CS-GY 6923 Machine Learning

Professor: Dr. Raman Kannan

HW1: Exploratory Data Analysis

Anil Poonai

Due 5th October 2021

Contents

1 Overview of the Data set

The data is called NYPD Complaint Data Current (Year to Date) and it includes all valid felony, misdemeanor, and violation crimes reported to the NYPD. It was taken from New York's open data portal: https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Current-Year-To-Date-/5uac-w243/data

The dataset is a record of all the complaints sent to NYPD and was created in 2016. It is updated every quarter of every year and is a part of NYC OpenData. It was last updated on the 4th August 2021. The data is straight from NYC Police Department and consists of all types of metadata from the location including latitude, longitude, borough, address, nearby subway stations, statues, etc. to the precinct that responded to the victim and suspect characteristics.

The dataset has 204,646 observations and 36 variables (mix of categorical and numerical). Of the 36 variables: 35 are independent and 1 is dependent. The dependent variable is categorical and has three values: felony, misdemeanor, and violation. This is the level of offense for the reported claim.

2 Loading the Data

The dataset is in csv format and is labelled 'NYPD_Complaint_Data_Current__Year_To_Date_.csv'. I used RStudio to first have a look at it and then transferred it to the remote computer provided by IBM for our class.

```
#Load the data
library(readr)
df <- read_csv("C:/Users/poona/Downloads/NYPD_Complaint_Data_Current__Year_To_Date_.csv")

#Check dimensions
dim(df)

#Change names to numbers to help reduce bias
names(df) = c(1:36)
names(df)
View(df)

#Label the dependent variable
names(df)[14] = 'Dependent'
names(df)
View(df)

head(df)

```

So I loaded the data into R and then checked the dimensions:

```
> #Check dimensions and names
> dim(df)
[1] 204646     36
```

I then renamed the dependent variable to label it:

```
> names(df)
 [1] "1"         "2"         "3"         "4"         "5"         "6"         "7"         "8"         "9"
[10] "10"        "11"        "12"        "13"        "Dependent" "15"        "16"        "17"        "18"
[19] "19"        "20"        "21"        "22"        "23"        "24"        "25"        "26"        "27"
[28] "28"        "29"        "30"        "31"        "32"        "33"        "34"        "35"        "36"
>
```

And then got a sample of what the dataset looks like to see if that came out correctly:

```
> head(df)
# A tibble: 6 x 36
        `1`   `2` `3`    `4`   `5`   `6`   `7`   `8`   `9`    `10`  `11` `12`   `13` Dependent `15`  `16`  `17`  `18`
      <dbl> <dbl> <chr> <chr> <tim> <chr> <tim> <chr> <chr> <dbl> <dbl> <chr> <dbl> <chr>     <chr> <chr> <chr> <chr> <chr>
1 758521794    67 NA    03/0~ 01:50 NA        NA COMP~ NA       NA    NA N.Y.~   101 FELONY    OUTS~ MURD~ NA    NA
2 696896951    60 NA    03/2~ 18:40 NA        NA COMP~ NA       NA    NA N.Y.~   101 FELONY    OUTS~ MURD~ NA    NA
3 972479923    75 BROO~ 03/2~ 15:55 03/2~ 16:00 COMP~ NA       NA     0 N.Y.~   113 FELONY    FRON~ FORG~ NA    PATR~
4 109344500    77 BROO~ 03/2~ 11:10 03/2~ 11:23 COMP~ NA       NA     0 N.Y.~   109 FELONY    FRON~ GRAN~ NA    PATR~
5 673945415    23 NA    03/2~ 23:13 NA        NA COMP~ NA       NA    NA N.Y.~   101 FELONY    OUTS~ MURD~ NA    NA
6 239448064    73 BROO~ 03/2~ 06:16 NA        NA ATTE~ NA       NA     0 N.Y.~   106 FELONY    INSI~ FELO~ NA    PATR~
# ... with 18 more variables: 19 <dbl>, 20 <chr>, 21 <chr>, 22 <chr>, 23 <chr>, 24 <chr>, 25 <chr>, 26 <chr>,
#   27 <dbl>, 28 <chr>, 29 <chr>, 30 <chr>, 31 <dbl>, 32 <dbl>, 33 <dbl>, 34 <dbl>, 35 <chr>, 36 <chr>
```

