CS-513 Knowledge Dis & Data Mining

Project Report

Prediction of H1B Visa Petitions Decision

Sumit Gupta

10441745

Contents

Name: Sumit Gupta
CWID: 10441745

H1B Visa Petitions Decision Prediction

Introduction:

The H-1B is a visa in the United States under the Immigration and Nationality Act, section 101(a)(15)(H) that allows U.S. employers to temporarily employ foreign workers in specialty occupations. A specialty occupation requires the application of specialized knowledge and a bachelor's degree or the equivalent of work experience.

Packages Required:

To analyze this data, we will use the following R packages:
# Install All the required Packages
install.packages("readr")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("lubridate")
install.packages("DT")
install.packages("tidyr")
install.packages("stats")

```
# Including Libraries
library(class)
library(readr)      #For reading csv file
library(stats)
library(dplyr)      #For Data transformation
library(ggplot2)    #For graphics
library(DT)         #For displaying data with formatting
library(tidyr)      #For data transformation

# Delete all the objects from your R- environment.
rm(list=ls())
```

Data:

Data is collected from UNITED STATES DEPARTMENT OF LABOR carry out its responsibility for the processing of labor certification and labor attestation applications, the Office of Foreign Labor Certification (OFLC) generates program data that is essential both for internal assessment of program effectiveness and for providing the Department's external stakeholders with useful information about the immigration programs administered by OFLC. It is always interesting to find out hidden details about the H-1B petitions from the data that is available at OFLC. I have downloaded the data of year 2015, 2016,2017 and 2018 for analysis and applying my model. Data was available in .xlxs format. I converted it to .csv so that I can load the data to R and perform analysis and implement prediction algorithms.

```
# Read CSV file
efile_data_2015 <- read_csv("C:\\Users\\sumit\\OneDrive\\Desktop\\KDD\\Raw Data\\H-1B_Disclosure_Data_FY15.csv")

## Parsed with column specification:
## cols(
```


Name: Sumit Gupta
CWID: 10441745

```
##   .default = col_character(),
##   EMPLOYER_PHONE = col_double(),
##   EMPLOYER_PHONE_EXT = col_double(),
##   NAIC_CODE = col_double(),
##   PREVAILING_WAGE = col_double(),
##   PW_WAGE_SOURCE_YEAR = col_double()
## )

## See spec(...) for full column specifications.

efile_data_2016 <- read_csv("C:\\Users\\sumit\\OneDrive\\Desktop\\KDD\\Raw Data\\H-1B_Disclosure_Data_FY16.csv")

## Parsed with column specification:
## cols(
##   .default = col_character(),
##   EMPLOYER_PHONE = col_double(),
##   NAIC_CODE = col_double(),
##   TOTAL_WORKERS = col_double(),
##   FULL_TIME_POSITION = col_logical(),
##   PREVAILING_WAGE = col_number(),
##   PW_SOURCE_YEAR = col_double(),
##   WAGE_RATE_OF_PAY_FROM = col_number(),
##   WAGE_RATE_OF_PAY_TO = col_number()
## )

efile_data_2017 <- read_csv("C:\\Users\\sumit\\OneDrive\\Desktop\\KDD\\Raw Data\\H-1B_Disclosure_Data_FY17.csv")

## Parsed with column specification:
## cols(
##   .default = col_character(),
##   EMPLOYER_PHONE = col_double(),
##   NAICS_CODE = col_double(),
##   TOTAL_WORKERS = col_double(),
##   NEW_EMPLOYMENT = col_double(),
##   CONTINUED_EMPLOYMENT = col_double(),
##   CHANGE_PREVIOUS_EMPLOYMENT = col_double(),
##   NEW_CONCURRENT_EMPLOYMENT = col_double(),
##   CHANGE_EMPLOYER = col_double(),
##   AMENDED_PETITION = col_double(),
##   PREVAILING_WAGE = col_number(),
##   PW_SOURCE_YEAR = col_double(),
##   WAGE_RATE_OF_PAY_FROM = col_number(),
##   WAGE_RATE_OF_PAY_TO = col_number(),
##   PUBLIC_DISCLOSURE_LOCATION = col_logical()
## )
## See spec(...) for full column specifications.

efile_data_2018 <- read_csv("C:\\Users\\sumit\\OneDrive\\Desktop\\KDD\\Raw Data\\H-1B_Disclosure_Data_FY18.csv")

## Parsed with column specification:
## cols(
##   .default = col_character(),
```

```
##   NAICS_CODE = col_double(),
##   TOTAL_WORKERS = col_double(),
##   NEW_EMPLOYMENT = col_double(),
##   CONTINUED_EMPLOYMENT = col_double(),
##   CHANGE_PREVIOUS_EMPLOYMENT = col_double(),
##   NEW_CONCURRENT_EMP = col_double(),
##   CHANGE_EMPLOYER = col_double(),
##   AMENDED_PETITION = col_double(),
##   PREVAILING_WAGE = col_number(),
##   PW_SOURCE_YEAR = col_double(),
##   WAGE_RATE_OF_PAY_FROM = col_number(),
##   WAGE_RATE_OF_PAY_TO = col_number(),
##   PUBLIC_DISCLOSURE_LOCATION = col_logical()
## )
## See spec(...) for full column specifications.
```

Data Preparation.

I created **FIN_YEAR** variable and assign it a value 2015, 2016, 2017 or 2018 based on the financial year a decision is made on the case by the authorities.

```
# we create FIN_YEAR variable and assign it a value of either 2016, 2017,
2018 OR 2019
efile_data_2015['FIN_YEAR'] <- 2015
efile_data_2016['FIN_YEAR'] <- 2016
efile_data_2017['FIN_YEAR'] <- 2017
efile_data_2018['FIN_YEAR'] <- 2018
```

Variable selection and data merging:

