

Predict if prescribed medication requires pre-authorization

Project REPORT

KUNDA VENKATESH | CPEE | 20-09-2017

# Overview

* Prior authorization is a process used by some health insurance companies to determine if they will cover a prescribed procedure, service, or medication. The process is intended to act as a safety and cost-saving measure, although it has received criticism from physicians for being costly and time-consuming
* Prior authorization is a check run by some insurance companies or third party payers before they will agree to cover certain prescribed medications or medical procedures. There are a number of reasons that insurance providers require prior authorization, including age, medical necessity, the availability of a generic alternative, or checking for drug interactions. A failed authorization can result in a requested service being denied, or an insurance company requiring the patient to go through a separate process known as "step therapy" or "fail first". Step therapy dictates that a patient must first see unsuccessful results from a medication or service preferred by the insurance provider, typically considered either more cost effective or safer, before the insurance company will cover a different service

## Purpose

A prior authorization is an additional requirement that some insurance companies require before they decide if they want to pay for your medicine. Our goal is to develop an engine, for doctors that tell whether prescribed medication will require a prior authorization or not. This helps the doctors prescribe those medicines that do not require Prior Authorization in turn and reduces the service time for pharmacy, service process time for insurance, increases the probability of medication purchases and thereby improved health of patients. This engine should be based on advanced machine learning technologies that looks at the past transactional data and predict with high degree of confidence whether a drug will require prior authorization for a particular patient or not.

The purpose of this study was to determine whether and under what circumstances prior authorization(PA) required To replace labor-intensive administrative procedure results in approval of the PA Which is Time Consuming through Automation using R ,Health plans and health policy analysts are interested in whether or not PA programs affect the original prescription rate and the actual utilization rate of affected drugs, given the expensive and administratively intensive nature of these procedures Usually companies Need prior authorization to process the particular prescription Prescription needs to be screened and authorized to get it processed Doctors are involved in screening the prescription

Drugs are of 2 types pertains to regulation

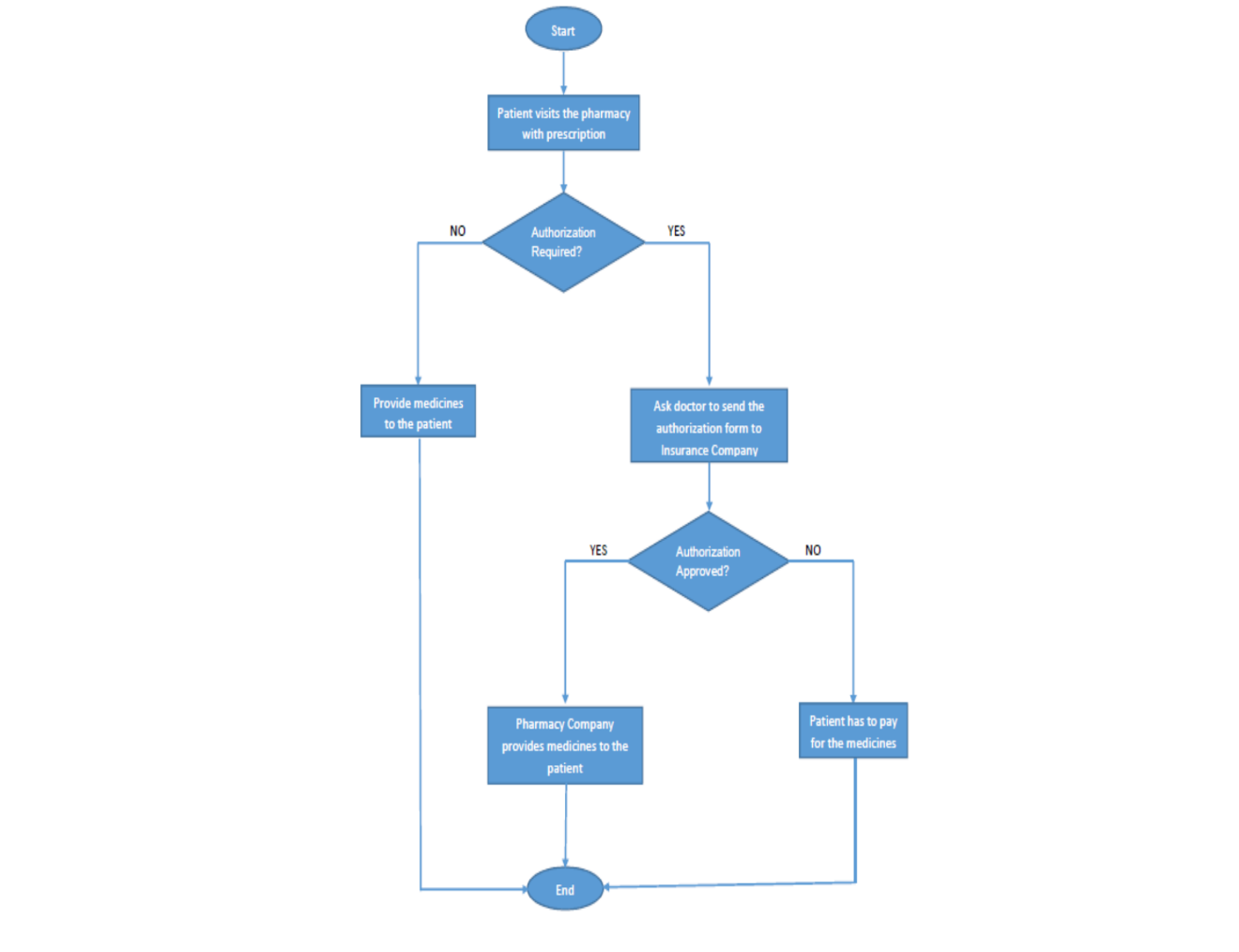
* Formulary –Authorized by Regulatory
* Non-Formulary – Reverse of Formulary

Doctors certify prescription whether its Formulary / Non-Formulary, Time Consuming due to Manual Intervention,Critical to Insurance Companies Based on Study by **KENNETH T. LAPENSEE, MPH, PhD SVP of** Cambridge Parma Consultancy ,observed that more than 95% of PA reviews resulted in payment for the originally prescribed Medicines,Creating An engine to Automate the process with 939Based on Data, statistical analysis was performed 3 Records ,Automation helps in reducing the cycle time of PA Insurance companies can increase their productivity

