An approach to reduce Readmission Rates

Sampath Kumar Gunasekaran, Nitish Rangarajan, Yuvaraj Sundarrajan, Ajay Kumar Prathap, Shrutha Kashyap

Abstract

A managed healthcare organization believes they suffer from over $30 million in preventable losses annually due to the readmission of patients who are discharged from the hospital too soon. However, keeping all patients in the hospital longer is costly and inconvenient to patients. The analysis of this large clinical database was undertaken to provide an assessment and to find future directions which might lead to improvements in patient safety and reduce the readmission rate. The objective for this project was to develop a model to reduce the readmission rate. The primary focus is to reduce the readmission rate of diabetic patients. We were provided with a dataset(10kdiabetes.csv) containing 52 features of patient records. We used the given dataset and applied chi-square analysis and built classification models based on machine learning algorithms to predict the most essential factors influencing the patient re-admission rate. Additionally, we proposed and analyzed other tasks that can strengthen the readmission risk model.

Keywords: Readmission rate, classification models, HbA1C

1. Introduction

The study used the Health Facts database from Cerner Corporation, Kansas City, MO, a national data warehouse which collects comprehensive clinical records across hospitals in the United States. The database has data systematically acquired from participating institutions electronic medical records and this has encounter data (emergency, outpatient, and inpatient), demographics (age, sex, and race), diagnoses and in-hospital procedures formed by ICD-9-CM codes, laboratory data, pharmacy data, in-hospital mortality, and hospital characteristics. The present analysis of this database was done to analyze historical patterns of diabetes care in patients with diabetes admitted to a US hospital and to get an idea about the future directions which might lead to advancements in patient safety and reduce readmission rate. We also examined the use of HbA1c as a marker of attention to diabetes care in a large number of individuals identified as having a diagnosis of diabetes mellitus. We hypothesized that measurement of HbA1c is associated with a reduction in readmission rates in individuals admitted to the hospital. We used other factors like insulin and change in medication but HbA1c was more related to the readmission than the other factors.

Literature Review

In Strack et al.'s (2014) article reviewing studies on predicting the readmission rate on the same clinical database, it was found that HbA1c with diabetes mellitus is a useful predictor of readmission rates which proved valuable to reduce readmission rates and costs of individuals diagnosed with diabetes mellitus. The analysis showed that the readmission varied in patients where Hba1c was checked in the context of a primary diabetes diagnosis, on comparing to those with a primary circulatory disorder. While readmission rates were high for patients with circulatory diagnoses, readmission rates for patients with diabetes was related with the need to test for HbA1c, rather than the values of the HbA1c result.

2. Data Preprocessing

a. Data Description

The dataset consisted of 52 features and over 10,000 records originally. Out of the 52 features over 17 were categorical variables, and the rest of the features were numerical.

b. Data Cleaning

The dataset originally had over 10000 records. The encounters that resulted in death of the patient or discharge to a hospice were removed to avoid biasing on the dataset. After then the dataset was reduced to 9772 records. We removed the inconsistencies in the data on selecting only the required attributes for the analysis. We ran a set of rules (refer appendix) to assess whether the data had any noise or inconsistencies.

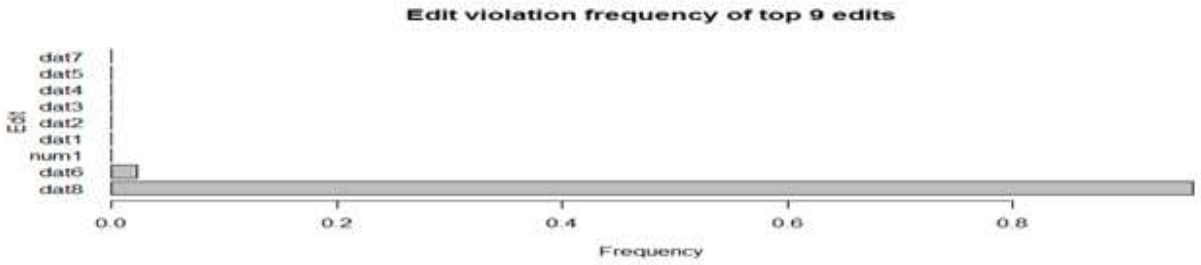

Figure 1: Edit Violation Frequency

After running these rules, we identified that weight attribute(dat8) had many missing values and it was not considered for further processing. Additionally, we removed null values from all the attributes.

3. Descriptive Statistics

Descriptive statistics are used to describe the basic features of the data in a study. They provide simple summary about the sample and the measures to visualize what the data shows. If the variable is categorical, we get the Frequency and Proportion. If the variable is numerical, we get the Mean, Median, Min and Max. On looking at the A1Cresult, we can see that there were 8167 patients who didn't opt for a HbA1c test and there were 7335 patients who took diabetes medications. The number of people who were readmitted were 3962 out of 9772 patient records.

Initial Visualizations

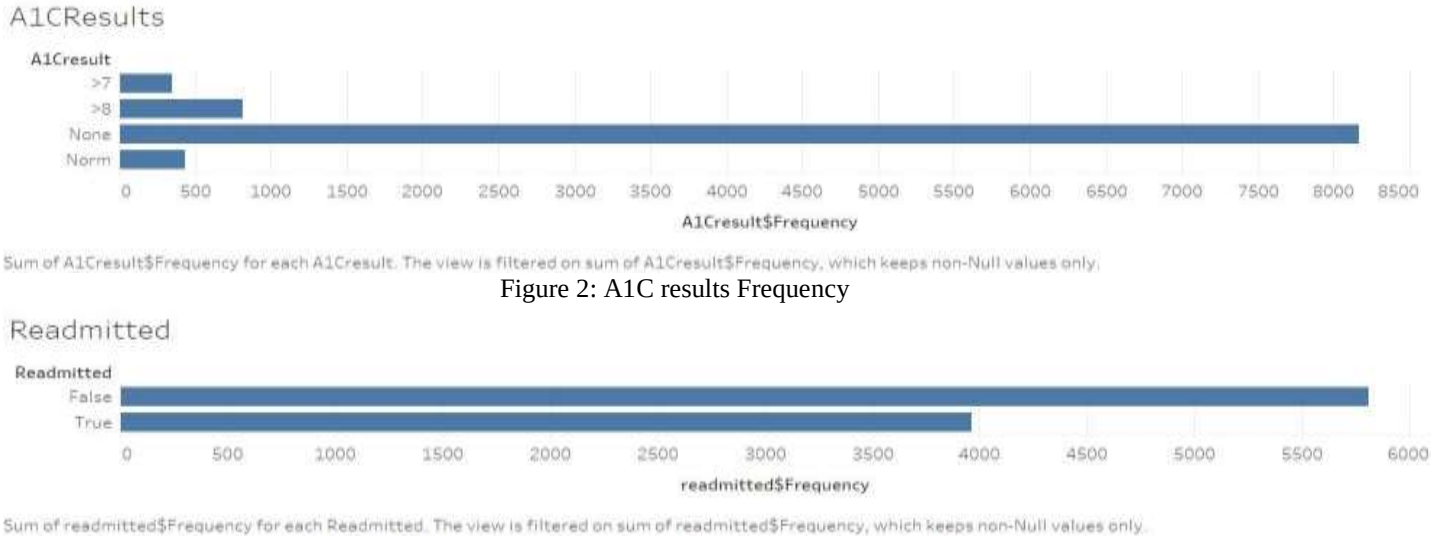

Figure 2: A1C results Frequency

Figure 3: Readmitted Count

4. Main Task

Problem Statement: An approach to predicting the hospital readmission rate.
Techniques: Correlation Analysis, PCA, Classification Model building,
Clustering Analysis, Pattern Mining
Language used: R
Visualization Tools: Tableau

1. Correlation Analysis

Since most of the variables were categorical, we chose to run a chi-square test using Xtabs for the correlation analysis. The analysis yielded four categorical variables that readmission has strong correlation with. They are

- Race
- Age
- A1Cresult
- Change in Medication

The reason why we decide to choose these four factors is because the p-value is less than 0.05 and the Chi-square value is considerable. Other factors didn't have a lesser p-value.