I have attached the File Structure pdf that was available at OFLC to understand the column attributes. I found that there are 40 columns in 2015 and 2016 whereas there are 52 columns in 2017 and 2018. one new column in added by me for all the data frames. Before we start cleaning the data, we need to merge the data frame to form one huge file that will be used for cleaning and implementing Prediction model. After merging all the 4 data frames we found that there are 31 matching columns in each of the data frames.

```
# Changing the colum name to match all the 4 data frames
names(efile_data_2015)[which(names(efile_data_2015) == "H-1B_DEPENDENT")]
<- "H1B_DEPENDENT"
names(efile_data_2016)[which(names(efile_data_2016) == "H-1B_DEPENDENT")]
<- "H1B_DEPENDENT"

#merging datasets and retaining the common columns
common_cols1<- intersect(colnames(efile_data_2015),colnames(efile_data_201
6))
efile1 <- rbind(efile_data_2015[common_cols1], efile_data_2016[common_cols
1])

common_cols2<- intersect(colnames(efile_data_2017),colnames(efile_data_201
8))
efile2 <- rbind(efile_data_2017[common_cols2], efile_data_2018[common_cols
2])

common_cols<- intersect(colnames(efile1),colnames(efile2))
efile_data <- rbind(efile1[common_cols], efile2[common_cols])
```

Cleaning the R-Rnviroment before moving forward.

```
# cleaning the environment
rm(efile_data_2015,efile_data_2016,efile_data_2017,efile_data_2018,efile1,
efile2,common_cols,common_cols1,common_cols2)
#View(efile_data)
```


Name: Sumit Gupta
CWID: 10441745

Data Cleaning:

The raw merged data is messy, and some cleaning steps needs to be performed on the data table before moving forward.

- Records are filtered out to retain only those records corresponding to **H-1B** data
- The merged dataset we have has only **10 variables** that are of interest for our analysis and **drop** else.
- We want to know if an employee is a **full time** or a **part time** worker
- We do the scaling of the **hourly, Weekly, Bi-Weekly and monthly** salaries to a **yearly pay**
- Remove the row with **empty cell**
- Records with **extreme** PREVAILING_WAGE values are discarded. (RANGE IS 25K TO 200K)

```
# Performing the cleaning of data
df1 <- efile_data %>%
  # records are filtered out to retain only those records corresponding to
H-1B data
  filter(VISA_CLASS %in% "H-1B") %>%

  # The merged dataset we have has only 10 variables that are of interest
for our analysis and drop else
  select(c("CASE_STATUS","DECISION_DATE","EMPLOYER_NAME","JOB_TITLE","SOC_
NAME"),PREVAILING_WAGE:PW_UNIT_OF_PAY,
         H1B_DEPENDENT,WORKSITE_CITY,WORKSITE_STATE,"FIN_YEAR") %>%

  # We want to know if an employee is a full time or a part time worker
  mutate( FULL_TIME = case_when(PW_UNIT_OF_PAY == 'Year' ~ 'Y', PW_UNIT_OF
_PAY != 'Year' ~ 'N'),DECISION_DATE =
            as.numeric(as.Date(DECISION_DATE, format = "%d-%m-%Y")))%>%

  # We do the scaling of the hourly, Weekly, Bi-Weekly and monthly salarie
s to an yearly pay
  mutate(PREVAILING_WAGE = case_when(PW_UNIT_OF_PAY == 'Hour' ~ PREVAILING
_WAGE*2087,PW_UNIT_OF_PAY == 'Year' ~ PREVAILING_WAGE,
                                     PW_UNIT_OF_PAY=='Month'~PREVAILING_WA
GE*12,PW_UNIT_OF_PAY=='Bi-Weekly'~PREVAILING_WAGE*21)) %>%

  select(-DECISION_DATE,-PW_UNIT_OF_PAY) %>%

  # Remove the row with empty cell
  na.omit(df1) %>%

  # Records with extreme PREVAILING_WAGE values are discarded.(rANGE IS 25
K TO 200K)
  filter(PREVAILING_WAGE > 25000 & (PREVAILING_WAGE < 200000))
```


Name: Sumit Gupta
CWID: 10441745

Filtering the data based on most frequently occurring occupations

```
df<-df1
df$SOC_NAME1<-NA
df$SOC_NAME1[grep("engineer",df$SOC_NAME, ignore.case = T)]<-"ENGINEER"
df$SOC_NAME1[grep("manager",df$SOC_NAME, ignore.case = T)]<-"MANAGER"
df$SOC_NAME1[grep("technician",df$SOC_NAME, ignore.case = T)]<-"TECHNICIAN
"
df$SOC_NAME1[grep("teacher",df$SOC_NAME, ignore.case = T)]<-"TEACHER"
df$SOC_NAME1[grep("executive",df$SOC_NAME, ignore.case = T)]<-"EXECUTIVE"
df$SOC_NAME1[grep("accountant",df$SOC_NAME, ignore.case = T)]<-"ACCOUNTANT
"
df$SOC_NAME1[grep("Developer",df$SOC_NAME, ignore.case = T)]<-"DEVELOPERS"
df$SOC_NAME1[grep("computer",df$SOC_NAME, ignore.case = T)]<-"SOFTWARE"
df$SOC_NAME<-df$SOC_NAME1
df<-na.omit(df)
df1<-df[1:10]
#View(df1)
```

Data Summary

The H-1B dataset has 10 variables

```
summary.default(df1)

##                   Length  Class  Mode
## CASE_STATUS       1980608 -none- character
## EMPLOYER_NAME     1980608 -none- character
## JOB_TITLE         1980608 -none- character
## SOC_NAME          1980608 -none- character
## PREVAILING_WAGE   1980608 -none- numeric
## H1B_DEPENDENT     1980608 -none- character
## WORKSITE_CITY     1980608 -none- character
## WORKSITE_STATE    1980608 -none- character
## FIN_YEAR          1980608 -none- numeric
## FULL_TIME         1980608 -none- character
```

Storing of DATA:

```
# Storing the cleaned data to CSV
write.csv(df1,"C:\\Users\\sumit\\OneDrive\\Desktop\\KDD\\cleaned_data.csv"
, row.names = FALSE)
```