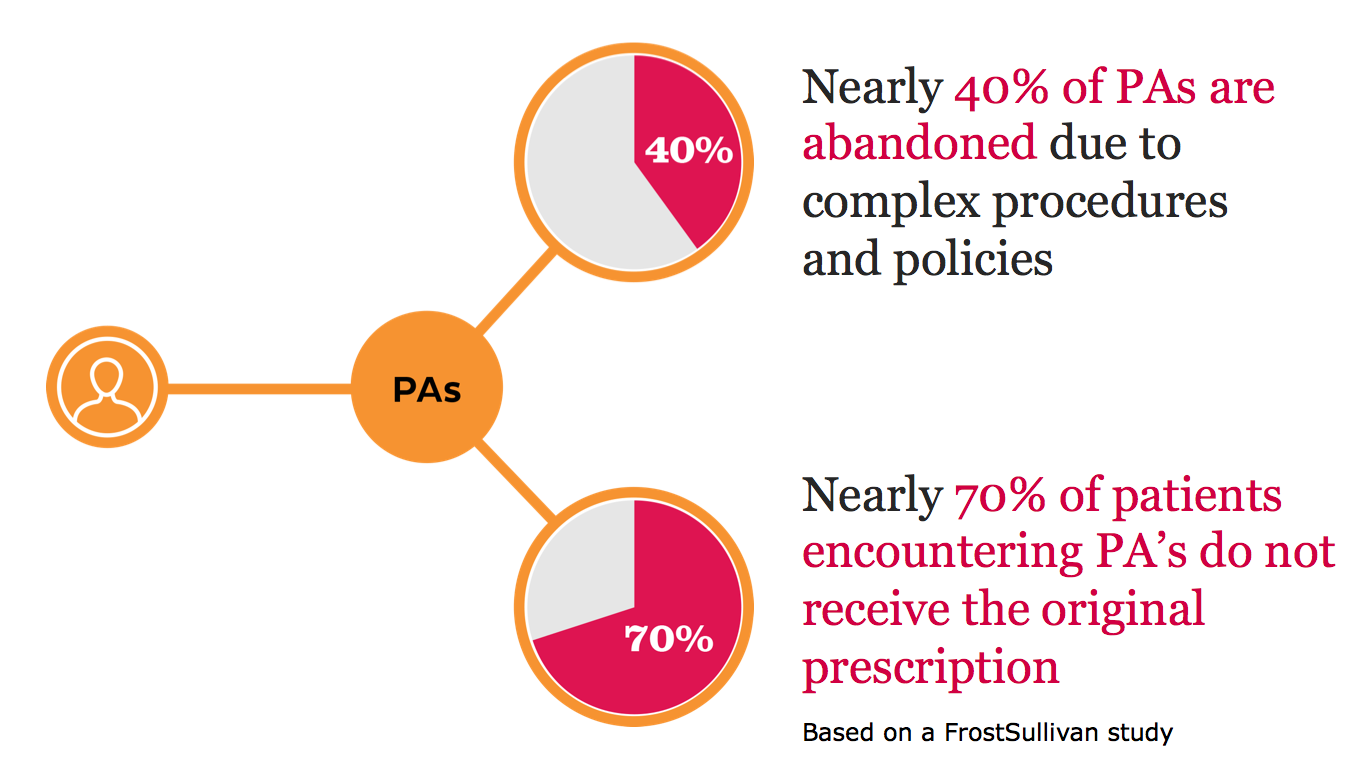
# State of Art

**Existing System:**

After a physician orders a medical service for a patient, the physician's staff will contact the patient's insurer to determine if they require a prior authentication check to be run. If at all it requires a prior authentication check they will initiate the authentication process which may take up to 30 days to approve a request.



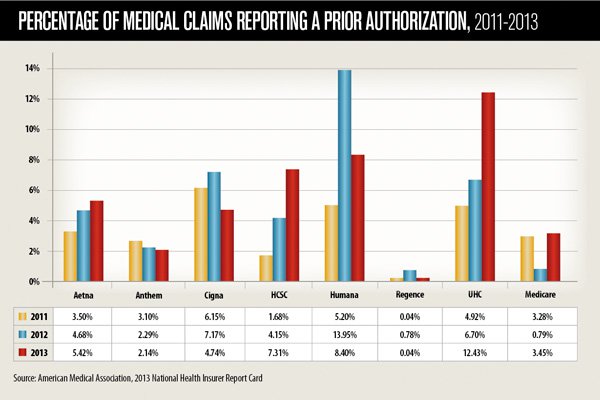
As of May 2013, the National Council for Prescription Drug Programs (NCPDP) had adopted a standardized process for the exchange of electronic prior authentication which reduced the time taken to 90% when compared to manual prior authentication process.



**The costs of prior authorization:**

Prior authorization has been an issue among healthcare providers for at least a quarter of a century, surprisingly little is known about its cost, either to individual practices or to the healthcare system as a whole. In 2006, PCPs spent a mean of 1.1 hours per week on authorizations, primary care nursing staffs spent 13.1 hours, and primary care clerical staff spent 5.6 hours, according to [a 2009 study](http://content.healthaffairs.org/content/28/4/w533.abstract) published in Health Affairs. The study estimated that the overall cost to the healthcare system of all practice interactions with health plans, including authorizations, was between $23 billion and $31 billion annually.

More recently, [a study](http://www.jabfm.org/content/26/1/93.abstract)of 12 primary care practices published earlier this year in the Journal of the American Board of Family Medicine put the mean annual projected cost per full-time equivalent physician for prior authorization activities between $2,161 and $3,430. The study’s authors concluded that “preauthorization is a measurable burden on physician and staff time.”



# Method

* R programming helps to automate the process
* Machine Learning techniques to be built in R used to Solve the Problem
* Initially reduce the Levels in the Data
* Forming the Groups based on similarity between records
* Clustering Technique used to form similar groups

* Fitting the whole data into Mean number of clusters required in order to reduce dimensionality
* Challenge is to find the clusters required to fit the data
* Plotting the centroids (closest point of all the data points in homogenous group) of the cluster

* Tuning the required no of clusters to Fit the data
* Data provided has Target with Binary type,In this case Classification Algorithm is best for prediction
* Using Decision Trees Algorithm to find the possible outcome with the combination of attributes

* Generating the rules to find the combinations which are predicting PA either TRUE/FALSE Pertaining to Target Attribute
* Generating rules until no division is possible
* Approach is bit greedy and also having a chance of over fit (risk)
* To Avoid this, Random Forest algorithm is useful
* Algorithm will take a tree from the forest and find the prediction

* Outcome of the Random Forest Algorithm decide the Authenticity of the Decision tree Algorithm
* Naïve Bayes Algorithm is used to find Whether outcome of the new Record PA is TRUE/FALSE
* Toughest part is to bringdown the number of levels in each and every attribute, I used clustering technique to reduce the number of levels in the data, We cannot build the model without getting number of levels down

# Data

In the data which was given has 15 attribute including Target Variable

**Attributes:**

1.UserID: Patient Identification Number

2.Drug: Drug Identification Number

3.DrugSubClass: Drug Sub Class ID

4.DrugClass: Drug Class ID

5.Drug\_Chemical\_Name: Drug Chemical Name

6.GPI: Generic Product Identifier is a 14-character hierarchical classification system that identifies drugs from their primary therapeutic use down to the unique interchangeable product regardless of manufacturer or package size.

7.NDC: National Drug Code is a unique 10-digit, 3 segment number

8.Drug Group: Group Identification ID

9.DoctorID: Doctor ID

10.RxGroupId: Represents the Group Identification for a medical prescription.

11.BIN: Bank Identification Number is the number that tells the pharmacy database which PBM is to receive the claim for a particular prescription. It is a 6-digit number which tells the pharmacy’s computer where the claim should be sent.

12.PCN: Processor Control Number is a secondary number on a health insurance card that is used to route pharmacy claim transactions for health insurers.

13.State: State ID

14.TransDate: Date for which the transactions happened.

15.Target: Predictor variable – Yes (TRUE) or not (FALSE).

After reading the data to R; analyzed data as follows:

Target is the target(output) variables and predictor variables are UserID,Drug, DrugSubClass, DrugClass, GPI, NDC, DoctorID, Drug Group, RxGroupId, BIN, PCN, TransDate and State.

All the predictor variables are found to be categorical attributes.

In the pre processing of the Data I removed UserId before building the model. And in the TranDate Column dates are not in the Unique format. I used R programming to get the date in the single format.