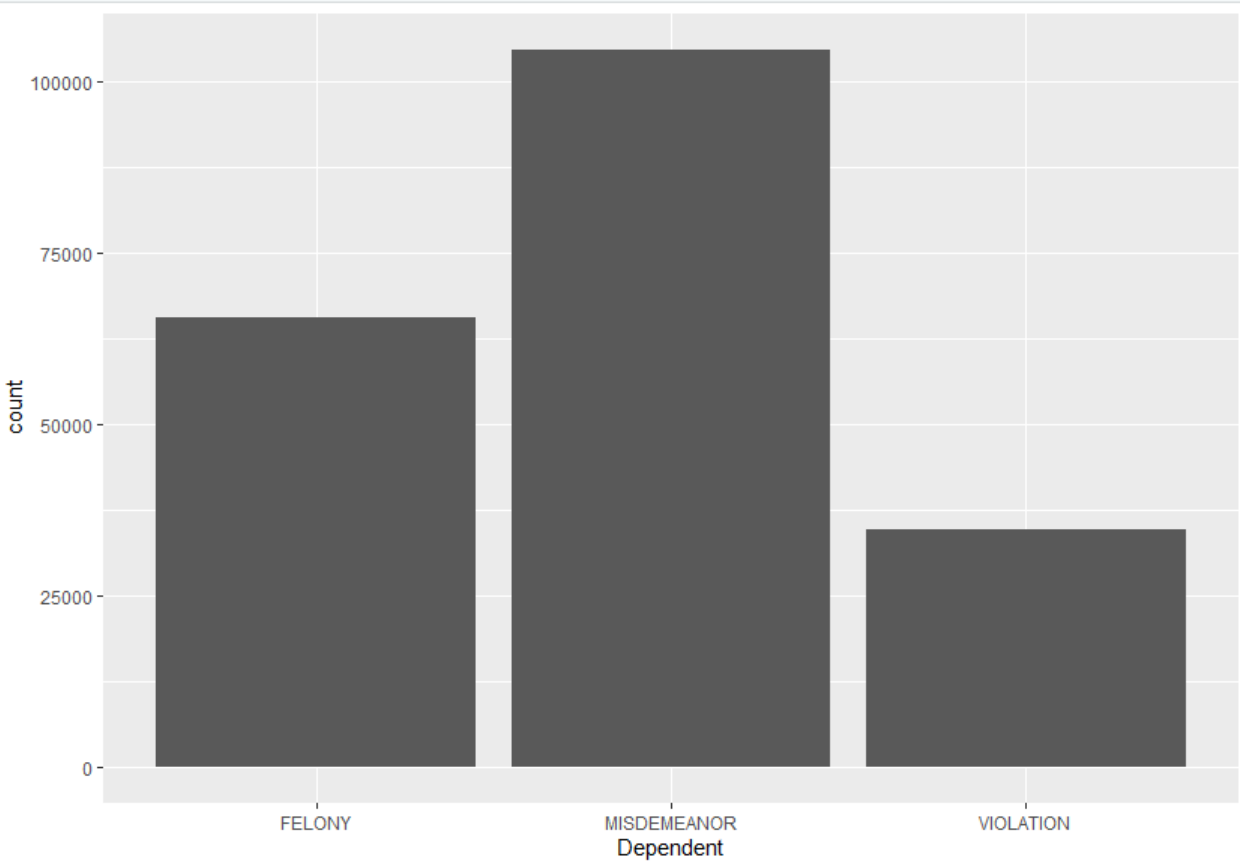
3 Imbalance Check

This has 204,646 observations and 36 variables, classifying it as a large multivariate dataset and will need a lot of computing power. The cardinality of the target is multi-class as it has three factors. Since the target value is categorical, we can use classification supervised models to help predict the outcome. Below is the code and results of the imbalance check in the dataset:

```
#Imbalance check
sum(df$Dependent=='FELONY')/nrow(df)
sum(df$Dependent=='MISDEMEANOR')/nrow(df)
sum(df$Dependent=='VIOLATION')/nrow(df)

ggplot(data = df) +
  geom_bar(mapping = aes(Dependent))
```

```
> sum(df$Dependent=='FELONY')/nrow(df)
[1] 0.3199232
> sum(df$Dependent=='MISDEMEANOR')/nrow(df)
[1] 0.5112389
> sum(df$Dependent=='VIOLATION')/nrow(df)
[1] 0.1688379
>
```



The minority classes have a mild to moderate difference or imbalance proportion. Which I'm determining as negligible due to the lowest minority having at least 25,000 observations.