Name: Sumit Gupta
CWID: 10441745

Exploratory Data Analysis

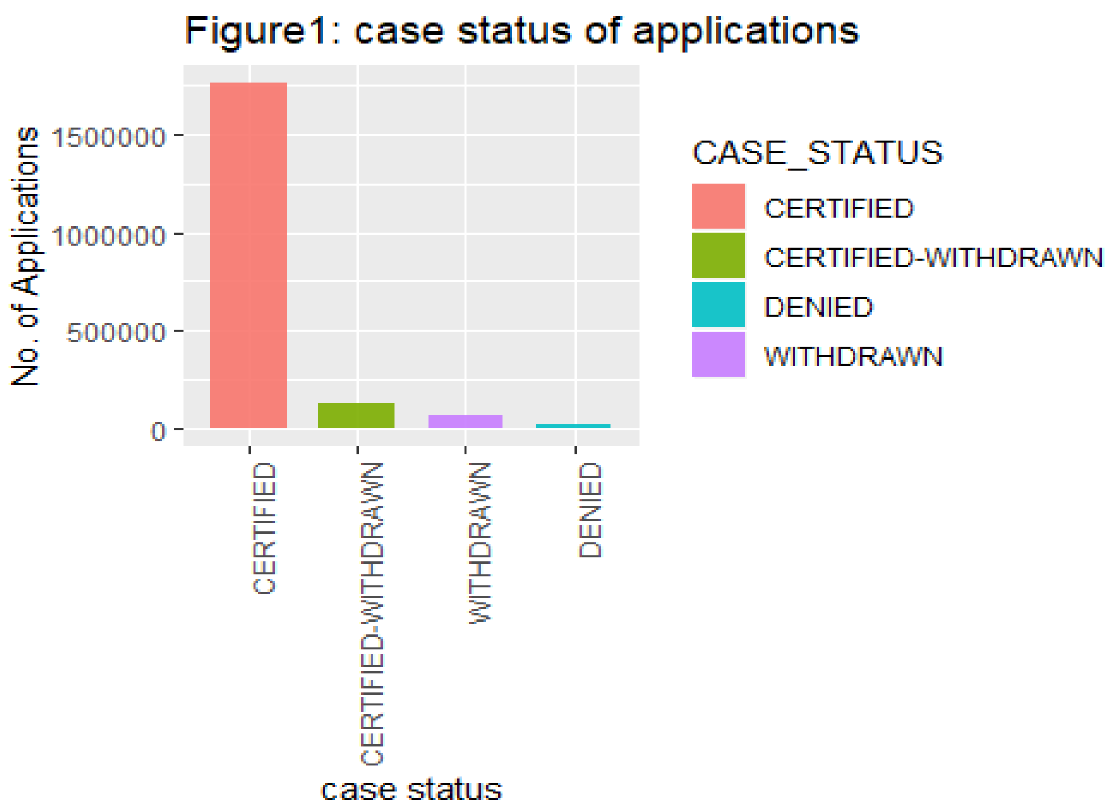
Here are some of the graphical analysis that could be interesting to look at. In order to look at the graphical representation of the data it is better to look at top 10 submissions for all the data that is data of all 4 years at once then we will analyze data of each year separately.

```
# Function to handle all the data at once
top_n_records <- function(col_name,n_rec) {
  col_name <- as.name(col_name)
  df <- df1 %>%
    group_by_("FIN_YEAR",col_name) %>%
    summarise(num_apps = n()) %>%
    arrange(desc(num_apps)) %>%
    slice(1:n_rec)
}


######function to extract top 5 records by a parameter based on year######

top_5_recordsby_year <- function(col_name) {
  col_name <- as.name(col_name)
  df <- df1 %>%
    group_by_("FIN_YEAR",col_name) %>%
    summarise(num_apps = n())
  df <- spread(df,key = FIN_YEAR, value = num_apps)
  df$num_apps <- df$`2015` + df$`2016` + df$`2017` + df$`2018`
  df <- arrange(df,desc(num_apps))
  df <- df[1:5,1:6]
  df <-  gather(df, key = "FIN_YEAR",value = "num_apps",2:5)
}
```


Name: Sumit Gupta
CWID: 10441745

Top H1B applicant states

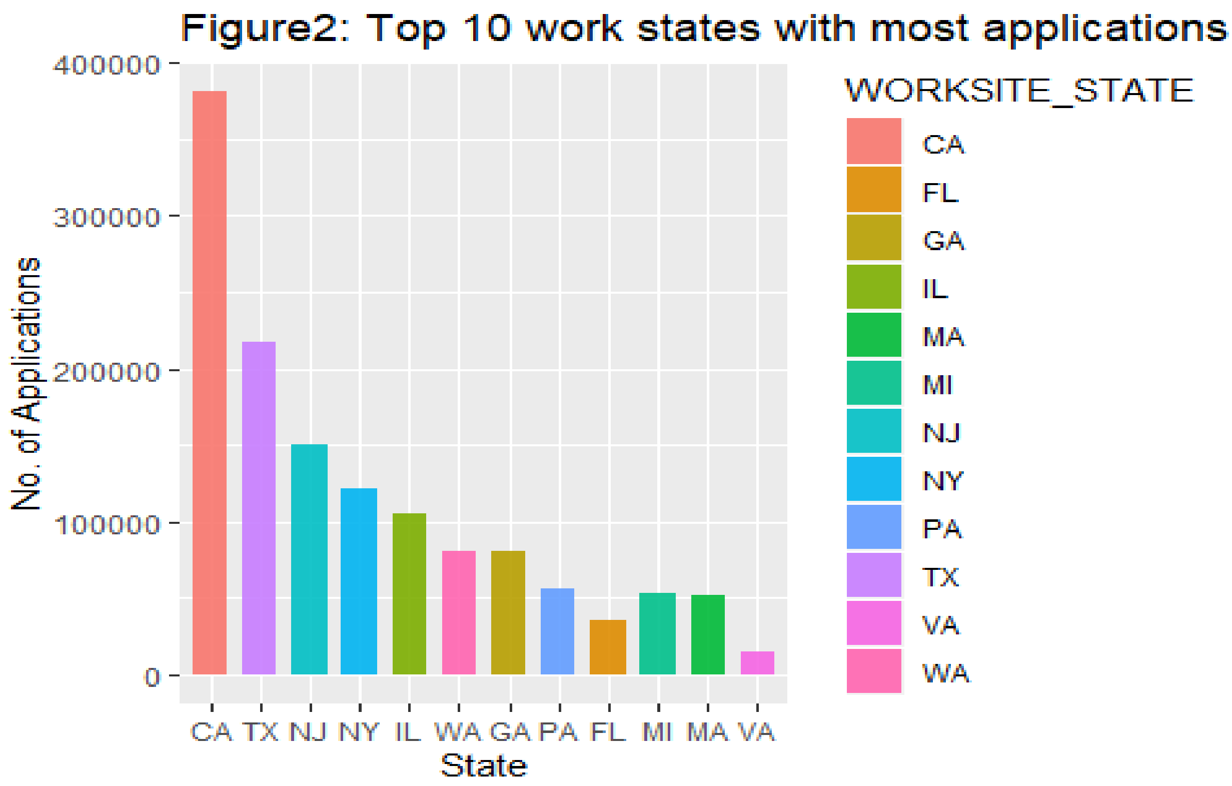
```
#### H1B Visa Case Status
ggplot(top_n_records("CASE_STATUS",10),aes(x = reorder(CASE_STATUS,-num_ap
ps),y = num_apps, fill = CASE_STATUS)) +
  geom_bar(stat = "identity", alpha = 0.9, width = 0.7) +
  ggtitle("Figure1: case status of applications") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  labs(x = "case status", y = "No. of Applications")

## Warning: group_by_() is deprecated.
## Please use group_by() instead
##
## The 'programming' vignette or the tidyeval book can help you
## to program with group_by() : https://tidyeval.tidyverse.org
## This warning is displayed once per session.
```