This is the R code which I used to get the date in the single format

####Making the unique date format in the data#####

a <- as.Date(Data$TransDate,format="%m/%d/%Y") # Produces NA when format is not "%m/%d/%Y"

b <- as.Date(Data$TransDate,format="%m-%d-%Y") # Produces NA when format is not "%d-%m-%Y"

a[is.na(a)] <- b[!is.na(b)] # Combine both while keeping their ranks

Data$TransDate <- a # Put it back in your dataframe

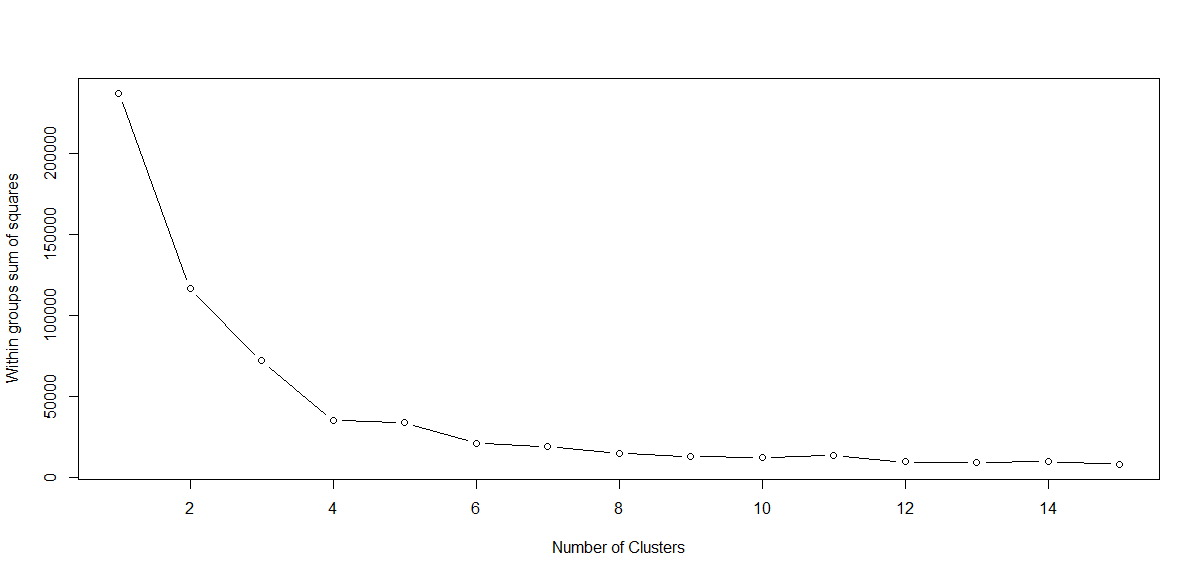
Data$TransDate

Data

# Results

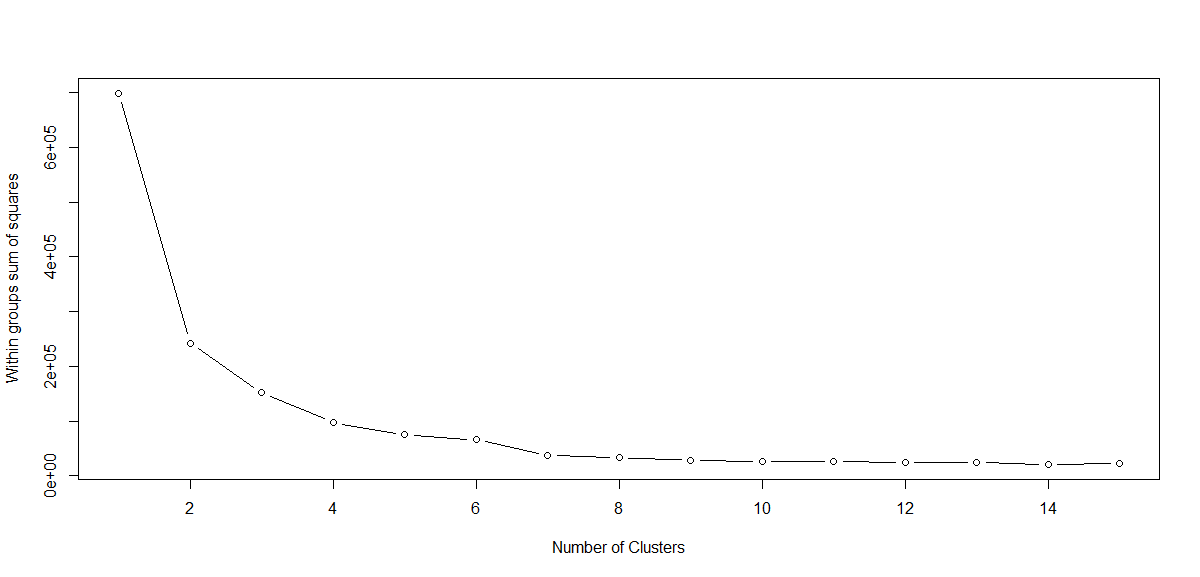
While reducing the number of levels in the each attribute, I used K-Means clustering technique and plotted the elbow cure to find the optimum number of clusters to be formed.