Chi-Square

Factors	
Age	100.81
Change in Medication	22.30
HbA1C	8.84
Insulin	34.07
Race	58.37

Sum of Chi-square value broken down by Factors.

p-Values

Factors	
Age	0.0000000000000011
Change in Medication	0.0000023270000000
HbA1C	0.0315600000000000
Insulin	0.0000001916000000
Race	0.0000000000264400

Sum of P-Value broken down by Factors.

Figure 4: Chi-Square & P-values

From figure 4, we noticed that all the above variables have lesser p values which signifies that they have high correlation with readmission factor compared to other variables.

2. Principal Component Analysis

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components (or sometimes, principal modes of variation). We performed PCA on our dataset to map similar attributes together and extract important features which proves to be useful for predicting readmission rate.

PCA test was run on the following three components since they had high correlation with readmission

1) A1Cresult
2) max_glu_serum
3) insulin

```
                item   PC1   PC2   PC3   h2    u2 com
A1Cresult          1  0.78                0.66 0.340 1.2
insulin            3  0.69  0.46          0.69 0.310 1.7
max_glu_serum      2        0.83 -0.34    0.82 0.177 1.4
readmitted         4                0.94 0.97 0.027 1.2

                         PC1  PC2  PC3
SS loadings             1.10 1.04 1.00
Proportion Var          0.28 0.26 0.25
Cumulative Var          0.28 0.54 0.79
Proportion Explained    0.35 0.33 0.32
Cumulative Proportion   0.35 0.68 1.00
```

Figure 5: Results of PCA

A Scree plot was used to identify the point of inflection, and it gave the following results:

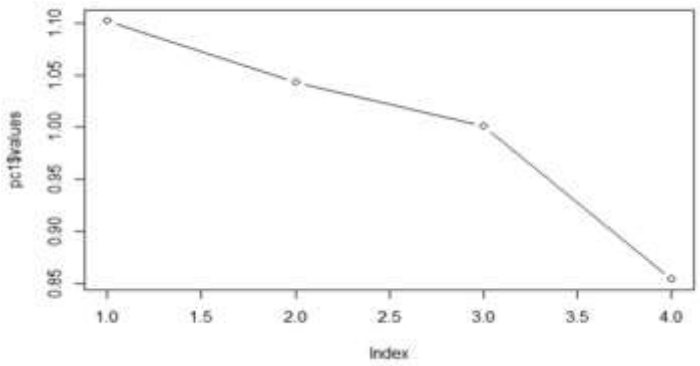

Figure 6: Scree Plot of PCA results

As we can see from the above scree plot, component 1(A1C result and insulin) explains 37% of the total variance. Thus these 2 attributes A1C result and insulin has great influence on readmission rates.

3. Pattern Analysis

We used Apriori, a candidate generation and test approach for mining frequent patterns to predict readmission rate. We tried mining patterns for all the variables, the associative rules we got were not satisfactory. Then we figured out the appropriate variables by taking into consideration the highly correlated variables for which the readmission rate is high. We used the features below for pattern mining.

Feature Extraction

race, gender, age, A1Cresult, insulin, change, diabetesMed

Training the model

We factored all the variables and then changed it to transactions. We ran Apriori, with support = 0.1 and confidence = 0.6 and sorted by lift. Initially, we got 42 associative rules. We pruned the redundant rules and drilled down to 12 associative rules. Finally, we plotted the association rules.

Results:

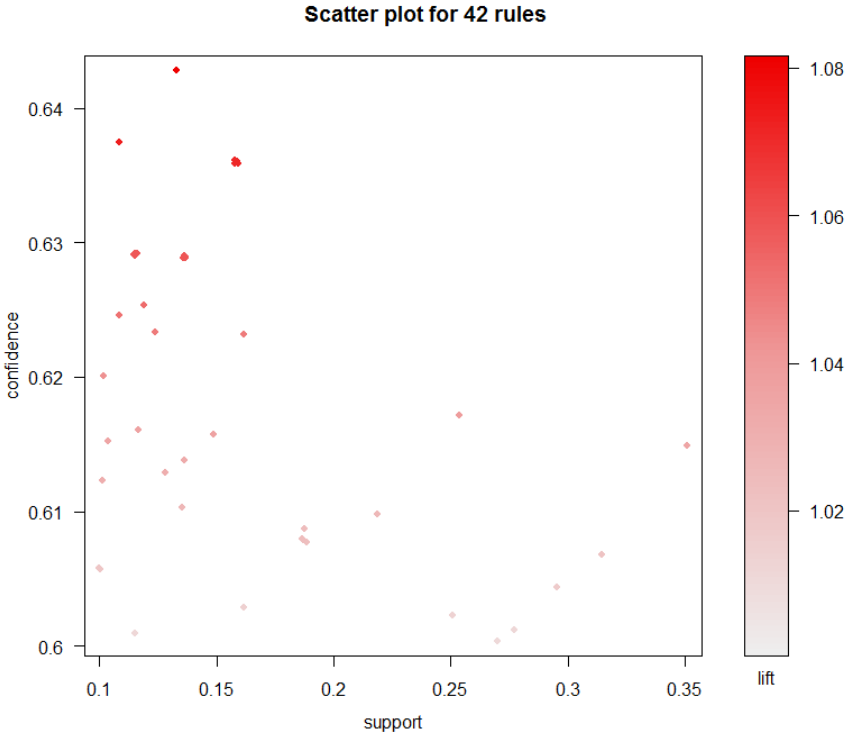

Figure 7: Scatter Plot for 42 rules

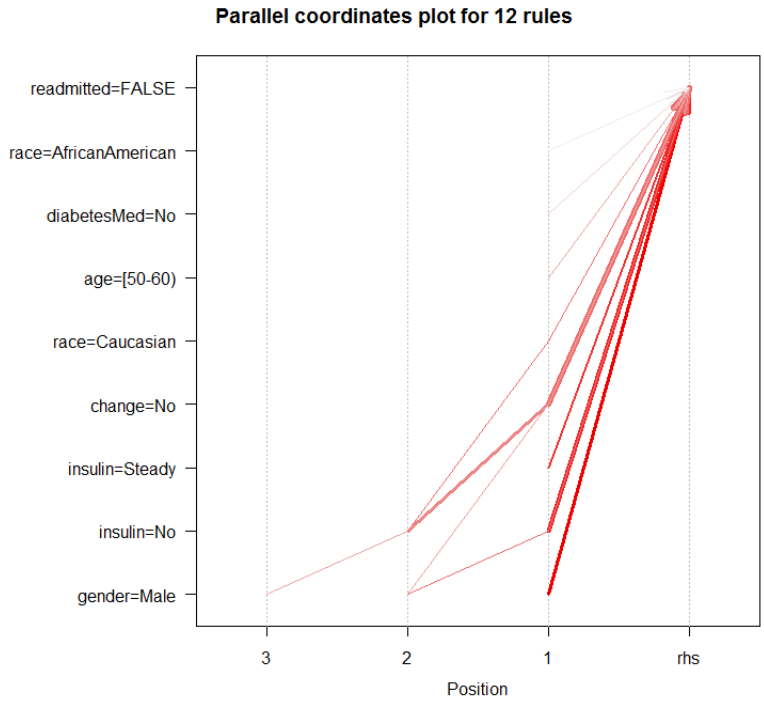

Figure 8: Parallel Coordinates for 12 rules

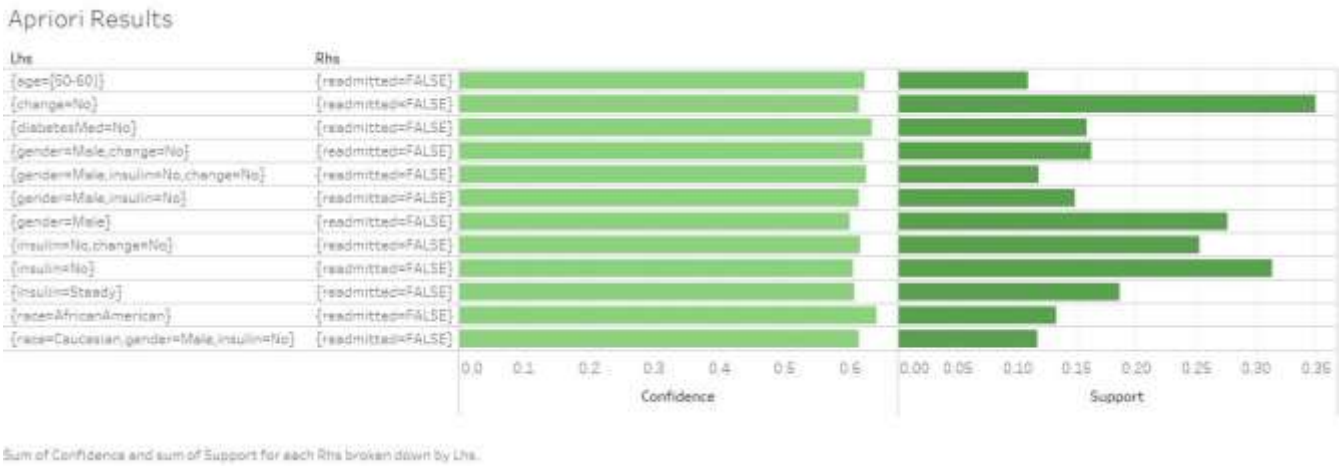

Figure 9: 12 Associative rules for readmission

When there is no change in medication, the support for not getting readmitted is slightly higher than the other rules.

d) Classification Models

The following classification models are developed to predict categorical class labels which classifies data based on current set and uses it in classifying future data.

Performance Evaluation

We used the following factors to evaluate the accuracy of the models

- Error Probability
- Accuracy

We worked on the following classification models to classify whether the patient is readmitted or not for the future patient records.

- Naïve Bayes
- Decision Tree
- Random Forest
- Neural Networks
- KNN
- Logistic Regression

Data Specifications:

- Training data: 80% of the data set
- Test data: 20% of the data set
- Initial Predictors: race, gender, age, admission_type_id, discharge_disposition_id, admission_source_id, time_in_hospital, medical_speciality, num_lab_procedures, num_procedures, num_medications, num_outpatient, num_emergency, num_inpatient, diag_1,diag_2, diag_3,number_diagnosis,max_glu_serum,

a1cresult,insulin,change, diabetes_medication, readmitted

- Outcome: readmitted

Model Name: Naïve Bayes

Naïve bayes performs probabilistic prediction

- Bayesian Rule

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid c)P(c)}{P(\mathbf{x})} \qquad Posterior = \frac{Likelihood \times Prior}{Evidence}$$

Discriminative Generative

Features Extraction

race, gender, age, change in medication, glucose level

Training the model

We performed two class Naïve Bayes

probability of class1 (readmission=true) given below features

probability of class2 (readmission=false) given below features

We started to build the model, considering all the variables, it didn't give satisfactory results. After trying out various features, we extracted the below significant features for building Naïve Bayes classification.

Results

Since Naïve Bayes is a linear classifier it gives a clear decision boundary between two classes (Class1: readmission - true, Class2: readmission - false).

Accuracy: 66.25

Model Name: Decision Tree

Decision tree is constructed in a top down recursive manner. Features are selected based on information gain.

Features Extraction

race, age, gender, change, insulin, A1Cresult

Training the model

Attributes are chosen based on the information gain and the Splitting point is chosen using Gini Index

Results

Model of decisions and their possible consequences are represented in a tree diagram.

Accuracy: 60.56

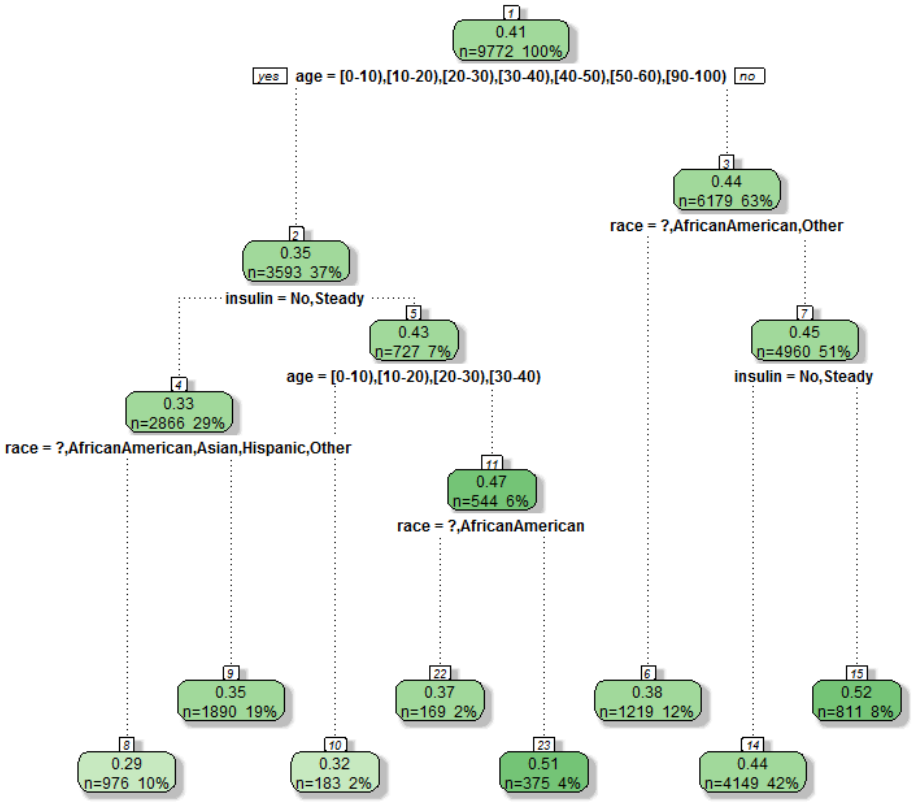

Figure 10: Decision Tree for Readmission

Model Name: Random Forest

In random forests, there is no need for a separate test set to get an unbiased estimate of the test set error. It is estimated internally during the run.

Features Extraction

race, gender, age, A1Cresult, insulin, change, diabetesMed

Training the model

Attributes are chosen based on the information gain and the Splitting point is chosen using Gini Index.

Results

MeanDecreaseGini for each individual variable over all trees in the forest gives a variable importance. It gave an estimate of what variables are important in the classification. From this we extracted age, race, insulin, A1C result as important features to be considered while predicting readmission rate.

Accuracy: 59.33

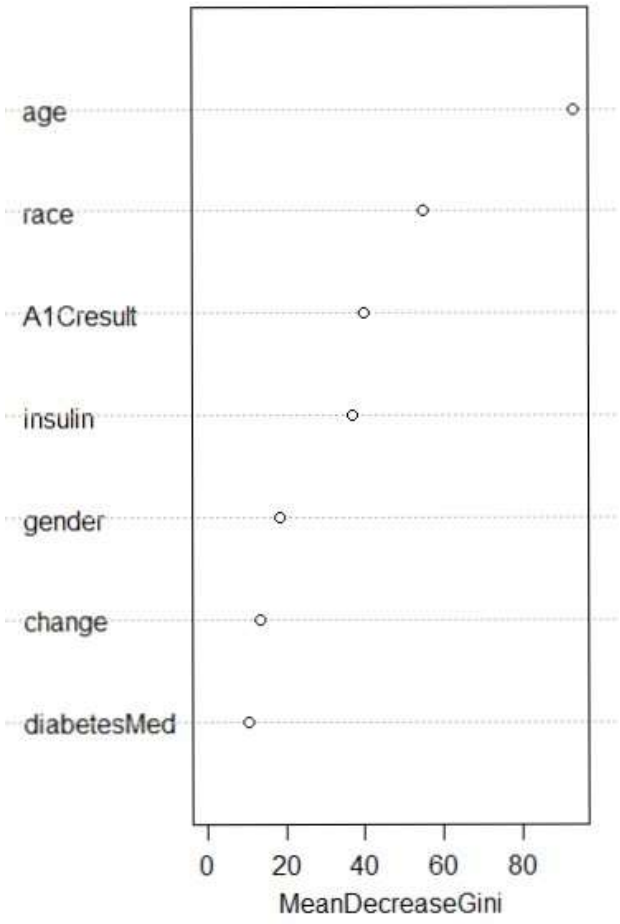

Figure 11: Mean Decrease Gini

Model Name: Neural Networks

Neutral is a back-propagation technique. It is a set of connected input and output units where each connection has a weight associated with it.