Name: Sumit Gupta
CWID: 10441745

Top H1B applicant States

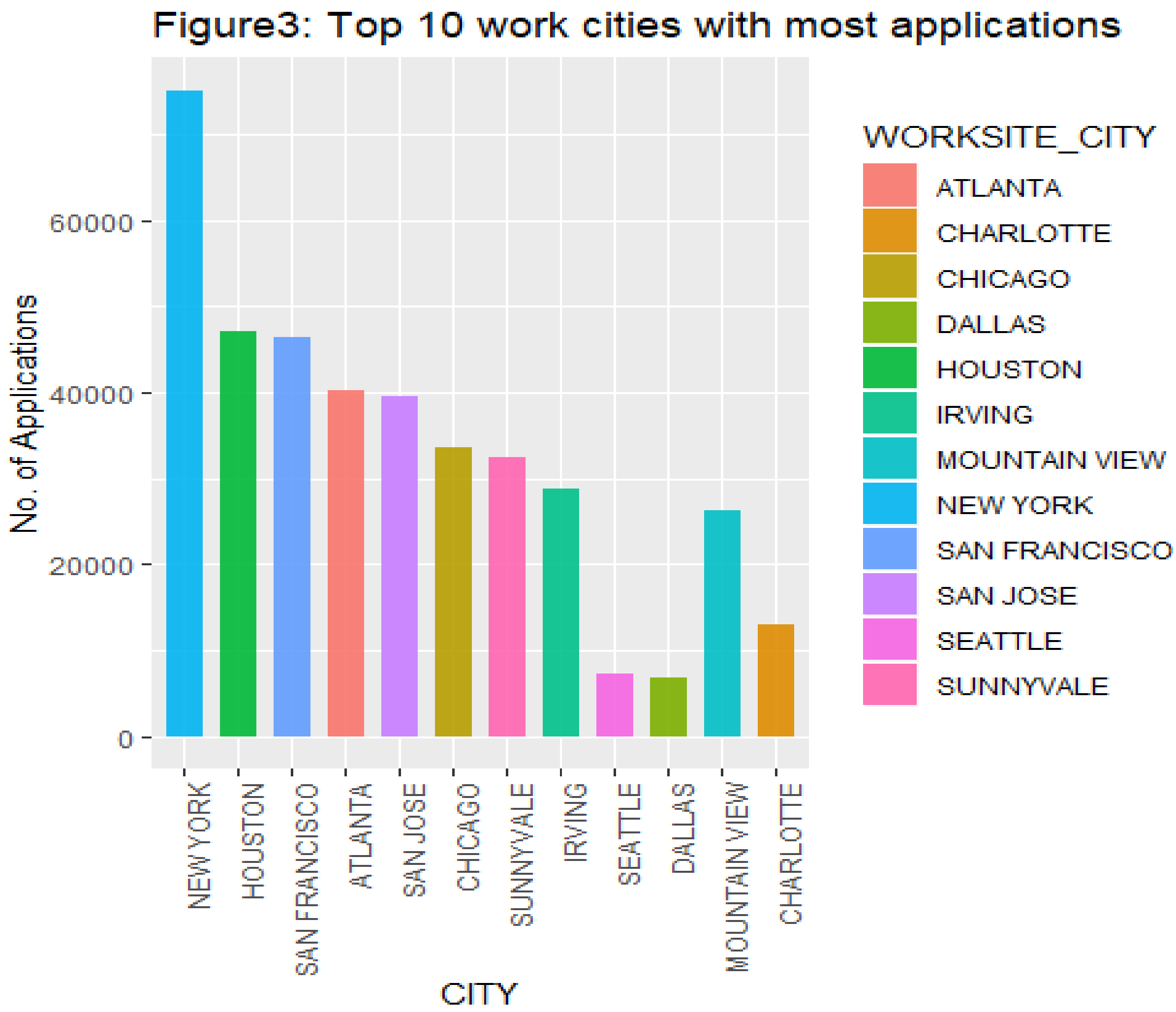
```
####  Top H1B applicant states
ggplot(top_n_records("WORKSITE_STATE",10),aes(x = reorder(WORKSITE_STATE,-
num_apps),y = num_apps, fill = WORKSITE_STATE)) +
  geom_bar(stat = "identity", alpha = 0.9, width = 0.7) +
  ggtitle("Figure2: Top 10 work states with most applications") +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))+
  labs(x = "State", y = "No. of Applications")
```



Name: Sumit Gupta
CWID: 10441745

Top H1B applicant cities

```
#### Top H1B applicant cities
ggplot(top_n_records("WORKSITE_CITY",10),aes(x = reorder(WORKSITE_CITY,-nu
m_apps),y = num_apps, fill = WORKSITE_CITY)) +
  geom_bar(stat = "identity", alpha = 0.9, width = 0.7) +
  ggtitle("Figure3: Top 10 work cities with most applications") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(x = "CITY", y = "No. of Applications")
```