1. For Drug



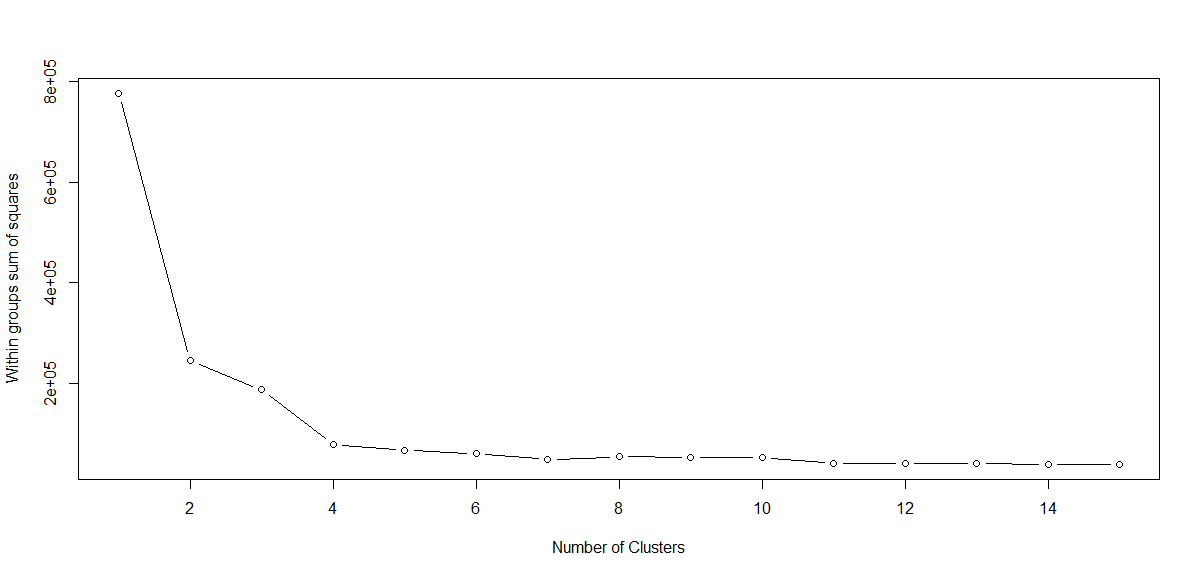
No.of clusters = 12

1. For DrugSubClass



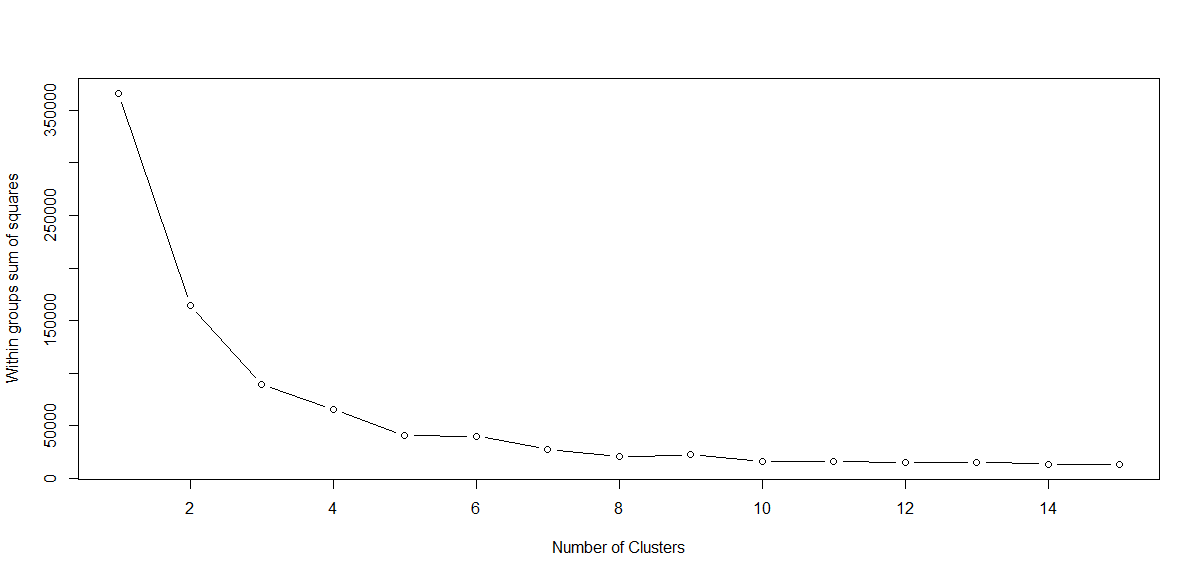
No.of clusters = 10

1. For DrugClass



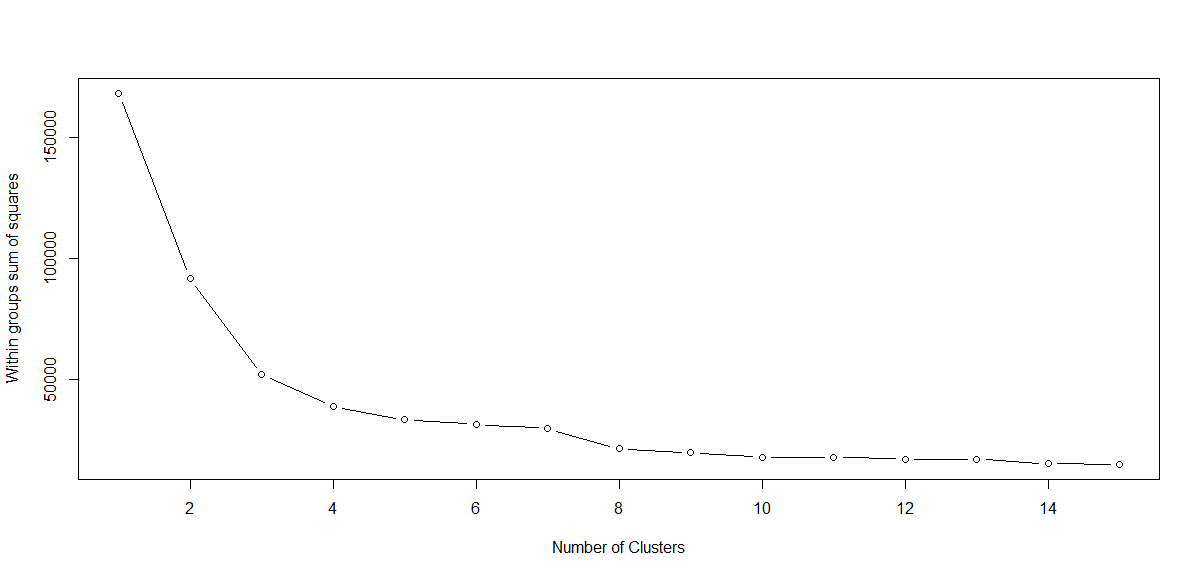
No.of clusters = 11

1. For Drug\_Chemical\_Name



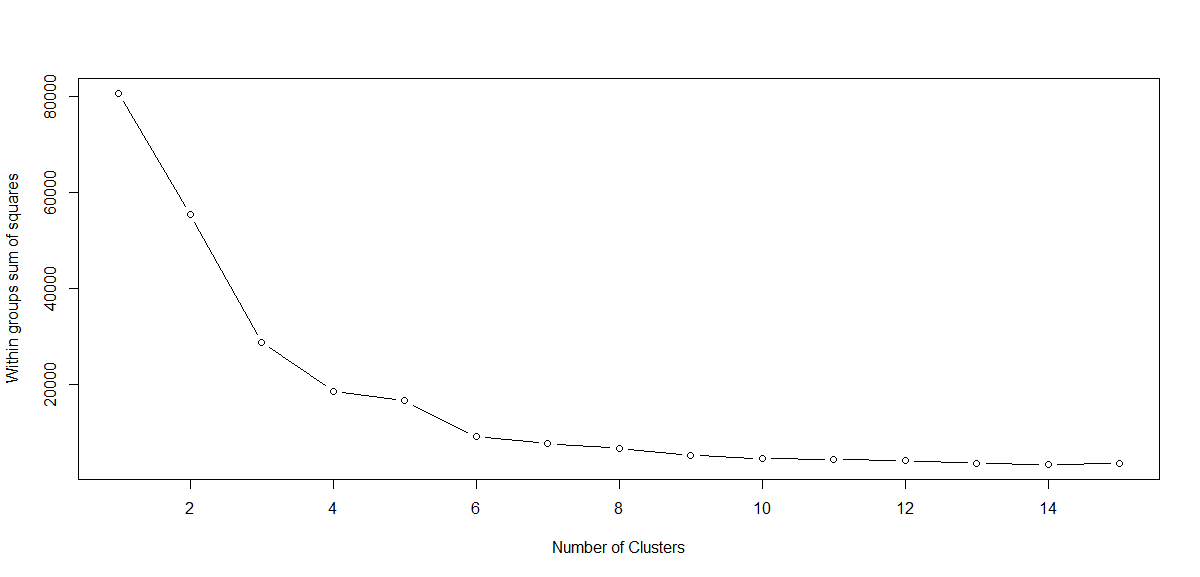
No.of clusters = 10

1. For GPI



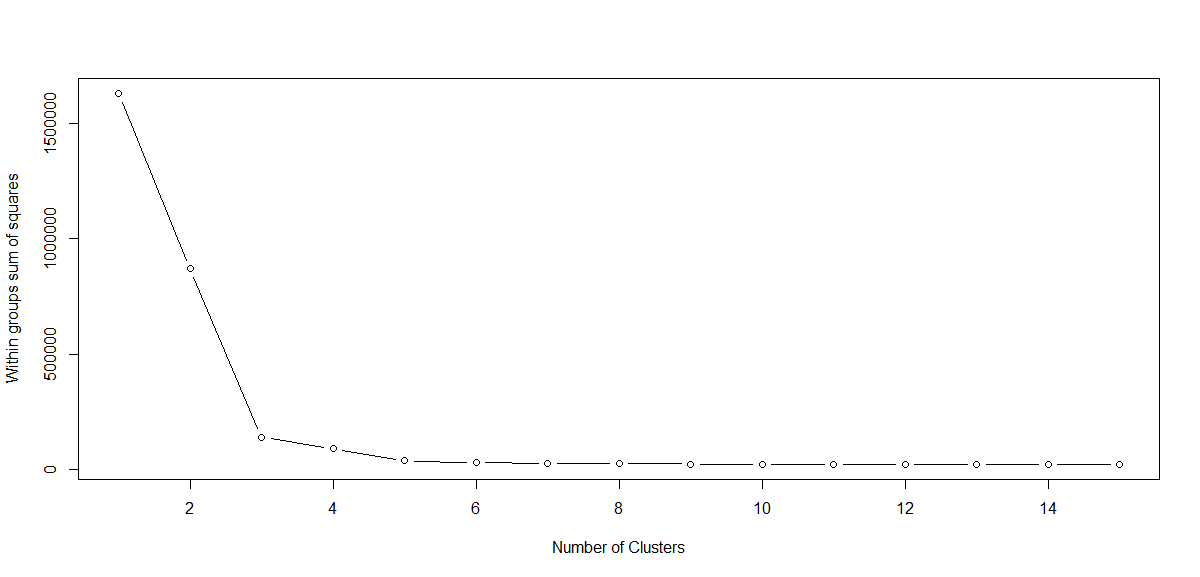
No.of clusters = 14

1. For NDC



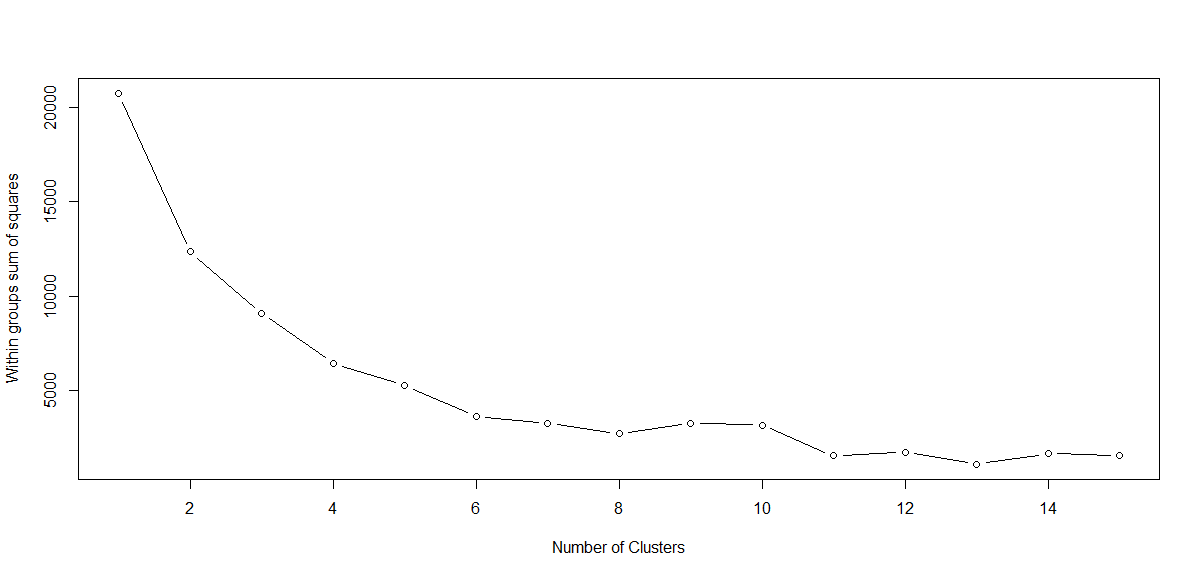
No.of clusters = 13

1. For DrugGroup



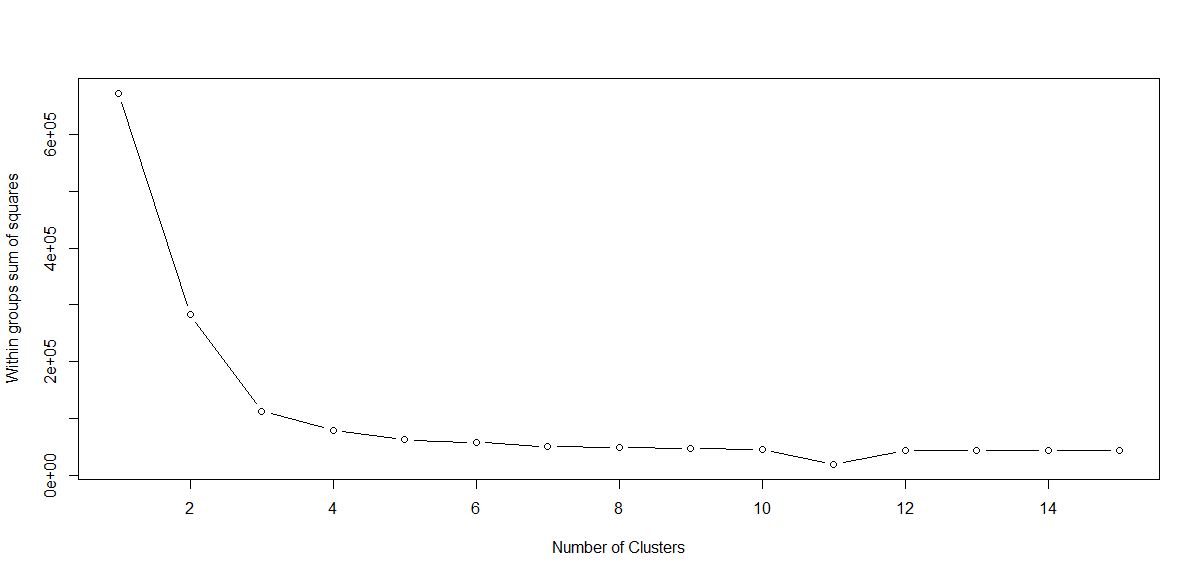
No.of clusters = 7

1. For DoctorId



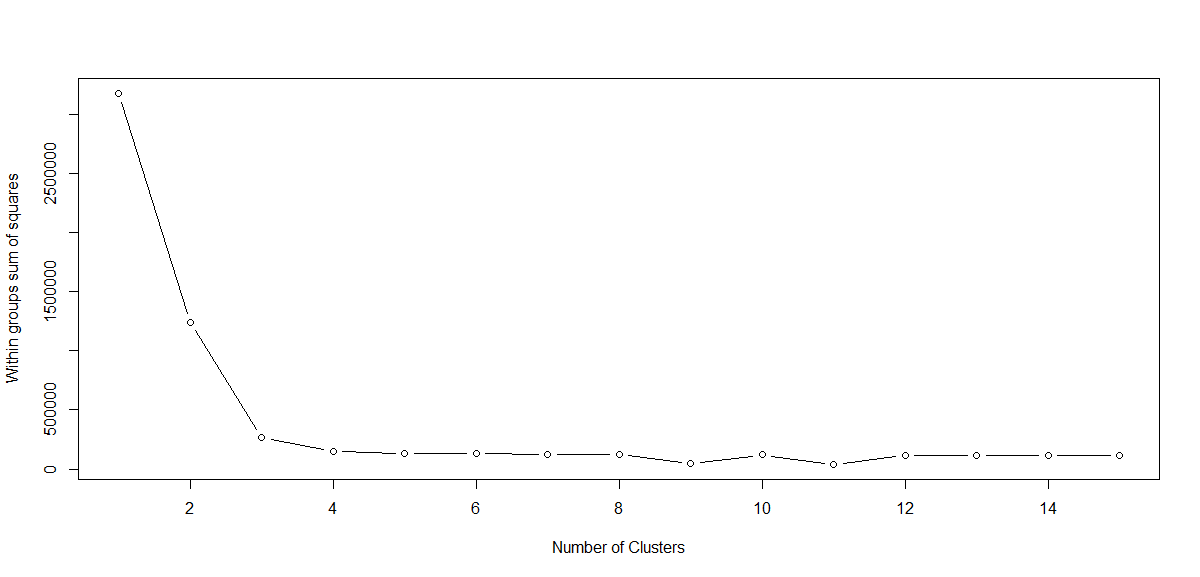
No.of clusters = 15

1. For RX Group Id



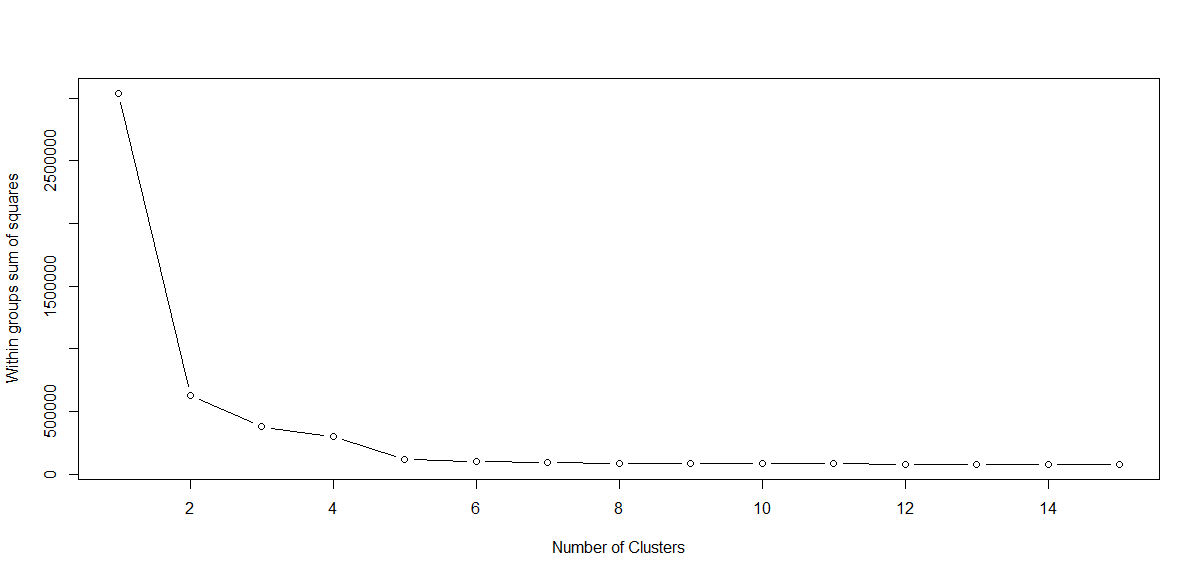
No.of clusters = 10

1. For BIN



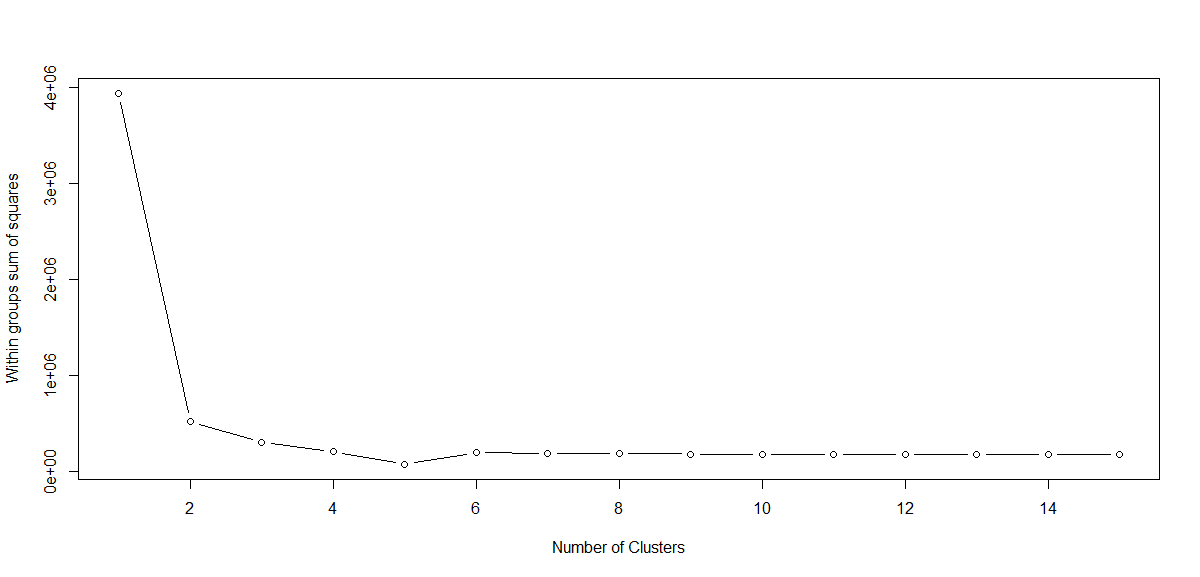
No.of clusters = 8

1. For PCN



No.of clusters = 6

1. For State



No.of clusters = 6

R Code for reducing the number of levels that is to form Clusters.

Forming Clusters to reduce the number of levels in the data##

names(Data)

C1<- 0

for (i in 1:15) {

set.seed(123)

C1[i] <- sum(kmeans(Drug\_Freq,centers=i)$withinss)

}

plot(1:15,C1,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C2<- 0

for (i in 1:15) {

set.seed(123)

C2[i] <- sum(kmeans(DrugSubClass\_Freq,centers=i)$withinss)

}

plot(C2,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C3<- 0

for (i in 1:15) {

set.seed(123)

C3[i] <- sum(kmeans(DrugClass\_Freq,centers=i)$withinss)

}

plot(C3,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C4<- 0

for (i in 1:15) {

set.seed(123)

C4[i] <- sum(kmeans(Drug\_Chemical\_Name\_Freq,centers=i)$withinss)

}

plot(C4,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C5<- 0

for (i in 1:15) {

set.seed(123)