Features Extraction

"race","gender","age","A1Cresult","insulin","change","diabetesMed

Training the model

During the training phase, the network learns by adjusting the weights, to be able to predict the correct. We factored the variables initially, and set hidden levels to 5 and threshold = 0.1

Results

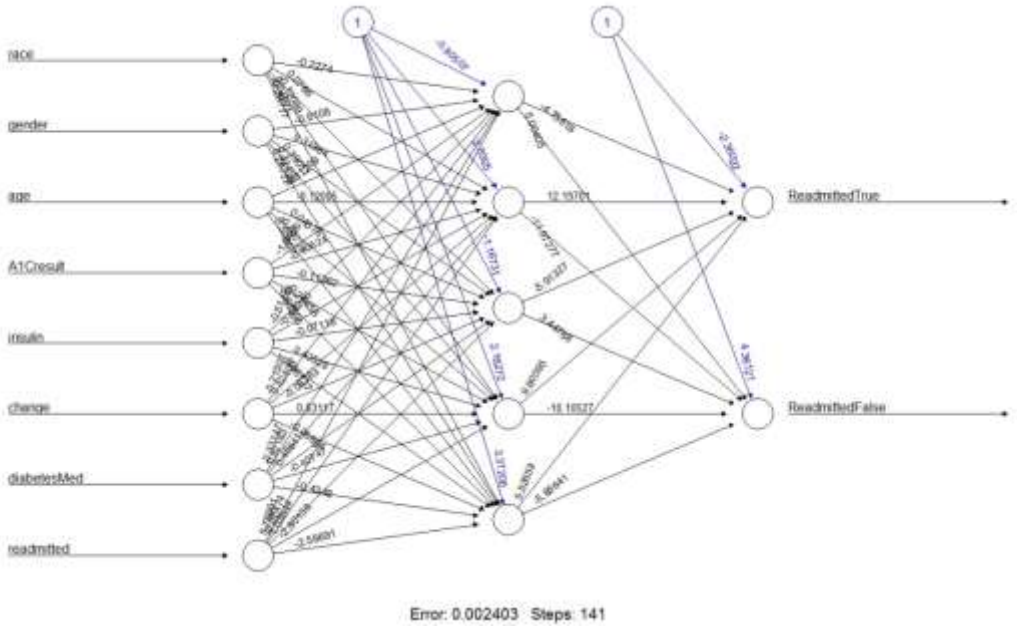

Figure 12: Neural Network for readmission

Accuracy: 63.19

Model Name: KNN

Lazy Learning: less time in training but more time in predicting

Features Extraction

Race, gender, age, A1Cresult, insulin, change, diabetesMed

Training the model

Choosing the value of k is important in KNN. We tried to train the data with different k values, to get the accurate value of K.

Results

Accuracy at different Ks

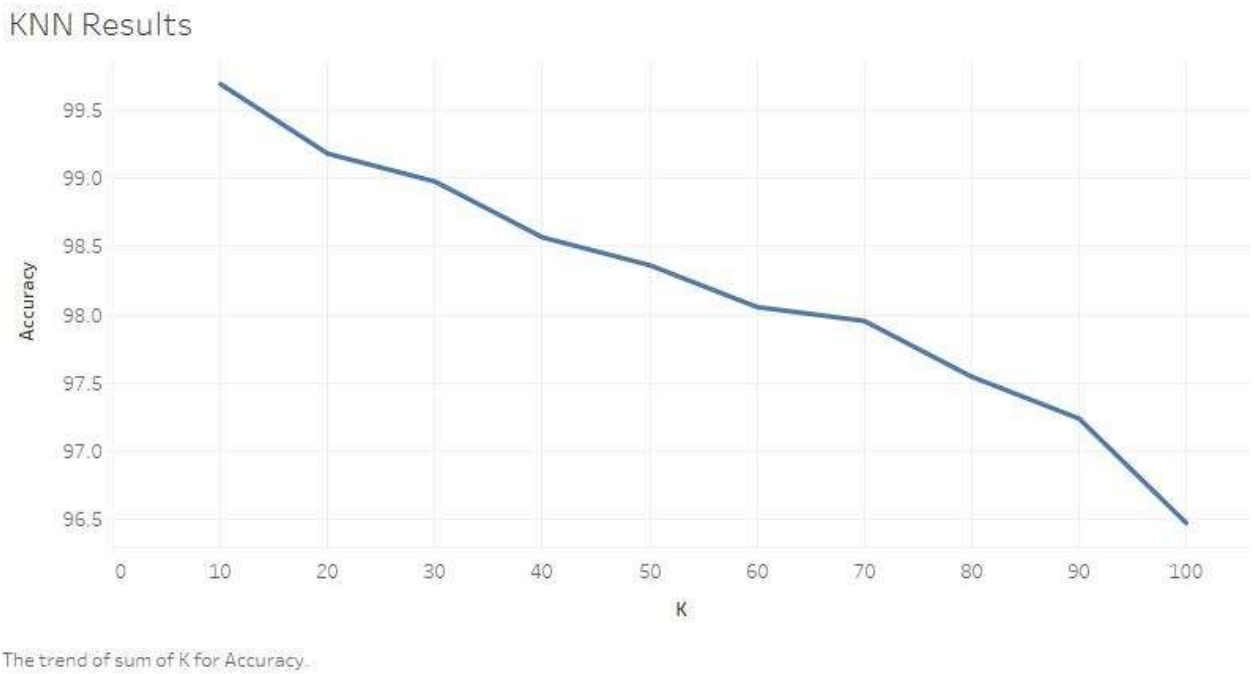

Figure 13: KNN Results at different K

Model Name: Logistic Regression

The outcome variable of our dataset is a nominal variable and thus logistic regression is used to build the model.

Feature Extraction

Race, age, time_in_hospital, a1c result, change_in_medication, diabetes, diagnosis_1,diagnosis_2,diagnosis_3

Results

Thus the above features are the significant ones we extracted from this model

Accuracy: 64.23

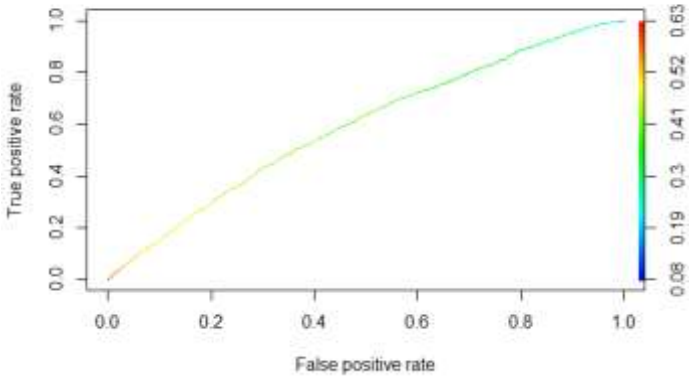

Figure 14: ROC plot for Logistic regression

Figure 15: Accuracy for various classification models

K-modes Clustering

Since the dataset also contains categorical variables, we used K-Modes instead of K-Means to cluster the similar set of patients. Following are the four different clusters we created.

Cluster Name	Readmitted	A1C result
Cluster 1	False	>7,>8,Norm
Cluster 2	False	None
Cluster 3	True	>7,>8,Norm
Cluster 4	True	None

Table 1: Summary of clusters

Results

We could infer from the below bar chart, the readmission factor is high for patients who didn't opt for A1C test (Cluster 4). Hence, the decision to opt for A1C test influences the readmission factor more than the case where the A1C test results are available.

