
```

Considering that “diag_1”, “diag_2” and “diag_3” are actually categorical variables which are representing a part of the ICD codes, we work towards converting them into numerical variables too.

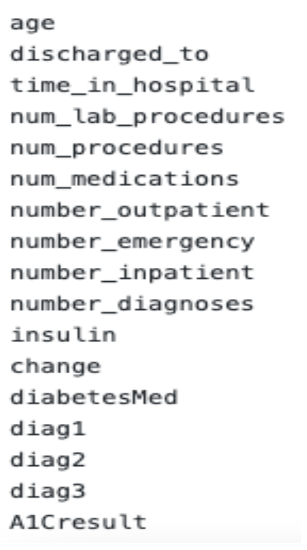
Then, in order to identify the most relevant features, we perform Principle Component Analysis (PCA) to obtain a list of the most relevant features.



```
> pcaObj$loadings

Loadings:
                         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9 Comp.10 Comp.11
admission_type_id                0.229  0.723  0.127  0.109  0.167  0.121  0.201  0.464   0.296   0.103
discharge_disposition_id -0.166         0.254  0.407 -0.621 -0.479        -0.238 -0.194   0.156
admission_source_id             -0.260  0.215  0.667  0.367  0.307               -0.398  -0.195
time_in_hospital         -0.522                0.137 -0.118         0.162  0.109  0.251  -0.605   0.472
num_lab_procedures       -0.372        -0.375  0.222         0.182  0.523 -0.117          0.577
num_procedures           -0.337  0.362  0.225 -0.380  0.123  0.168 -0.154 -0.168 -0.569   0.149   0.346
num_medications          -0.558  0.112  0.117 -0.109                                     -0.156  -0.786
number_outpatient               -0.314  0.253 -0.190  0.466 -0.641  0.380 -0.104
number_emergency                -0.516  0.227 -0.266 -0.212  0.331        -0.638  0.212
number_inpatient         -0.113 -0.543  0.163 -0.193 -0.317  0.118         0.649 -0.275   0.116
number_diagnoses         -0.346 -0.253 -0.129         0.259 -0.219 -0.708         0.269   0.290   0.135

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9 Comp.10 Comp.11
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000   1.000   1.000
Proportion Var  0.091  0.091  0.091  0.091  0.091  0.091  0.091  0.091  0.091   0.091   0.091
Cumulative Var  0.091  0.182  0.273  0.364  0.455  0.545  0.636  0.727  0.818   0.909   1.000
```



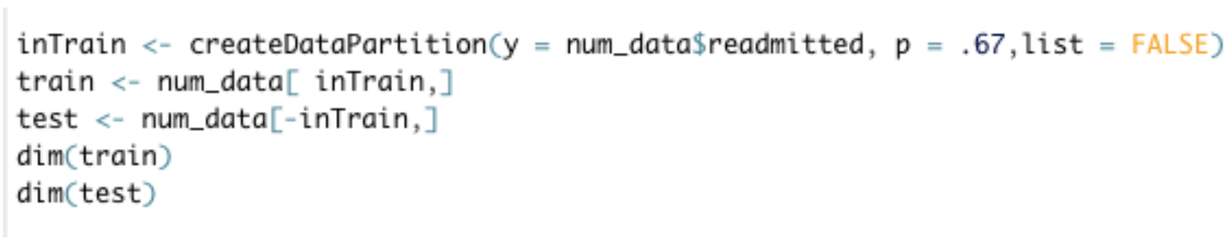
The above figures provide us with an understanding on the most important components / features of our dataset. Using this, we obtain a list of the most important features and how they're ranked:



```
age
discharged_to
time_in_hospital
num_lab_procedures
num_procedures
num_medications
number_outpatient
number_emergency
number_inpatient
number_diagnoses
insulin
change
diabetesMed
diag1
diag2
diag3
A1Cresult
```

DATA ANALYSIS

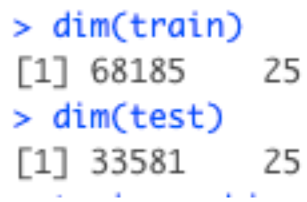
In this part, we split our dataset into training and testing dataset as seen below:



```
inTrain <- createDataPartition(y = num_data$readmitted, p = .67,list = FALSE)
train <- num_data[ inTrain,]
test <- num_data[-inTrain,]
dim(train)
dim(test)
```

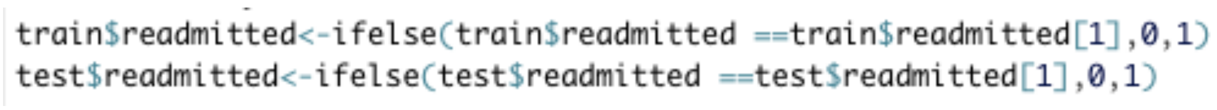
We split it initially into a train dataset so that we can apply and train our model on the dataset, and we also divide our dataset into test dataset so as to validate our data.

We split them up into 2 different datasets in the following ratio:



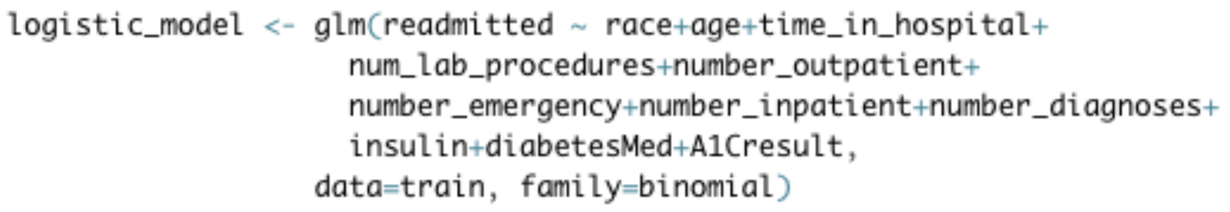
```
> dim(train)
[1] 68185    25
> dim(test)
[1] 33581    25
```

We further convert the features into a type that can be easily interpretated by our logistic model.



```
train$readmitted<-ifelse(train$readmitted ==train$readmitted[1],0,1)
test$readmitted<-ifelse(test$readmitted ==test$readmitted[1],0,1)
```

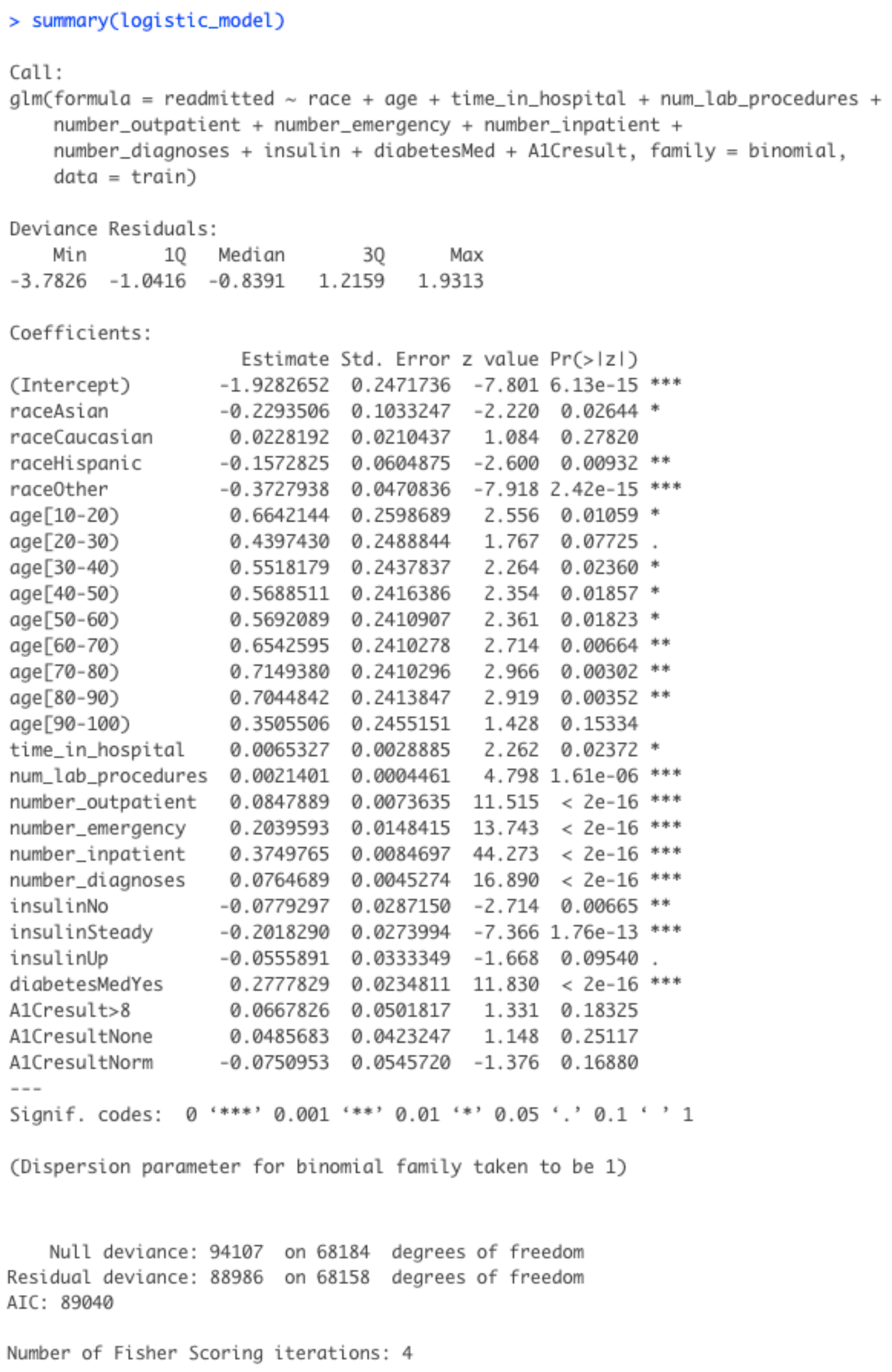
Now, we perform our logistic regression technique to test how accurately our model is classifying the diabetic patients who could be re-admitted in the hospital. To perform our analysis, we use the glm model in R as shown:



```
logistic_model <- glm(readmitted ~ race+age+time_in_hospital+
                        num_lab_procedures+number_outpatient+
                        number_emergency+number_inpatient+number_diagnoses+
                        insulin+diabetesMed+A1Cresult,
                      data=train, family=binomial)
```

We consider only those features here which give us better accuracy and we get remove the features from our model which are not being classified appropriately. Some of the features like “diag_1”, “diag_2”, “diag_3” are being excluded although they are significant to the analysis. It is mainly due to issues like class imbalance. Based on the ICD-9 score, it is difficult for our model to accurately interpret the diagnosis codes and generate higher accuracy.

We use the generalized logistic regression technique because we think its appropriate when the dependent variable is categorical and binary. The goal of the logistic regression is to find the best fitting model to describe the dependent variable based on a set of independent variables. Using a logistic model here we get a probability score that reflects the probability of patient readmission.

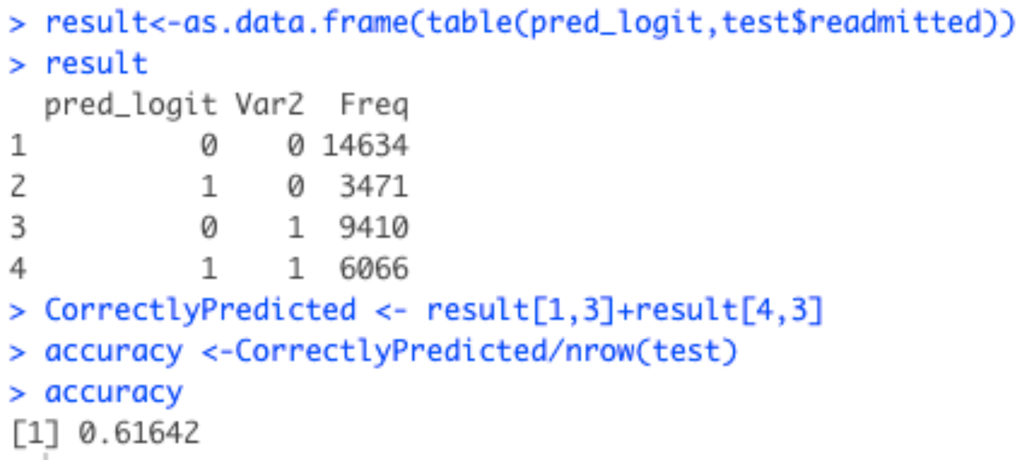


```
> summary(logistic_model)

Call:
glm(formula = readmitted ~ race + age + time_in_hospital + num_lab_procedures +
    number_outpatient + number_emergency + number_inpatient +
    number_diagnoses + insulin + diabetesMed + A1Cresult, family = binomial,
    data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.7826  -1.0416  -0.8391   1.2159   1.9313