C5[i] <- sum(kmeans(GPI\_Freq,centers=i)$withinss)

}

plot(C5,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C6<- 0

for (i in 1:15) {

set.seed(123)

C6[i] <- sum(kmeans(NDC\_Freq,centers=i)$withinss)

}

plot(C6,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C7<- 0

for (i in 1:15) {

set.seed(123)

C7[i] <- sum(kmeans(DrugGroup\_Freq,centers=i)$withinss)

}

plot(C7,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C8<- 0

for (i in 1:15) {

set.seed(123)

C8[i] <- sum(kmeans(DoctorID\_Freq,centers=i)$withinss)

}

plot(C8,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C9<- 0

for (i in 1:15) {

set.seed(123)

C9[i] <- sum(kmeans(RxGroupId\_Freq,centers=i)$withinss)

}

plot(C9,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C10<- 0

for (i in 1:15) {

set.seed(123)

C10[i] <- sum(kmeans(Bin\_Freq,centers=i)$withinss)

}

plot(C10,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C11<- 0

for (i in 1:15) {

set.seed(123)

C11[i] <- sum(kmeans(PCN\_Freq,centers=i)$withinss)

}

plot(C11,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C12<- 0

for (i in 1:15) {

set.seed(123)

C12[i] <- sum(kmeans(State\_Freq,centers=i)$withinss)

}

plot(C12,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

After Reducing the number of levels

|  |  |  |
| --- | --- | --- |
| **Feature** | **Before** | **After** |
| Drug | 742 | 12 |
| DrugSubClass | 310 | 10 |
| DrugClass | 249 | 11 |
| Drug\_Chemical\_Name | 489 | 10 |
| GPI | 974 | 14 |
| NDC | 1609 | 13 |
| DrugGroup | 77 | 7 |
| Doctor ID | 2468 | 15 |
| RX Group ID | 849 | 10 |
| Bin | 97 | 8 |
| PCN | 196 | 6 |
| State | 47 | 6 |

# Model Building

1. Provided Data is divided in to Training and Testing
2. Model is built on training and evaluated on Testing