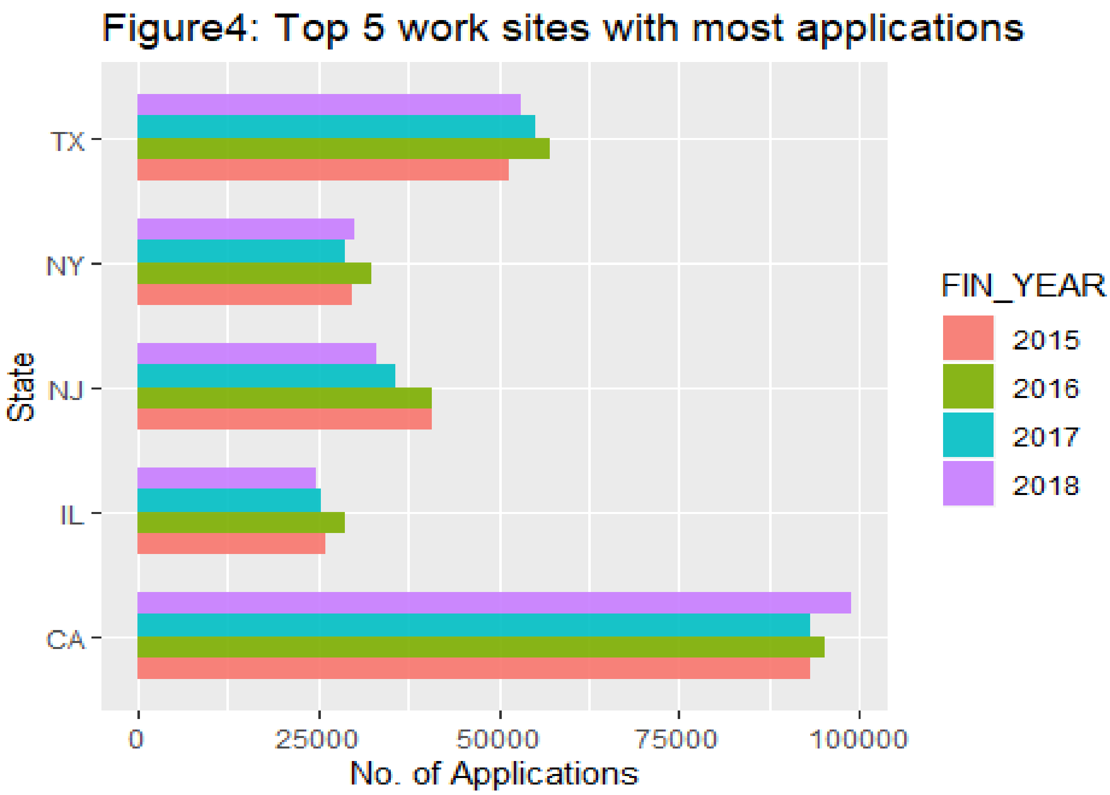


Name: Sumit Gupta
CWID: 10441745

In order to look at the graphical representation of the data it is better to look at top 5 submissions based on each year.

Top H1B applicant states by year

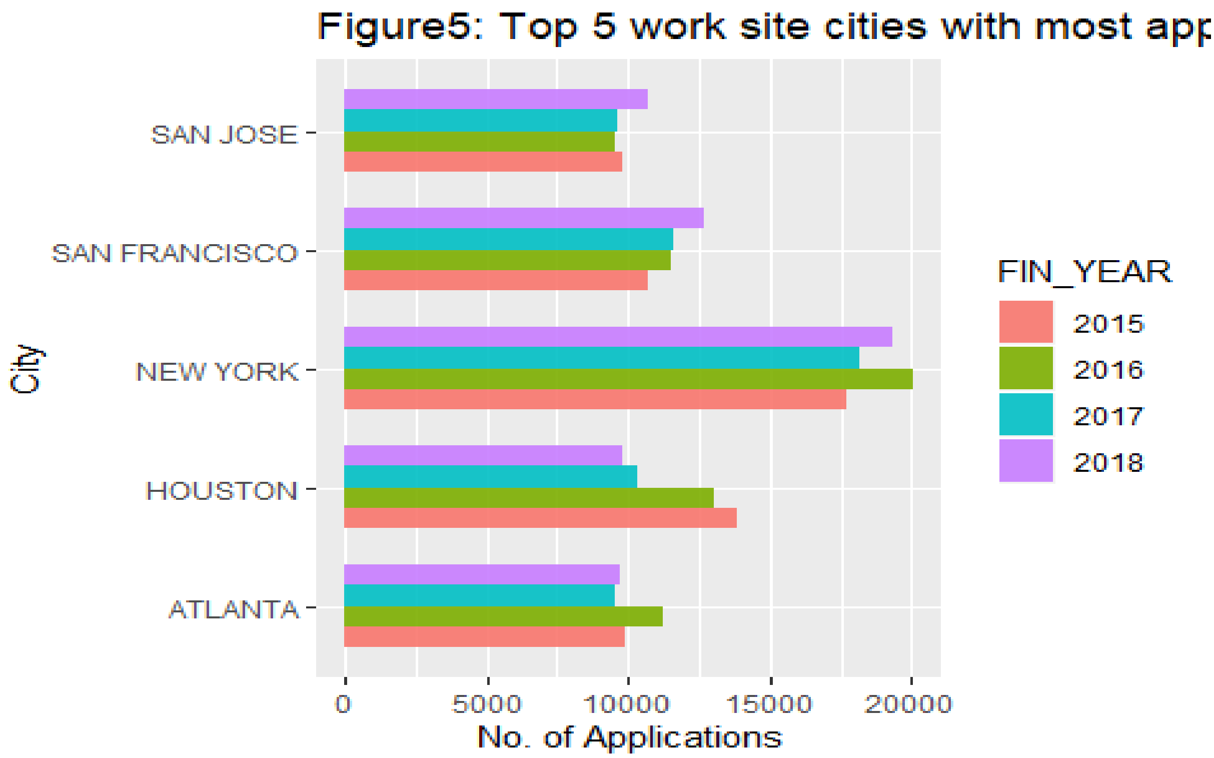
```
#### Top H1B applicant states by year
ggplot(top_5_recordsby_year("WORKSITE_STATE"),aes(x = WORKSITE_STATE, y =
num_apps,fill = FIN_YEAR)) +
  geom_bar(stat = "identity", position = position_dodge() , alpha = 0.9,
width = 0.7) +
  coord_flip() +
  ggtitle("Figure4: Top 5 work sites with most applications") +
  labs(x = "State", y = "No. of Applications")
```