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)        -1.9282652  0.2471736  -7.801 6.13e-15 ***
raceAsian          -0.2293506  0.1033247  -2.220  0.02644 *
raceCaucasian       0.0228192  0.0210437   1.084  0.27820
raceHispanic       -0.1572825  0.0604875  -2.600  0.00932 **
raceOther          -0.3727938  0.0470836  -7.918 2.42e-15 ***
age[10-20)          0.6642144  0.2598689   2.556  0.01059 *
age[20-30)          0.4397430  0.2488844   1.767  0.07725 .
age[30-40)          0.5518179  0.2437837   2.264  0.02360 *
age[40-50)          0.5688511  0.2416386   2.354  0.01857 *
age[50-60)          0.5692089  0.2410907   2.361  0.01823 *
age[60-70)          0.6542595  0.2410278   2.714  0.00664 **
age[70-80)          0.7149380  0.2410296   2.966  0.00302 **
age[80-90)          0.7044842  0.2413847   2.919  0.00352 **
age[90-100)         0.3505506  0.2455151   1.428  0.15334
time_in_hospital    0.0065327  0.0028885   2.262  0.02372 *
num_lab_procedures  0.0021401  0.0004461   4.798 1.61e-06 ***
number_outpatient   0.0847889  0.0073635  11.515  < 2e-16 ***
number_emergency    0.2039593  0.0148415  13.743  < 2e-16 ***
number_inpatient    0.3749765  0.0084697  44.273  < 2e-16 ***
number_diagnoses    0.0764689  0.0045274  16.890  < 2e-16 ***
insulinNo          -0.0779297  0.0287150  -2.714  0.00665 **
insulinSteady      -0.2018290  0.0273994  -7.366 1.76e-13 ***
insulinUp          -0.0555891  0.0333349  -1.668  0.09540 .
diabetesMedYes      0.2777829  0.0234811  11.830  < 2e-16 ***
A1Cresult>8         0.0667826  0.0501817   1.331  0.18325
A1CresultNone       0.0485683  0.0423247   1.148  0.25117
A1CresultNorm      -0.0750953  0.0545720  -1.376  0.16880
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)


    Null deviance: 94107  on 68184  degrees of freedom
Residual deviance: 88986  on 68158  degrees of freedom
AIC: 89040

Number of Fisher Scoring iterations: 4
```

Above is the output that we get on running our GLM model.

Based on the results obtained from our GLM model, we perform an accuracy test to evaluate how well our model is performing and how accurately it is classifying the patients that could be re-admitted to the hospital.

To predict the accuracy test, we run our logistic model as shown below using the predict function:

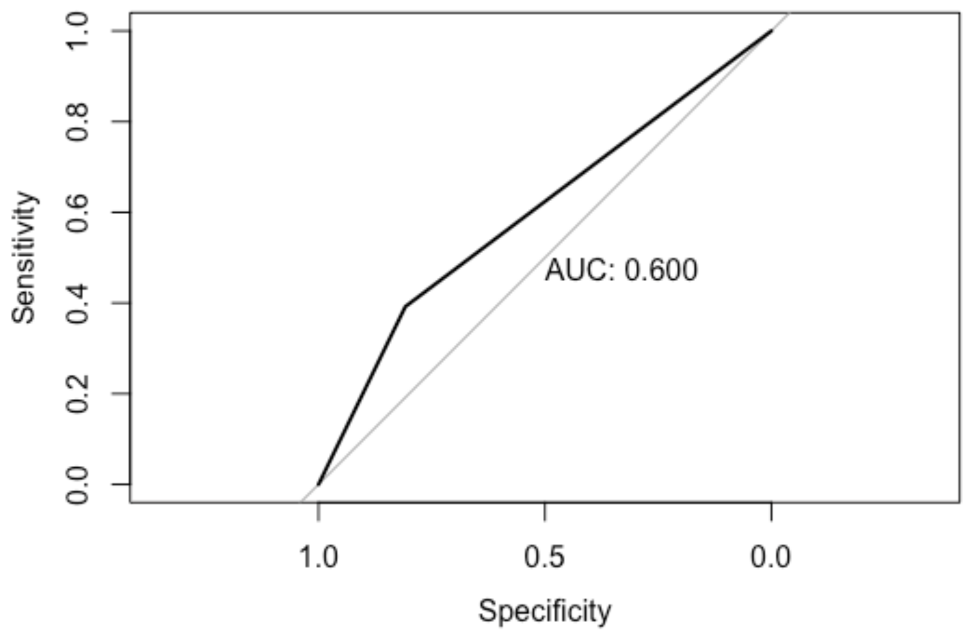