3. Data provided has Target with Binary type
4. In this case Classification Algorithm is best for prediction
5. Using Decision Trees Algorithm to find the possible outcome with the combination of attributes
6. Generating the rules to find the combinations which are predicting PA either TRUE/FALSE Pertaining to Target Attribute
7. Generating rules until no division is possible Approach is bit greedy and also having a chance of over fit (risk). To Avoid this Random Forest algorithm is useful
8. Algorithm will take a tree from the forest and find the prediction
9. Outcome of the Random Forest Algorithm decide the Authenticity of the Decision tree Algorithm
10. Naïve Bayes Algorithm is used to find Whether outcome of the new Record PA is TRUE/FALSE

# Analysis

After building the models, these are the confusion matrices built

|  |  |  |
| --- | --- | --- |
| **Decision Tree** | | |
| T/F | FALSE | TRUE |
| FALSE | 438 | 562 |
| TRUE | 70 | 1748 |

|  |  |  |
| --- | --- | --- |
| **Random Forest** | | |
| T/F | FALSE | TRUE |
| FALSE | 408 | 592 |
| TRUE | 376 | 1442 |

|  |  |  |
| --- | --- | --- |
| **Naïve Bayes** | | |
| T/F | FALSE | TRUE |
| FALSE | 509 | 491 |
| TRUE | 284 | 1534 |

# Metrics

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Model** | | |
| **C50** | **Random Forest** | **Naïve Bayes** |
| **Accuracy** | 77.57275 | 65.494 | 72.49823 |
| **Recall** | 96.14961 | 79.31793 | 84.37844 |
| **Precision** | 75.671 | 70.89479 | 75.75309 |

# R code used to build this Project

#### Prior authentication prediction for prescribed medication ###

### Coding ###

rm(list=ls(all=TRUE))

setwd("G:/INSOFE/Project Prior Authorization/")

Data<-read.csv("PriorAuth\_Data.csv",header=T)

str(Data)