Name: Sumit Gupta
CWID: 10441745

Top H1B applicant cities by year

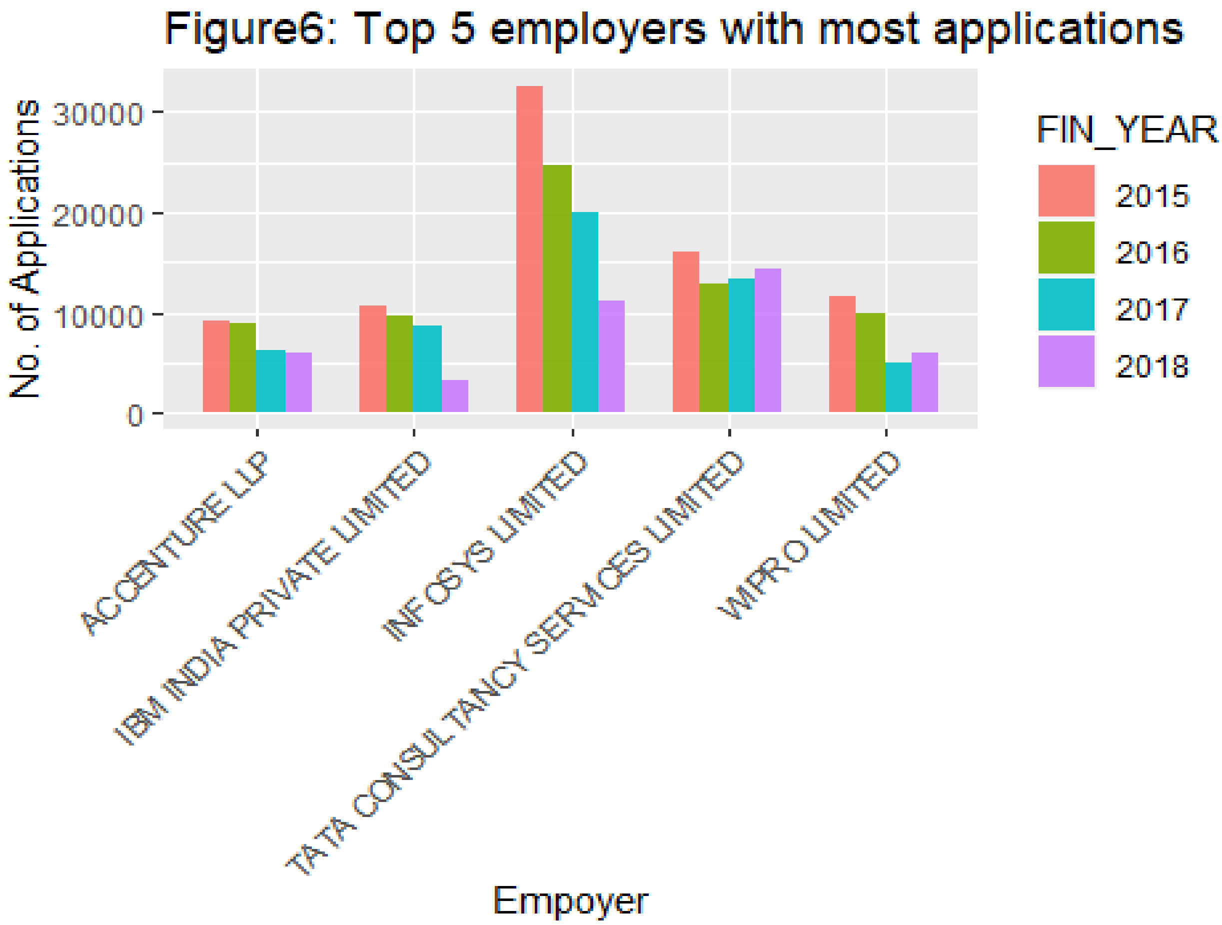
```
#### Top H1B applicant cities by year
ggplot(top_5_recordsby_year("WORKSITE_CITY"),aes(x = WORKSITE_CITY, y = nu
m_apps,fill = FIN_YEAR)) +
  geom_bar(stat = "identity", position = position_dodge() , alpha = 0.9,
width = 0.7) +
  coord_flip() +
  ggtitle("Figure5: Top 5 work site cities with most applications") +
  labs(x = "City", y = "No. of Applications")
```