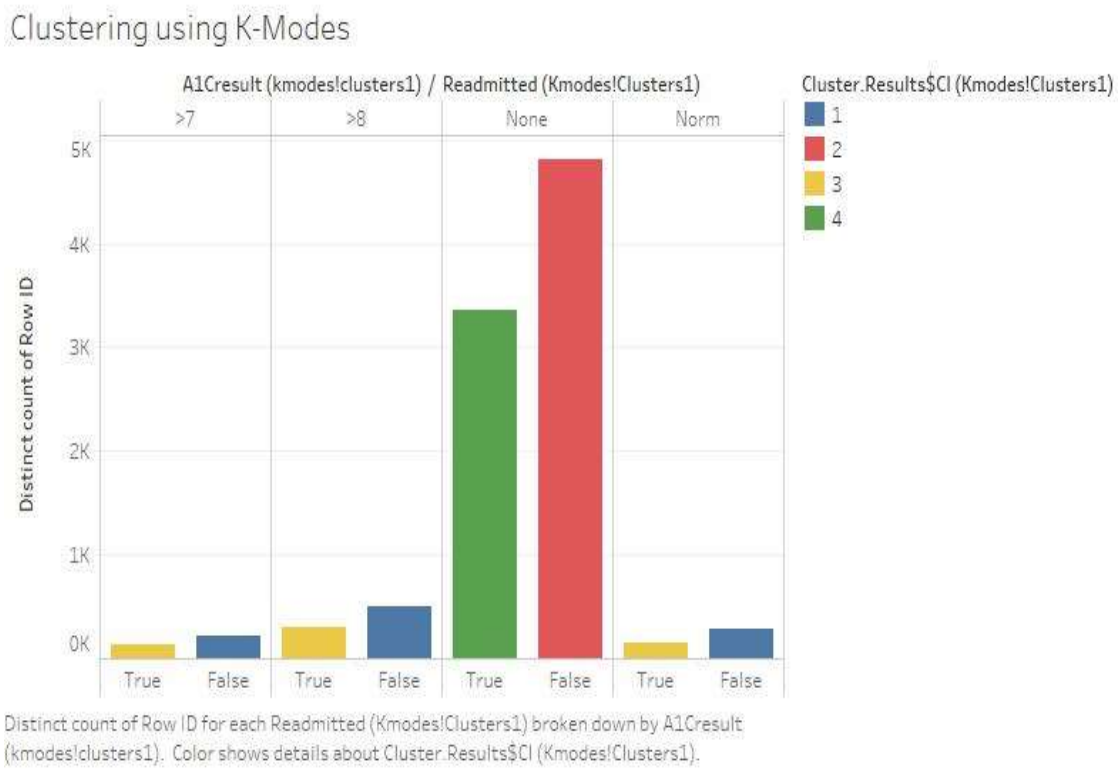

Figure 16: Clustering Results

5. Task 1

Problem Statement: How insulin and change in medication affects HbA1C result.
Why?
This task helps prevent early readmission indirectly by monitoring the administration of medication and insulin levels that affect the HbA1C result. Some patients may choose not to take the HbA1C tests. Hence, we could use insulin and change in medication to predict readmission.
Techniques
Bayesian Network, Logistic Regression

We are using Bayesian network to find the conditional dependencies between A1C result, insulin, change in medication and readmission. Using these dependencies, we are finding the effects of insulin and change in medication of A1C result.
We ran a few queries using the cpquery method for the event A1C result being none given different combination of values for insulin and change in medication as evidence.

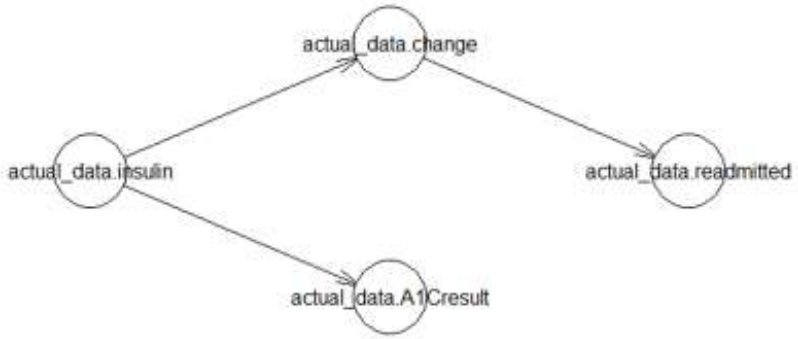

Figure 17: Bayesian Network Dependency Graph

The above figure shows that insulin level affects A1Cresult and change in med, and change in med affects readmission rate. From our initial analysis, it was proved that there is a strong correlation between A1Cresult and readmission rate. Thus, it proves that readmission rate indirectly depends on insulin and change_in_medication. On fitting this graph using bn.fit, and running a few cp queries to check what values of insulin and change in medication affected the probability of A1C test not being taken, we got the following results:

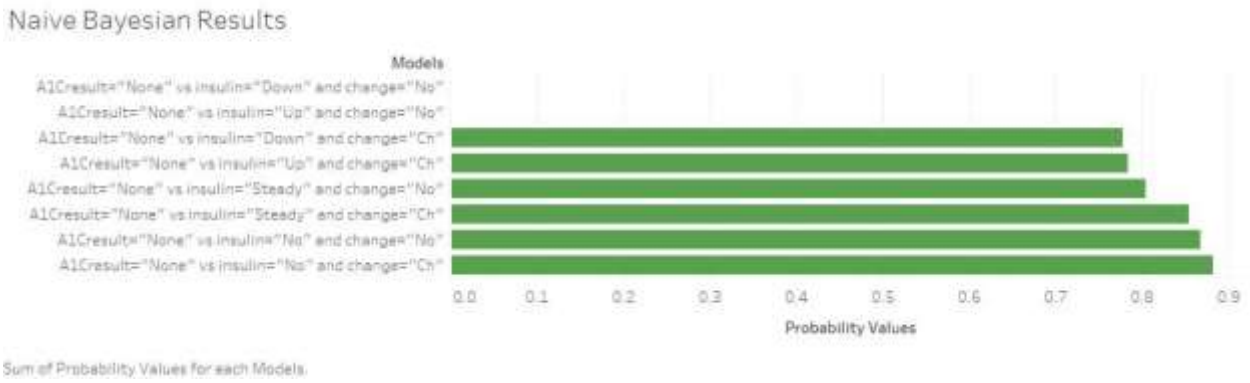

Figure 18: Cp Query Results

From the results, it shows that the probability of A1Cresult being “none” is highest when there is no insulin medication administered, but there was a change in medication. This implies that perhaps another diabetes medication was administered which reinforces the fact that the encounter was in fact diabetic, and there was no necessity for an A1Cresult.

Logistic Regression Analysis

The following regression models were developed

1. Predicting the linear relationship between readmission rate (dependent

variable) and HbA1C result (independent variable)
2. Predicting the linear relationship between readmission rate (dependent variable) and HbA1C result, insulin (independent variables)
3. Predicting the linear relationship between readmission rate (dependent variable) and HbA1C result, insulin, change in medication (independent variables)

ANOVA Results

Resid. Df	Resid. Dev	Df	Deviance
1	7813		
2	7810	3	-0.9486
3	7809	1	-26.8344

Table 2: ANOVA results

From the above table, we could infer that model 3 (HbA1C, insulin, change in medication) influences readmission rate the most.

6. Task 2

Problem Statement: Predicting the likelihood of a non-diabetic patient being readmitted

Why?