####Making the unique date format in the data#####

a <- as.Date(Data$TransDate,format="%m/%d/%Y") # Produces NA when format is not "%m/%d/%Y"

b <- as.Date(Data$TransDate,format="%m-%d-%Y") # Produces NA when format is not "%d-%m-%Y"

a[is.na(a)] <- b[!is.na(b)] # Combine both while keeping their ranks

Data$TransDate <- a # Put it back in your dataframe

Data$TransDate

Data

#To check whether Missing are cohesed while transforming Date#

sum(is.na(Data$TransDate))

#Removing the Attribute User ID and TransDate#

Data<-subset(Data,select=-c(UserID))

Data<-subset(Data,select=-c(TransDate))

names(Data)

########Finding the frequencies########

Drug<-as.data.frame(table(Data$Drug,Data$Target))

DrugSubClass<-as.data.frame(table(Data$DrugSubClass,Data$Target))

DrugClass<-as.data.frame(table(Data$DrugClass,Data$Target))

Drug\_Chemical\_Name<-as.data.frame(table(Data$Drug\_Chemical\_Name,Data$Target))

GPI<-as.data.frame(table(Data$GPI,Data$Target))

NDC<-as.data.frame(table(Data$NDC,Data$Target))

DrugGroup<-as.data.frame(table(Data$DrugGroup,Data$Target))

DoctorID<-as.data.frame(table(Data$DoctorID,Data$Target))

RxGroupId<-as.data.frame(table(Data$RxGroupId,Data$Target))

Bin<-as.data.frame(table(Data$Bin,Data$Target))

PCN<-as.data.frame(table(Data$PCN,Data$Target))

State<-as.data.frame(table(Data$State,Data$Target))

#Reshaping Data#

library(reshape)

Drug\_Freq<-cast(Drug,Var1~Var2)

DrugSubClass\_Freq<-cast(DrugSubClass,Var1~Var2)

DrugClass\_Freq<-cast(DrugClass,Var1~Var2)

Drug\_Chemical\_Name\_Freq<-cast(Drug\_Chemical\_Name,Var1~Var2)

GPI\_Freq<-cast(GPI,Var1~Var2)

NDC\_Freq<-cast(NDC,Var1~Var2)

DrugGroup\_Freq<-cast(DrugGroup,Var1~Var2)

DoctorID\_Freq<-cast(DoctorID,Var1~Var2)

RxGroupId\_Freq<-cast(RxGroupId,Var1~Var2)

Bin\_Freq<-cast(Bin,Var1~Var2)

PCN\_Freq<-cast(PCN,Var1~Var2)

State\_Freq<-cast(State,Var1~Var2)

##Forming Clusters to reduce the number of levels in the data##

names(Data)

C1<- 0

for (i in 1:15) {

set.seed(123)

C1[i] <- sum(kmeans(Drug\_Freq,centers=i)$withinss)

}

plot(1:15,C1,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C2<- 0

for (i in 1:15) {

set.seed(123)

C2[i] <- sum(kmeans(DrugSubClass\_Freq,centers=i)$withinss)

}

plot(C2,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C3<- 0

for (i in 1:15) {

set.seed(123)

C3[i] <- sum(kmeans(DrugClass\_Freq,centers=i)$withinss)

}

plot(C3,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C4<- 0

for (i in 1:15) {

set.seed(123)

C4[i] <- sum(kmeans(Drug\_Chemical\_Name\_Freq,centers=i)$withinss)

}

plot(C4,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C5<- 0

for (i in 1:15) {

set.seed(123)

C5[i] <- sum(kmeans(GPI\_Freq,centers=i)$withinss)

}

plot(C5,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C6<- 0

for (i in 1:15) {

set.seed(123)

C6[i] <- sum(kmeans(NDC\_Freq,centers=i)$withinss)

}

plot(C6,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C7<- 0

for (i in 1:15) {

set.seed(123)

C7[i] <- sum(kmeans(DrugGroup\_Freq,centers=i)$withinss)

}

plot(C7,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C8<- 0

for (i in 1:15) {

set.seed(123)

C8[i] <- sum(kmeans(DoctorID\_Freq,centers=i)$withinss)

}

plot(C8,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C9<- 0

for (i in 1:15) {

set.seed(123)

C9[i] <- sum(kmeans(RxGroupId\_Freq,centers=i)$withinss)

}

plot(C9,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C10<- 0

for (i in 1:15) {

set.seed(123)

C10[i] <- sum(kmeans(Bin\_Freq,centers=i)$withinss)

}

plot(C10,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C11<- 0

for (i in 1:15) {

set.seed(123)

C11[i] <- sum(kmeans(PCN\_Freq,centers=i)$withinss)

}

plot(C11,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

C12<- 0

for (i in 1:15) {

set.seed(123)

C12[i] <- sum(kmeans(State\_Freq,centers=i)$withinss)

}

plot(C12,

type="b",

xlab="Number of Clusters",

ylab="Within groups sum of squares")

##Found number of clusters to be formed on each attribute using elbow curve,

#Merging Frequencies and Clusters##

set.seed(123)

Merge1 <- kmeans(Drug\_Freq,12)

set.seed(123)

Merge2 <- kmeans(DrugSubClass\_Freq,10)

set.seed(123)

Merge3 <- kmeans(DrugClass\_Freq,11)

set.seed(123)

Merge4 <- kmeans(Drug\_Chemical\_Name\_Freq,10)

set.seed(123)

Merge5 <- kmeans(GPI\_Freq,14)

set.seed(123)

Merge6 <- kmeans(NDC\_Freq,13)

set.seed(123)

Merge7 <- kmeans(DrugGroup\_Freq,07)

set.seed(123)

Merge8 <- kmeans(DoctorID\_Freq,15)

set.seed(123)

Merge9 <- kmeans(RxGroupId\_Freq,10)

set.seed(123)

Merge10 <- kmeans(Bin\_Freq,8)

set.seed(123)

Merge11 <- kmeans(PCN\_Freq,6)

set.seed(123)

Merge12 <- kmeans(State\_Freq,6)

##Appending Clusters to orginal Data##

Append1<-as.data.frame(cbind(Drug\_Freq,c1=Merge1$cluster))

Drug.Final<-as.data.frame(Append1[,-c(2,3)])

Drug.Final<-rename(Drug.Final,c("Var1"="Drug"))

Append2<-as.data.frame(cbind(DrugSubClass\_Freq,c2=Merge2$cluster))

DrugSubClass.Final<-as.data.frame(Append2[,-c(2,3)])

DrugSubClass.Final<-rename(DrugSubClass.Final,c("Var1"="DrugSubClass"))

Append3<-as.data.frame(cbind(DrugClass\_Freq,c3=Merge3$cluster))

Drugclass.Final<-as.data.frame(Append3[,-c(2,3)])

Drugclass.Final<-rename(Drugclass.Final,c("Var1"="DrugClass"))

Append4<-as.data.frame(cbind(Drug\_Chemical\_Name\_Freq,c4=Merge4$cluster))

Drug\_Chemical\_Name.Final<-as.data.frame(Append4[,-c(2,3)])

Drug\_Chemical\_Name.Final<-rename(Drug\_Chemical\_Name.Final,c("Var1"="Drug\_Chemical\_Name"))

Append5<-as.data.frame(cbind(GPI\_Freq,c5=Merge5$cluster))

GPI.Final<-as.data.frame(Append5[,-c(2,3)])

GPI.Final<-rename(GPI.Final,c("Var1"="GPI"))

Append6<-as.data.frame(cbind(NDC\_Freq,c6=Merge6$cluster))

NDC.Final<-as.data.frame(Append6[,-c(2,3)])

NDC.Final<-rename(NDC.Final,c("Var1"="NDC"))

Append7<-as.data.frame(cbind(DrugGroup\_Freq,c7=Merge7$cluster))

DrugGroup.Final<-as.data.frame(Append7[,-c(2,3)])

DrugGroup.Final<-rename(DrugGroup.Final,c("Var1"="DrugGroup"))

Append8<-as.data.frame(cbind(DoctorID\_Freq,c8=Merge8$cluster))