Name: Sumit Gupta
CWID: 10441745

Popular H1B Visa Sponsors by year

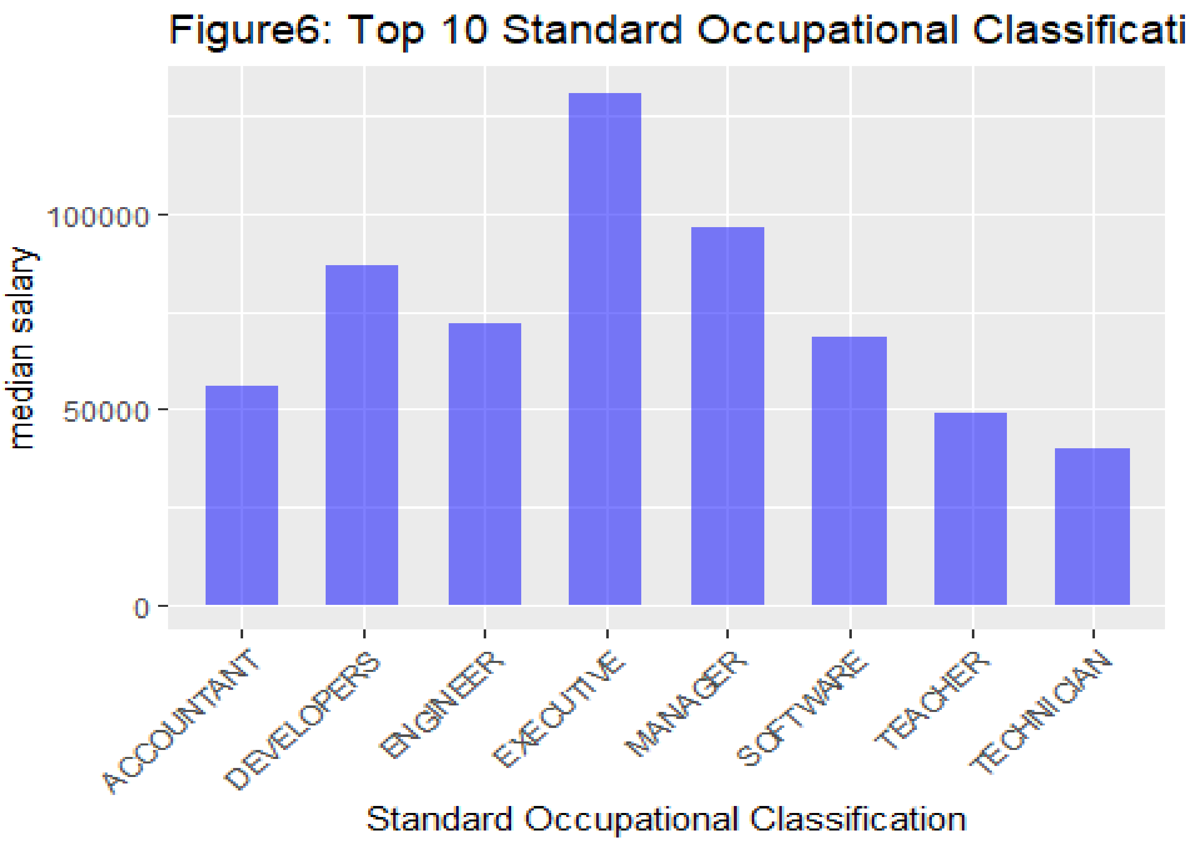
```
#### Popular H1B Visa Sponsors by year
ggplot(top_5_recordsby_year("EMPLOYER_NAME"),aes(x = EMPLOYER_NAME, y = nu
m_apps,fill = FIN_YEAR)) +
  geom_bar(stat = "identity", position = position_dodge() , alpha = 0.9,
width = 0.7) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  ggtitle("Figure6: Top 5 employers with most applications") +
  labs(x = "Empoyer", y = "No. of Applications")
```



Name: Sumit Gupta
CWID: 10441745

Occupations with highest wages

```
#####Top 8 socs with the highest wages
top_8_soc_highest_wage <- df1 %>%
  group_by(SOC_NAME) %>%
  summarise(median_wage = median(PREVAILING_WAGE)) %>%
  arrange(desc(median_wage)) %>%
  slice(1:8) %>%
  select(SOC_NAME, median_wage)

ggplot(top_8_soc_highest_wage,aes(x = SOC_NAME, y = median_wage)) +
  geom_bar(stat = "identity", fill = "Blue", alpha = 0.5,  width = 0.6) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  ggtitle("Figure6: Top 10 Standard Occupational Classification names and
their median pay") +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))+
  labs(x = "Standard Occupational Classification", y = "median salary")
```



Name: Sumit Gupta
CWID: 10441745

Apply Different Models to our Data:

Factorizing data to convert string to factors.

```
df1$CASE_STATUS<-factor(ifelse(df1$CASE_STATUS %in% c("CERTIFIED"),"1","0"
))
#df1$CASE_STATUS <-factor(df1$CASE_STATUS, levels = c('CERTIFIED','DENIED'
,'WITHDRAWN','CERTIFIED-WITHDRAWN'), labels = c(1,2,3,4))
df1$EMPLOYER_NAME <- factor(df1$EMPLOYER_NAME)
df1$JOB_TITLE <- factor(df1$JOB_TITLE)
df1$SOC_NAME <- factor(df1$SOC_NAME)
df1$H1B_DEPENDENT <- factor(df1$H1B_DEPENDENT)
df1$WORKSITE_CITY <- factor(df1$WORKSITE_CITY)
df1$WORKSITE_STATE <- factor(df1$WORKSITE_STATE)
df1$FULL_TIME <- factor(df1$FULL_TIME)
df1$FIN_YEAR <- factor(df1$FIN_YEAR)
str(df1)

## Classes 'tbl_df', 'tbl' and 'data.frame':    1980608 obs. of  10 variab
les:
##  $ CASE_STATUS    : Factor w/ 2 levels "0","1": 2 1 2 2 2 2 1 2 1 2 ...
##  $ EMPLOYER_NAME  : Factor w/ 103723 levels "\"K\" LINE AMERICA",..: 96
155 64358 66272 33150 99038 99038 44288 99038 46914 99038 ...
##  $ JOB_TITLE      : Factor w/ 178266 levels "-  SAP PO/PI CONSULTANT",.
.: 8811 103402 29881 35819 158509 96978 30595 77978 95698 121607 ...
##  $ SOC_NAME       : Factor w/ 8 levels "ACCOUNTANT","DEVELOPERS",..: 7
6 6 6 2 6 1 2 6 6 ...
##  $ PREVAILING_WAGE: num  42860 73965 65998 96907 133976 ...
##  $ H1B_DEPENDENT  : Factor w/ 2 levels "N","Y": 1 1 1 1 1 1 1 1 2 1 ...
##  $ WORKSITE_CITY  : Factor w/ 11947 levels " BOTHELL"," CHICAGO",..: 73
39 9211 4807 6273 7878 7878 10447 7878 6519 7878 ...
##  $ WORKSITE_STATE : Factor w/ 58 levels "AK","AL","AR",..: 41 5 50 52 5
5 12 5 9 5 ...
##  $ FIN_YEAR       : Factor w/ 4 levels "2015","2016",..: 1 1 1 1 1 1 1
1 1 1 ...
##  $ FULL_TIME      : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 1 2 2 2 ...
##  - attr(*, "na.action")= 'omit' Named int  6 18 21 24 25 26 35 42 43 44
...
##   ..- attr(*, "names")= chr  "6" "18" "21" "24" ...
```

Dividing the data into training and test set:

70% of random data is training data. 30% of random data is test data.

```
# Divide data in training_set and test_set
#remove(list=c("training_set", "sam", "test_set"))
set.seed(123)
sam<- sort(sample(nrow(df1),as.integer(0.70*nrow(df1))))
training_set<- df1[sam,]
summary(training_set)

##  CASE_STATUS                           EMPLOYER_NAME
##  0: 153847   INFOSYS LIMITED                    :  62073
##  1:1232578   TATA CONSULTANCY SERVICES LIMITED:  39465
```