```
> result<-as.data.frame(table(pred_logit,test$readmitted))
> result
  pred_logit Var2  Freq
1          0    0 14634
2          1    0  3471
3          0    1  9410
4          1    1  6066
> CorrectlyPredicted <- result[1,3]+result[4,3]
> accuracy <-CorrectlyPredicted/nrow(test)
> accuracy
[1] 0.61642
```

From the above output, we obtain our confusion matrix and see how many of the values are being accurately classified in our data. We can see that our model is predict re-admission rate with an accuracy of ~62%. Using this model, we can conclude with ~62% accuracy/certainity whether a patient will be re-admitted or not.

To further evaluate our classification model's performance, we use the AUC-ROC curve. From the AUC score, we can see that the model is probably having difficulties in distinguishing between the classes hence only a score of 0.6 as seen below:

```
library(pROC)
test_prob = predict(logistic_model, newdata = test, type = "response")
test_roc = roc(test$readmitted ~ pred_logit, plot = TRUE, print.auc = TRUE)
```



While Sensitivity (true positive rate) measure is used to determine the proportion of actual positive cases, which got predicted correctly, Specificity (true negative rate) measure is used to determine the proportion of actual negative cases, which got predicted correctly.

DISCUSSION & COMPARISON

CLASSIFIER	ACCURACY	SENSITIVITY	SPECIFICITY	RUN TIME RATING
KNN	80.19%	80.29%	88.68%	2
Logistic Regression	61.87%	63.60%	61.16%	1

Here we are comparing the results our 2 techniques. From the table we can see that although ideally Logistic Regression should have had a higher accuracy score than the KNN algorithm, however, logistic regression did not perform as well as it should have with its accuracy score. Talking about the Run time rating, logistic regression took a slightly lesser time to run than the KNN model, however, with 80%+ accuracy, the KNN model best describes our dataset and can predict with 80%+ accuracy whether a patient will get re-admitted or not and is the one which we would recommend for predicting the likelihood of readmission. With ~80% accuracy – the KNN algorithm best describes our data and is the most fit to be predicting the likelihood of patient readmission.

The low accuracy obtained from our logistic model could be largely down to the exclusion of "diag_1", "diag_2", "diag_3" from our GLM model. There were also issues of class imbalance. In the future, we hope to use the diagnosis codes and incorporate the same into our code. Using the diagnosis codes, i.e. the first time the patient was diagnosed and with what issue, the second time and with what issue and so on, we are certain to obtain a higher accuracy of from results. Furthermore, we aim to use other techniques like SVM which is a discriminative classifier defined by separating hyperplane. It is a supervised machine learning algorithm mainly used for classification problems. Using the "e1071" library, we can obtain the SVM model. SVMs also take a lot less time to run.

CONCLUSION

1 Enzyme specific dependency is not considered in the analysis.
2 Majority classes were being overrepresented in predictions.
3 A significant proportion of the features were categorical.
4 KNN took a long time to run, hence improving model performance and accuracy would be a part of future work.

5 Based on the comparison of the accuracy of the two models, KNN model is suggested to solve the classification problem.

6 There are chances of obtaining an even higher accuracy for predicting the likelihood of patients being readmitted using other predictive modelling techniques like Random Forest, SVM, Naïve Bayes, Neural Networks.

REFERENCE

[1] edureka! (Dec 18, 2018). 'KNN Algorithm Using R | KNN Algorithm Example | Data Science Training | Edureka' Retrieved on Nov 12, 2020 from: https://www.youtube.com/watch?v=XSoau_q0kz8

[2] 'How to choose the value of K in knn algorithm'. Retrieved on Nov 12, 2020 from: https://discuss.analyticsvidhya.com/t/how-to-choose-the-value-of-k-in-knn-algorithm/2606

[3] James Le, (2018). Logistic Regression in R. Retrieved from, https://www.datacamp.com/community/tutorials/logistic-regression-R

[4] USING LOGISTIC REGRESSION TO MODEL AND PREDICT CATEGORICAL VALUES. Retrieved from, http://rstudio-pubs-static.s3.amazonaws.com/74431_8cbd662559f6451f9cd411545f28107f.html