This task was proposed in order to find the pattern between non-diabetic patients and readmitted. Since we concentrate mostly on diabetic encounters, we wanted to see what effect a non-diabetic encounter has on readmission rate

Techniques

Pattern Mining, Logistic Regression

In order to determine this, pattern mining techniques were used to see the patterns in the variables of LHS for “diabetesMed” = No as the RHS. The following results were obtained:

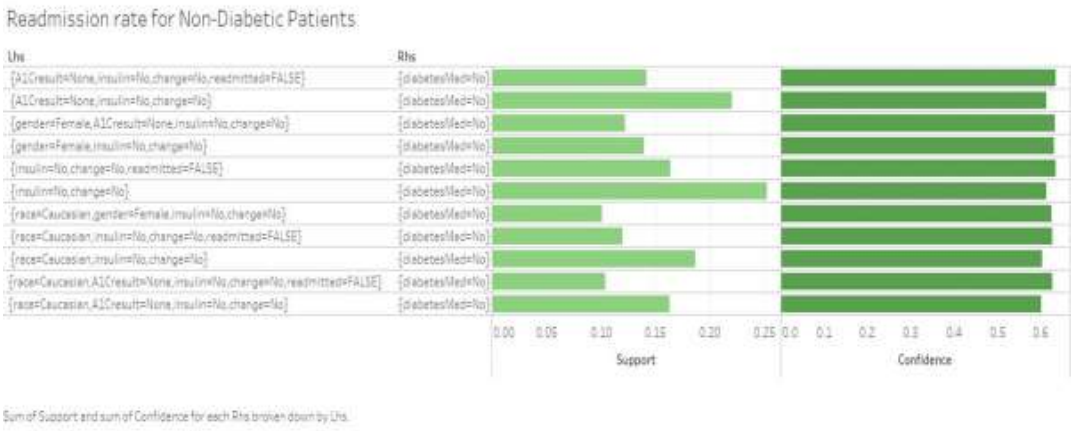

Figure 19: Rules for Diabetic Patients being Readmitted

Logistic Regression

Logistic regression was run on the readmission data to determine the relationship between a non-diabetic patient getting readmitted.
The following regression models were developed:

1. Predicting the linear relationship between readmission rate(dependent variable) and primary diagnosis as diabetes (independent variable)
2. Predicting the linear relationship between readmission rate (dependent variable) and primary diagnosis as circulatory disorder, secondary diagnosis as diabetes (independent variables)

ANOVA Results

Resid. Df	Resid. Dev	Df	Deviance
1	10550		
2	10350	2	-0.7646

Table 3: ANOVA Results

From the above table, we could infer that model 2 (patients with primary diagnosis as circulatory & secondary diagnosis as diabetes) influences readmission rate the most.

7. Conclusion

- By Chi-squared analysis, we found that the HBA1C results, influences the readmission rate.
- By Logistic regression, we anticipate that A1C result, insulin and change in medication highly influences the readmission rate.
- Using various classification models, we can predict the readmission rate of the future patients.
- By Bayesian network, we find that even though A1C was not performed, we can use change in medication and insulin to predict readmission rate to a certain extent.
- By clustering analysis, we envisage that the decision to test for A1C result, impacted the readmission rates rather than the actual values of the A1C result.
- From our analysis, the profile of readmission differed significantly where HbA1C in a setting of those with primary circulatory disorder & diabetes as secondary disorder compared to those with primary diabetes disorder.

REFERENCES:

Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios and John N. Clore. (2014). Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records. Hindawi Publishing Corporation, 2014.