Name: Sumit Gupta
CWID: 10441745

```
##              WIPRO LIMITED                    :  22741
##              IBM INDIA PRIVATE LIMITED        :  22347
##              CAPGEMINI AMERICA INC            :  21507
##              ACCENTURE LLP                    :  20950
##              (Other)                          :1197342
##                     JOB_TITLE             SOC_NAME      PREVAILING_WAGE
##  PROGRAMMER ANALYST      : 117304   SOFTWARE   :722616   Min.   : 25002
##  SOFTWARE ENGINEER       :  81230   DEVELOPERS:450628   1st Qu.: 61485
##  SOFTWARE DEVELOPER      :  52571   ENGINEER  :101222   Median : 73560
##  SYSTEMS ANALYST         :  29185   MANAGER   : 37588   Mean   : 77964
##  COMPUTER PROGRAMMER     :  24964   TEACHER   : 37244   3rd Qu.: 90646
##  SENIOR SOFTWARE ENGINEER:  19364   ACCOUNTANT: 30395   Max.   :199981
##  (Other)                 :1061807   (Other)   :  6732
##  H1B_DEPENDENT       WORKSITE_CITY      WORKSITE_STATE    FIN_YEAR
##  N:745642      NEW YORK     :  52599   CA      :266659   2015:339427
##  Y:640783      HOUSTON      :  32654   TX      :151705   2016:355956
##                SAN FRANCISCO:  32534   NJ      :105568   2017:340370
##                ATLANTA      :  28299   NY      : 84667   2018:350672
##                SAN JOSE     :  27678   IL      : 73707
##                CHICAGO      :  23620   WA      : 56767
##                (Other)      :1189041   (Other):647352
##  FULL_TIME
##  N:  61808
##  Y:1324617
##
##
##
##
##
```

```
test_set<- df1[-sam,]
summary(test_set)
```

```
##  CASE_STATUS                           EMPLOYER_NAME
##  0: 65974    INFOSYS LIMITED                     : 26459
##  1:528209    TATA CONSULTANCY SERVICES LIMITED: 16877
##              WIPRO LIMITED                       :  9757
##              IBM INDIA PRIVATE LIMITED           :  9691
##              CAPGEMINI AMERICA INC               :  9186
##              ACCENTURE LLP                       :  8955
##              (Other)                             :513258
##                     JOB_TITLE            SOC_NAME      PREVAILING_WAGE
##  PROGRAMMER ANALYST      : 50289   SOFTWARE   :310103   Min.   : 25044
##  SOFTWARE ENGINEER       : 35051   DEVELOPERS:192518   1st Qu.: 61485
##  SOFTWARE DEVELOPER      : 22475   ENGINEER  : 43774   Median : 73611
##  SYSTEMS ANALYST         : 12560   MANAGER   : 15992   Mean   : 77960
##  COMPUTER PROGRAMMER     : 10797   TEACHER   : 15768   3rd Qu.: 90646
##  SENIOR SOFTWARE ENGINEER:  8038   ACCOUNTANT: 13048   Max.   :199930
##  (Other)                 :454973   (Other)   :  2980
##  H1B_DEPENDENT       WORKSITE_CITY     WORKSITE_STATE    FIN_YEAR      FU
LL_TIME
##  N:319091      NEW YORK     : 22577   CA      :114302   2015:145251   N:
26855
##  Y:275092      HOUSTON      : 14401   TX      : 65342   2016:152528   Y:
```


Name: Sumit Gupta
CWID: 10441745

```
567328
##                    SAN FRANCISCO: 13909   NJ     : 45274   2017:145885
##                    ATLANTA      : 12046   NY     : 36286   2018:150519
##                    SAN JOSE     : 11981   IL     : 31473
##                    CHICAGO      :  9985   WA     : 24148
##                    (Other)      :509284   (Other):277358

rm(df, efile_data)
```


Name: Sumit Gupta
CWID: 10441745

Naïve Bayes methodology:

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a feature in a class is unrelated to the presence of any other feature.

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

# Implementation Naïve Bayes methodology to develop a classification model
for the Diagnosis.
model <- naiveBayes(CASE_STATUS~., data =training_set)
pred <-predict(model,test_set)
table(pred,test_set$CASE_STATUS)

##
## pred      0      1
##    0  24318 119346
##    1  41656 408863

# Check accuracy
accuracy<- test_set$CASE_STATUS==pred
value<-100*(sum(accuracy)/length(accuracy))
cat("Accuracy for Model is ",value)

## Accuracy for Model is  72.90363

remove(list=c("model", "pred", "accuracy", "value"))
```

CART methodology:

The decision tree correctly identified that if a claim involved a rear-end collision, the claim was most likely fraudulent. By default, rpart uses gini impurity to select splits when performing classification.

```
# Install packages
#install.packages("rpart")
#install.packages("rpart.plot")

# Import Library
library(rpart)
library(rpart.plot)
model<-rpart( factor(CASE_STATUS)~.,data=training_set)
#rpart.plot(model)
pred<-predict(model,test_set, type="class")
#table(Actual=test_set,CART=pred)


#Check accuracy
accuracy<- test_set$CASE_STATUS==pred
value<-100*(sum(accuracy)/length(accuracy))
cat("Accuracy for Model is ",value)
```

```
## Accuracy for Model is  87.98501
```

```
remove(list=c("model", "pred", "value", "accuracy"))
```


Name: Sumit Gupta
CWID: 10441745

Random forests:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

```
#install.packages("randomForest")
library(randomForest)

# Implementation Random Forest methodology to develop a classification mod
el for the Diagnosis.
model<- randomForest(factor(CASE_STATUS)~.,data=training_set[,c(1,9)], imp
ortance=TRUE,ntree=10 )

pred<-predict(model,test_set)
#table(actual=test_set,pred)

# Check Error Rate for model
accuracy<- test_set$CASE_STATUS==pred
value<-100*(sum(accuracy)/length(accuracy))
cat("Accuracy for Model is ",value)

## Accuracy for Model is  88.89669
```


Name: Sumit Gupta
CWID: 10441745

Conclusion:

Model	Accuracy
Naïve Bayes	72.90363
CART	87.98501
Random forests	88.89669

Random Forest Classification algorithm best predicts the results with an accuracy of 88.90% for the data.

Name: Sumit Gupta
CWID: 10441745