DoctorID.Final<-as.data.frame(Append8[,-c(2,3)])

DoctorID.Final<-rename(DoctorID.Final,c("Var1"="DoctorID"))

Append9<-as.data.frame(cbind(RxGroupId\_Freq,c9=Merge9$cluster))

RxGroupId.Final<-as.data.frame(Append9[,-c(2,3)])

RxGroupId.Final<-rename(RxGroupId.Final,c("Var1"="RxGroupId"))

Append10<-as.data.frame(cbind(Bin\_Freq,c10=Merge10$cluster))

Bin.Final<-as.data.frame(Append10[,-c(2,3)])

Bin.Final<-rename(Bin.Final,c("Var1"="Bin"))

Append11<-as.data.frame(cbind(PCN\_Freq,c11=Merge11$cluster))

PCN.Final<-as.data.frame(Append11[,-c(2,3)])

PCN.Final<-rename(PCN.Final,c("Var1"="PCN"))

Append12<-as.data.frame(cbind(State\_Freq,c12=Merge12$cluster))

State.Final<-as.data.frame(Append12[,-c(2,3)])

State.Final<-rename(State.Final,c("Var1"="State"))

##Building Models###

Data.Merge<-merge(x=Data,y=Drug.Final,by.x="Drug",by.y="Drug",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=DrugSubClass.Final,by.x="DrugSubClass",by.y="DrugSubClass",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=Drugclass.Final,by.x="DrugClass",by.y="DrugClass",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=Drug\_Chemical\_Name.Final,by.x="Drug\_Chemical\_Name",by.y="Drug\_Chemical\_Name",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=GPI.Final,by.x="GPI",by.y="GPI",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=NDC.Final,by.x="NDC",by.y="NDC",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=DrugGroup.Final,by.x="DrugGroup",by.y="DrugGroup",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=DoctorID.Final,by.x="DoctorID",by.y="DoctorID",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=RxGroupId.Final,by.x="RxGroupId",by.y="RxGroupId",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=Bin.Final,by.x="Bin",by.y="Bin",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=PCN.Final,by.x="PCN",by.y="PCN",all.x="TRUE")

Data.Merge<-merge(x=Data.Merge,y=State.Final,by.x="State",by.y="State",all.x="TRUE")

Data.Merge<-subset(Data.Merge,select=-c(Drug,DrugSubClass,DrugClass,Drug\_Chemical\_Name,GPI,

NDC,DrugGroup,DoctorID,RxGroupId,Bin,PCN,State))

summary(Data.Merge)

str(Data.Merge)

##Finding Correlation##

Data.Merge\_N<-subset(Data.Merge,select =-c(Target))

c<-cor(Data.Merge\_N,use="complete.obs",method="pearson")

cor<-as.table(sort(c))

head(cor)

tail(cor)

###############################

names(Data.Merge)

names(Data)

str(Data.Merge)

Data.Merge\_N<-subset(Data.Merge,select=-c(Target))

str(Data.Merge\_N)

cor(Data.Merge\_N,method="pearson")

####Dividing Training & Testing Data ##

#Split the data into train and test data sets (70:30 ratio)

rows = seq(1,nrow(Data.Merge),1)

set.seed(123)

trainRows = sample(rows,(70\*nrow(Data.Merge))/100)

train = Data.Merge[trainRows,]

test = Data.Merge[-trainRows,]

#Converting to Factor Attributes#

train.cart<-data.frame(apply(train,2,function(x){as.factor(x)}))

#Converting to Factor Attributes#

test.cart<-data.frame(apply(test,2,function(x){as.factor(x)}))

##Decision Trees using C50##

library(C50)

dtC50= C5.0(Target~ .,

data = train.cart,

rules=TRUE)

help(C5.0)

C5imp(dtC50, pct=TRUE)

dtC50$rules

summary(dtC50)

# predict C5.0 model on test data

Predict\_C50 <- predict.C5.0(object = dtC50, newdata = test.cart)

Predict\_C50

table(Predict\_C50)

# write rules into text file

write(capture.output(summary(dtC50)), "Model\_C50\_Rules.txt")

# confusion matrix for prediction#

#using dataset "test" to have correct formation of cofusion matrix#

CM\_C50 <- table(test$Target, Predict\_C50)

CM\_C50

Accuracy\_C50 <- sum(diag(CM\_C50))/sum(CM\_C50)\*100

Accuracy\_C50

Recall\_C50 <- CM\_C50[2,2]/(CM\_C50[2,1]+CM\_C50[2,2])\*100

Recall\_C50

Precision\_C50 <- CM\_C50[2,2]/(CM\_C50[1,2]+CM\_C50[2,2])\*100

Precision\_C50

###########################################################################################

##Random Forest##

library(randomForest)

set.seed(123)

rf <- randomForest(Target ~ ., data=train.cart, keep.forest=TRUE, ntree=30)

round(importance(rf), 2)

# test predict using model#

Predict\_RF <- predict(object = rf, newdata = test.cart, type = "response",

norm.votes = TRUE)

#using dataset "test" to have correct formation of cofusion matrix#

CM\_RF <- table(test$Target, Predict\_RF)

CM\_RF

Accuracy\_RF <- sum(diag(CM\_RF))/sum(CM\_RF)\*100

Accuracy\_RF

Recall\_RF <- ((CM\_RF[2,2])/(CM\_RF[2,2]+CM\_RF[2,1])\*100)

Recall\_RF

Precision\_RF <- ((CM\_RF[2,2])/(CM\_RF[2,2]+CM\_RF[1,2])\*100)

Precision\_RF

##################################################################################

## Naive Bayes##

library(e1071)

model <- naiveBayes(train.cart$Target~., data = train.cart)

model

Predict\_NB = predict(model, test.cart)

Predict\_NB

###Confusion Matrix####

#using dataset "test" to have correct formation of cofusion matrix#

CM\_NB <- table(test$Target,Predict\_NB)

CM\_NB

Accuracy\_NB <- sum(diag(CM\_NB))/sum(CM\_NB)\*100

Accuracy\_NB

Recall\_NB <- ((CM\_NB[2,2])/(CM\_NB[2,2]+CM\_NB[2,1])\*100)

Recall\_NB

Precision\_NB <- ((CM\_NB[2,2])/(CM\_NB[2,2]+CM\_NB[1,2])\*100)

Precision\_NB

# prepare data frame for ensemble model

Data\_Ensemble <- data.frame(cbind(Predict\_C50, Predict\_RF,

Predict\_NB, Response = test.cart$Target))

# rename ensemble data frame

names(Data\_Ensemble) <- c("Decision", "RandomForest","naiveBayes","Response")

# study data ensemble

summary(Data\_Ensemble)

str(Data\_Ensemble)

# factorise attribute

Data\_Ensemble$Response <- as.factor(Data\_Ensemble$Response)

# check for missing values

sum(is.na(Data\_Ensemble))

# ///////////////////// BUILD ENSEMBLE MODEL /////////////////

# run a logistic regression model on ensemble

Ensemble\_model <- glm(formula = Response ~ ., data = Data\_Ensemble, family = binomial)

# study ensemble model

summary(Ensemble\_model)

# predict ensemble model

predict\_ensemble <- predict.glm(object = Ensemble\_model, newdata = Data\_Ensemble[,-5],

type = 'response')

# recode logistic attribute

predict\_ensemble <- ifelse(test = predict\_ensemble > 0.5, yes = 1, no = 0)

############################## Confusion Matrix ###########################

ConfusionMatrix <- table(test$Target, predict\_ensemble)

ConfusionMatrix

Accuracy <- sum(diag(ConfusionMatrix))/sum(ConfusionMatrix)\*100

Recall <- ((ConfusionMatrix[2,2])/(ConfusionMatrix[2,2]+ConfusionMatrix[2,1])\*100)

Precision <- ((ConfusionMatrix[2,2])/(ConfusionMatrix[2,2]+ConfusionMatrix[1,2])\*100)

Accuracy

Recall

Precision

######################################### END ##############################################