APPENDIX

R Code

```
#Loading the data set
setwd("C:/Users/werms/Desktop/KDD/Project")
patients <- read.csv("10kDiabetes.csv", stringsAsFactors = F)

Data Cleaning:
#Removing inconsistencies

library(editrules)
(E<-editfile("rules.txt"))
ve <- violatedEdits(E, patients)
summary(ve)
plot(ve)

#rules.txt
A1Cresult %in% c('Norm','None','>7','>8')
readmitted %in% c('FALSE','TRUE')
gender %in% c('Male','Female')
insulin %in% c('Down','No','Up','Steady')
time_in_hospital>0
weight %in% c('[0-25)','[25-50)','[50-75)','[75-100)','[100-125)','[125-150)','[150-
175)')
race %in% c('Caucasian','AfricanAmerican','Asian','Hispanic','Other')
change %in% c('No','Ch')
diabetesMed %in% c('Yes','No')
insulin %in% c('Up','Down','No','Steady')


#Removing patients with hospice and expired
patients <- subset(patients, discharge_disposition_id != "Expired")
patients <- subset(patients, discharge_disposition_id != "Hospice / home")

#Changing to numeric variables
head(patternData)
patientData
patientData$A1Cresult <- as.character(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == "None"] <- 1
patientData$A1Cresult <- as.character(patientData$A1Cresult)
```

```
patientData$A1Cresult[patientData$A1Cresult == "Norm"] <- 2
patientData$A1Cresult <- as.character(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == ">7"] <- 3
patientData$A1Cresult <- as.character(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == ">8"] <- 4
patientData$A1Cresult <- as.numeric(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == "None"] <- 1
patientData$A1Cresult <- as.numeric(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == "Norm"] <- 2
patientData$A1Cresult <- as.numeric(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == ">7"] <- 3
patientData$A1Cresult <- as.numeric(patientData$A1Cresult)
patientData$A1Cresult[patientData$A1Cresult == ">8"] <- 4
patientData$A1Cresult
patientData$max_glu_serum <- as.character(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == "None"] <- 1
patientData$max_glu_serum <- as.character(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == "Norm"] <- 2
patientData$max_glu_serum <- as.character(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == ">200"] <- 3
patientData$max_glu_serum <- as.character(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == ">300"] <- 4
patientData$max_glu_serum <- as.numeric(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == "None"] <- 1
patientData$max_glu_serum <- as.numeric(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == "Norm"] <- 2
patientData$max_glu_serum <- as.numeric(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == ">200"] <- 3
patientData$max_glu_serum <- as.numeric(patientData$max_glu_serum)
patientData$max_glu_serum[patientData$max_glu_serum == ">300"] <- 4
patientData$max_glu_serum
patientData$insulin <- as.character(patientData$insulin)
patientData$insulin[patientData$insulin == "No"] <- 1
patientData$insulin <- as.character(patientData$insulin)
patientData$insulin[patientData$insulin == "Down"] <- 2
patientData$insulin <- as.character(patientData$insulin)
patientData$insulin[patientData$insulin == "Steady"] <- 3
patientData$insulin <- as.character(patientData$insulin)
patientData$insulin[patientData$insulin == "Up"] <- 4
```

```
patientData$insulin <- as.numeric(patientData$insulin)
patientData$insulin[patientData$insulin == "No"] <- 1
patientData$insulin <- as.numeric(patientData$insulin)
patientData$insulin[patientData$insulin == "Down"] <- 2
patientData$insulin <- as.numeric(patientData$insulin)
patientData$insulin[patientData$insulin == "Steady"] <- 3
patientData$insulin <- as.numeric(patientData$insulin)
patientData$insulin[patientData$insulin == "Up"] <- 4
patientData$change <- as.character(patientData$change)
patientData$change[patientData$change == "No"] <- 0
patientData$change <- as.character(patientData$change)
patientData$change[patientData$change == "Ch"] <- 1
patientData$change <- as.numeric(patientData$change)
patientData$change[patientData$change == "No"] <- 0
patientData$change <- as.numeric(patientData$change)
patientData$change[patientData$change == "Ch"] <- 1
patientData$readmitted <- as.character(patientData$readmitted)
patientData$readmitted[patientData$readmitted == "TRUE"] <- 1
patientData$readmitted <- as.character(patientData$readmitted)
patientData$readmitted[patientData$readmitted == "FALSE"] <- 0
patientData$readmitted <- as.numeric(patientData$readmitted)
patientData$readmitted[patientData$readmitted == "TRUE"] <- 1
patientData$readmitted <- as.numeric(patientData$readmitted)
patientData$readmitted[patientData$readmitted == "FALSE"] <- 0

#Chi-Square Analysis
tab <- xtabs(~race + readmitted, data = patients) barplot(tab, main="Data
Distribution by Readmitted patients vs Race results",xlab="Race
Results",col=c("darkgreen","blue","red","orange")  legend = rownames(tab),
beside=TRUE) summary(assocstats(tab))

tab <- xtabs(~age + readmitted, data = patients)
barplot(tab, main="Data Distribution by Readmitted patients vs Age
results",xlab="AgeResults",col=c("darkgreen","blue","red","orange","yellow","ma
roon","pink","violet","cyan","magenta"),legend = rownames(tab), beside=TRUE)

summary(assocstats(tab))

tab <- xtabs(~A1Cresult + readmitted, data = patients)
barplot(tab, main="Data Distribution by Readmitted patients vs HBA1C
```

```
results",xlab="HBA1C Results", col=c("darkgreen","blue","red","orange"),legend
= rownames(tab), beside=TRUE)

summary(assocstats(tab))

tab <- xtabs(~change + readmitted , data = patients)
barplot(tab, main="Data Distribution by Readmitted patients vs Change in
Medication results", xlab="Change in Medication Results",
col=c("darkgreen","red"), legend = rownames(tab), beside=TRUE)

summary(assocstats(tab))

#Building the correlation matrix

patientMatrix <- cor(patientData)
patientMatrix
round(patientMatrix,2)

#statistic descriptives
install.packages("pastecs")

library(pastecs)
round(stat.desc(patientMatrix),2)

#Bartlett test
library(psych)
cortest.bartlett(patientMatrix)

#PCA
pc1 <- principal(patientMatrix,nfactors = 3, rotate = "none")
pc1$values
pc1
plot(pc1$values, type= "b")

print.psych(pc1, cut=0.3, sort= TRUE)

#Pattern mining
install.packages("arules")
library(arules)
```

```
patternData <-
subset(patients,select=c("race","gender","age","A1Cresult","insulin","change","dia
betesMed","readmitted","diag_1_desc"))
patternData[,"age"] <- factor(patternData[,"age"])
patternData[,"race"] <- factor(patternData[,"race"])
patternData[,"gender"] <- factor(patternData[,"gender"])
patternData[,"A1Cresult"] <- factor(patternData[,"A1Cresult"])
patternData[,"insulin"] <- factor(patternData[,"insulin"])
patternData[,"change"] <- factor(patternData[,"change"])
patternData[,"diabetesMed"] <- factor(patternData[,"diabetesMed"])
patternData[,"readmitted"] <- factor(patternData[,"readmitted"])
patternData[,"diag_1_desc"] <- factor(patternData[,"diag_1_desc"])
patternPatients <- as(patternData,"transactions")
rules<-
apriori(patternPatients,parameter=list(minlen=2,support=0.1,confidence=0.6),
          appearance=list
(rhs=c("readmitted=TRUE","readmitted=FALSE"),default="lhs"))

inspect(head(rules))
rules.sort <- sort(rules, by="lift")
rules.sort
inspect(head(rules.sort))
subset.matrix<-is.subset(rules.sort,rules.sort)
subset.matrix
subset.matrix[lower.tri(subset.matrix,diag=T)]<-0
redudant<-colSums(subset.matrix)>=1
head(redudant)

rules.pruned<-rules.sort[!redudant]
inspect(rules.pruned)
inspect(head(rules.pruned))

library(arulesViz)
plot(rules.pruned)

plot(rules.pruned,method="graph",control=list(type="items"))
plot(rules.pruned,method="paracoord",control=list(reorder=TRUE))
```

```
#Decision Tree
install.packages("rpart")
library(rpart)
treeData <-
subset(patients,select=c("race","age","gender","change","insulin","A1Cresult","rea
dmitted"))
treeData

fit <- rpart(formula=readmitted ~ ., data=treeData,
    control=rpart.control(minsplit=1, minbucket=1, cp=0.001))
fit
plot(fit)
text(fit,pretty=0)
install.packages("rattle")
library(rattle)

install.packages("rpart.plot")
library(rpart.plot)
install.packages("RColorBrewer")
library(RColorBrewer)
fancyRpartPlot(fit)

set.seed(100)
training <- sample (1:nrow(treeData), 0.8*nrow(treeData)) # training row indices
trainingData <- treeData[training, ] # training data
testData <- treeData[-training, ] # test data
fit <- rpart(formula=readmitted ~ .,
data=trainingData,control=rpart.control(minsplit=1, minbucket=1, cp=0.001))

treeData$readmitted <- factor(treeData$readmitted)
fit <- rpart(readmitted~.,data=treeData)
test_predictions = predict(fit, testData, type = "class")

head(test_predictions)
table(test_predictions)
test_error = sum(test_predictions != testData$readmitted)/nrow(testData)
test_error

confMat <- table(testData$readmitted,test_predictions)
confMat
```

```
accuracy <- sum(diag(confMat))/sum(confMat)
accuracy

#Random Forest
rfData <-
subset(patients,select=c("race","gender","age","A1Cresult","insulin","change","dia
betesMed","readmitted"))
rfData[,"age"] <- factor(rfData[,"age"])
rfData[,"race"] <- factor(rfData[,"race"])
rfData[,"gender"] <- factor(rfData[,"gender"])

rfData[,"A1Cresult"] <- factor(rfData[,"A1Cresult"])
rfData[,"insulin"] <- factor(rfData[,"insulin"])
rfData[,"change"] <- factor(rfData[,"change"])

rfData[,"diabetesMed"] <- factor(rfData[,"diabetesMed"])
rfData[,"readmitted"] <- factor(rfData[,"readmitted"])

training <- sample (1:nrow(rfData), 0.8*nrow(rfData)) # training row indices
trainingData <- rfData[training, ] # training data
testData <- rfData[-training, ] # test data

library(randomForest)
rfData.rf <- randomForest(readmitted ~ ., data = trainingData,mtry = 2, importance
= TRUE)
rfData.rf
prediction<-predict(rfData.rf,testData,type="class")
summary(prediction)
test_error = sum(prediction != testData$readmitted)/nrow(testData)
test_error

rfData.rf$importance
mean(prediction == testData$readmitted)*100
library(caret)
confusionMatrix(data=as.factor(prediction), reference =
as.factor(testData$readmitted))

library(doParallel)
clus<-makeCluster(spec=8,type="PSOCK")
registerDoParallel(clus)
```

```
rfData.rf1<-
train(readmitted~.,data=trainingData,method="rf",metric='Accuracy',tuneGrid=exp
and.grid(.mtry=1:6),ntree=500)
summary(rfData.rf1)
plot(rfData.rf1)
rfData.rf1

#Neural Network
library(neuralnet)
library(nnet)
nnetData <-
subset(patients,select=c("race","gender","age","A1Cresult","insulin","change","dia
betesMed","readmitted"))
uniqueLevels <- unique(nnetData$readmitted)
uniqueLevels

training <- sample (1:nrow(nnetData), 0.8*nrow(nnetData)) # training row indices
trainingData <- nnetData[training, ] # training data
testData <- nnetData[-training, ] # test data
nnetData$gender <- factor(nnetData$gender, labels = seq(1:2), levels =
unique(nnetData$gender))

nnetData$race <- factor(nnetData$race, labels = seq(1:6), levels =
unique(nnetData$race))

nnetData$age <- factor(nnetData$age, labels = seq(1:10), levels =
unique(nnetData$age))

nnetData$A1Cresult <- factor(nnetData$A1Cresult, labels = seq(1:4), levels =
unique(nnetData$A1Cresult))

nnetData$insulin <- factor(nnetData$insulin, labels = seq(1:4), levels =
unique(nnetData$insulin))

nnetData$change <- factor(nnetData$change, labels = seq(1:2), levels =
unique(nnetData$change))

nnetData$diabetesMed <- factor(nnetData$diabetesMed, labels = seq(1:2), levels =
unique(nnetData$diabetesMed))
```

```
nnetData$readmitted <- factor(nnetData$readmitted, labels = seq(1:2), levels =
unique(nnetData$readmitted))

trainset <- cbind(nnetData[,1:8], class.ind(nnetData$readmitted))
testset <- cbind(nnetData[,1:8], class.ind(nnetData$readmitted))
colnames(trainset)[9:10] <- c("ReadmittedTrue","ReadmittedFalse")
colnames(testset)[9:10] <- c("ReadmittedTrue","ReadmittedFalse")

name_data <- names(trainset)
fmla <- as.formula(paste("ReadmittedTrue + ReadmittedFalse ~", paste(name_data
[!name_data %in% c("ReadmittedTrue","ReadmittedFalse")], collapse = " + ")))
trainset <- sapply (trainset, as.numeric)
testset <- sapply(testset, as.numeric)

nnetData.nn <- neuralnet(fmla, data = trainset, hidden=5, threshold = 0.01,
linear.output = F)
plot(nnetData.nn)

pred <- compute (nnetData.nn, testset[,1:8])
pred

summary(pred)
maxidx <- function(arr) { return(which(arr == max(arr)))}
idx <- apply(pred$net.result, c(1), maxidx)
prediction <- c(as.vector(unique(nnetData$readmitted)))[idx]
prediction <- as.factor(prediction)

confusionMatrix(data=factor(prediction), reference = factor(nnetData$readmitted))

#KNN

library(class)
library(VIM)

knnData <-
subset(patients,select=c("race","gender","age","A1Cresult","insulin","change","dia
betesMed","readmitted"))

imputed.knnData <- kNN(knnData)
```

```
new.knnData <- imputed.knnData[,1:8]

new.knnData$gender <- factor(new.knnData$gender, labels = seq(1:2), levels =
unique(new.knnData$gender))

new.knnData$race <- factor(new.knnData$race, labels = seq(1:6), levels =
unique(new.knnData$race))

new.knnData$age <- factor(new.knnData$age, labels = seq(1:10), levels =
unique(new.knnData$age))

new.knnData$A1Cresult <- factor(new.knnData$A1Cresult, labels = seq(1:4),
levels = unique(new.knnData$A1Cresult))

new.knnData$insulin <- factor(new.knnData$insulin, labels = seq(1:4), levels =
unique(new.knnData$insulin))

new.knnData$change <- factor(new.knnData$change, labels = seq(1:2), levels =
unique(new.knnData$change))

new.knnData$diabetesMed <- factor(new.knnData$diabetesMed, labels = seq(1:2),
levels = unique(new.knnData$diabetesMed))

new.knnData$readmitted <- factor(new.knnData$readmitted, labels = seq(1:2),
levels = unique(new.knnData$readmitted))

train <- sample(1:nrow(new.knnData), 0.8*nrow(new.knnData))
train_data <- new.knnData[train,]
test_data <- new.knnData[-train,]
train_class_variable <- train_data[,8]
test_class_variable <- test_data[,8]

for(i in seq(10, 100, 10))
{
 knnTest <- knn(train_data, test_data, train_class_variable, k = i)
 acc <- mean(knnTest == test_class_variable)
 print(paste("Accuracy at k = ", i, " is: ", acc * 100, "%"))
 }
```

```
confusionMatrix(data=factor(knnTest), reference = factor(test_data[,8]), positive =
"1")


#Clustering
library(klaR)
cluster.results <-kmodes(patientData[,c(6,9)], 4, iter.max = 10, weighted = FALSE
)
cluster.results$cluster
cluster.output <- cbind(patientData,cluster.results$cl)
write.csv(cluster.output, file = "kmodes_clusters1.csv", row.names = TRUE)


#Logistic Regression
patients <- subset(patients, discharge_disposition_id != "Expired")
patients <- subset(patients, discharge_disposition_id != "Hospice / home")

patientData <-
subset(patients,select=c("race","age","time_in_hospital","A1Cresult","change","in
sulin","diabetesMed","readmitted","diag_1","diag_2","diag_3"))

training <- sample (1:nrow(patientData), 0.8*nrow(patientData)) # training row
indices
trainingData <- patientData[training, ] # training data
testData <- patientData[-training, ] #test data

install.packages("nnet")
library(nnet)

install.packages("caret")
library(caret)


logisticData <-multinom(readmitted~., data=trainingData)

#Naive Bayes

install.packages("e1071")
library(e1071)
```

```
set.seed(100)
sample<-sample(1:10000,size = 0.8*nrow(diabetesData)) # 80% train and 20% test

train_data<-diabetesData[sample,]
test_data<-diabetesData[-sample,]

nbClassifier<-naiveBayes(train_data[,1:48],train_data[,49]) # 1:48- all columns
until readmitted
table(predict(nbClassifier,test_data[,1:48]),test_data[,49])
test_prediction<-predict(nbClassifier, test_data, type = "class")
test_prediction

test_error = sum(test_prediction != test_data$readmitted)/nrow(test_data) #nb-
main task
test_error
print(paste('Accuracy',1-test_error))


#Bayesian Network-Task 1

install.packages(“bnlearn”)
actual_data<-diabetesData
actual_data$readmitted=ifelse(actual_data$readmitted=="FALSE",0,1)

X4<-data.frame(actual_data$insulin,actual_data$A1Cresult,
actual_data$readmitted, actual_data$change)
res4<-hc(X4)
plot(res4)

fittedbn <- bn.fit(res4, data = X4)

fittedbn
fittedbn$actual_data.insulin
fittedbn$actual_data.A1Cresult
fittedbn$actual_data.readmitted
fittedbn$actual_data.change

//queries
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="Up" & actual_data.change=="Ch" ))
```

```
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="Down" & actual_data.change=="Ch" ))
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="Steady" & actual_data.change=="Ch" ))
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="Up" & actual_data.change=="No" ))
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="Down" & actual_data.change=="No" ))
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="Steady" & actual_data.change=="No" ))
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="No" & actual_data.change=="No" ))
cpquery(fittedbn, event = (actual_data.A1Cresult=="None"), evidence = (
actual_data.insulin=="No" & actual_data.change=="Ch" ))


#Logistic Regression
patientData <-
subset(patients,select=c("A1Cresult",”change”,"insulin","readmitted"))
patientData <- subset(patients,select=c("A1Cresult","insulin","readmitted"))
patientData <- subset(patients,select=c("A1Cresult","readmitted"))
training <- sample (1:nrow(patientData ), 0.8*nrow(patientData )) # training row
indices
trainingData <- patientData [training, ] # training data
testData <- patientData [-training, ] #test data
logisticData <-multinom(readmitted~., data=trainingData)
model2 <- glm (readmitted~., data = trainingData, family = binomial)
model1 <- glm (readmitted~., data = trainingData, family = binomial)
model <- glm (readmitted~., data = trainingData, family = binomial)
anova(model2,model1,model)


#Task 2

#Pattern Mining

install.packages("arules")
library(arules)
```

```
patientData<-
subset(diabetesData,select=c("race","gender","age","A1Cresult","insulin","change
","readmitted","diabetesMed"))
patientData[,"age"] <- factor(patientData[,"age"])
patientData[,"race"] <- factor(patientData[,"race"])
patientData[,"gender"] <- factor(patientData[,"gender"])

patientData[,"A1Cresult"] <- factor(patientData[,"A1Cresult"])
patientData[,"insulin"] <- factor(patientData[,"insulin"])
patientData[,"change"] <- factor(patientData[,"change"])

patientData[,"diabetesMed"] <- factor(patientData[,"diabetesMed"])
patientData[,"readmitted"] <- factor(patientData[,"readmitted"])

patternPatients <- as(patientData,"transactions")

rules<-
apriori(patternPatients,parameter=list(minlen=2,support=0.1,confidence=0.6),
        appearance=list (rhs="diabetesMed=No",default="lhs"))

inspect(rules)

rules.sort <- sort(rules, by="lift")
inspect(rules.sort)


install.packages("arulesViz")
library(arulesViz)
plot(rules.pruned)

plot(rules.pruned,method="graph",control=list(type="items"))

plot(rules.pruned,method="paracoord",control=list(reorder=TRUE))

#Logistic Regression
patientData <- subset(patientData,select=c("readmitted","diag_1","diag_2"))
diagnosis_3<-patientData
diagnosis_3$diag_1 <- as.character(diagnosis_3$diag_1)
diagnosis_3$diag_2 <- as.character(diagnosis_3$diag_2)
diagnosis_3$diag_2[diagnosis_3$diag_2 =="Diabetes"] <- 1
```

```
diagnosis_3$diag_2[diagnosis_3$diag_2 !="Diabetes"] <- 0
diagnosis_3$diag_1[diagnosis_3$diag_1 =="Circulatory"] <- 1
diagnosis_3$diag_1[diagnosis_3$diag_1 !="Circulatory"] <- 0
training <- sample (1:nrow(diagnosis_3), 0.8*nrow(diagnosis_3))
trainingData <- diagnosis_3[training, ] # training data
testData <- diagnosis_3[-training, ] #test data
logisticData <-multinom(readmitted~., data=trainingData)
model1 <- glm (readmitted~., data = trainingData, family = binomial)
model <- glm (readmitted~., data = trainingData, family = binomial)
anova(model1,model)
```