Next thing I did was getting rid of the identifier column:

```
#Get rid of Identifier
df = df[-c(1)]
View(df)
```

4 Missing value treatment

Listed out all of the data types of each column next:

```
> str(df)
tibble [204,646 x 32] (S3: tbl_df/tbl/data.frame)
 $ 2         : num [1:204646] 67 60 75 77 23 73 113 105 46 109 ...
 $ 3         : chr [1:204646] NA NA "BROOKLYN" "BROOKLYN" ...
 $ 4         : chr [1:204646] "03/07/2021" "03/26/2021" "03/27/2021" "03/28/2021" ...
 $ 5         : 'hms' num [1:204646] 01:50:00 18:40:00 15:55:00 11:10:00 ...
  ..- attr(*, "units")= chr "secs"
 $ 6         : chr [1:204646] NA NA "03/27/2021" "03/28/2021" ...
 $ 7         : 'hms' num [1:204646] NA NA 16:00:00 11:23:00 ...
  ..- attr(*, "units")= chr "secs"
 $ 8         : chr [1:204646] "COMPLETED" "COMPLETED" "COMPLETED" "COMPLETED" ...
 $ 9         : chr [1:204646] NA NA NA NA ...
 $ 10        : num [1:204646] NA NA NA NA NA NA NA NA NA NA ...
 $ 11        : num [1:204646] NA NA 0 0 NA 0 0 NA 0 0 ...
 $ 13        : num [1:204646] 101 101 113 109 101 106 109 101 341 112 ...
 $ Dependent: chr [1:204646] "FELONY" "FELONY" "FELONY" "FELONY" ...
 $ 15        : chr [1:204646] "OUTSIDE" "OUTSIDE" "FRONT OF" "FRONT OF" ...
 $ 17        : chr [1:204646] NA NA NA NA ...
 $ 18        : chr [1:204646] NA NA "PATROL BORO BKLYN NORTH" "PATROL BORO BKLYN NORTH" ...
 $ 19        : num [1:204646] NA NA 729 457 NA 109 421 NA 357 739 ...
 $ 21        : chr [1:204646] NA NA "DEPARTMENT STORE" "STREET" ...
 $ 22        : chr [1:204646] "03/07/2021" "03/26/2021" "03/27/2021" "03/28/2021" ...
 $ 23        : chr [1:204646] NA NA NA NA ...
 $ 24        : chr [1:204646] NA NA "25-44" NA ...
 $ 25        : chr [1:204646] NA NA "WHITE HISPANIC" NA ...
 $ 26        : chr [1:204646] NA NA "M" NA ...
 $ 27        : num [1:204646] NA NA NA NA NA NA NA NA NA NA ...
 $ 28        : chr [1:204646] "18-24" "25-44" "25-44" "UNKNOWN" ...
 $ 29        : chr [1:204646] "BLACK" "BLACK HISPANIC" "BLACK" "UNKNOWN" ...
 $ 30        : chr [1:204646] "M" "M" "F" "D" ...
 $ 31        : num [1:204646] 1008789 987641 1020990 1004507 1000267 ...
 $ 32        : num [1:204646] 177637 148923 186549 182865 228200 ...
 $ 33        : num [1:204646] 40.7 40.6 40.7 40.7 40.8 ...
 $ 34        : num [1:204646] -73.9 -74 -73.9 -73.9 -73.9 ...
 $ 35        : chr [1:204646] "(40.654223423000076, -73.91156340899995)" "(40.57544276900006, -73.98779476299995)" "(40.6
7864264400004, -73.867542755)" "(40.66858395700007, -73.92697993199994)" ...
 $ 36        : chr [1:204646] "POINT (-73.91156340899995 40.654223423000076)" "POINT (-73.98779476299995 40.5754427690000
6)" "POINT (-73.867542755 40.67864264400004)" "POINT (-73.92697993199994 40.66858395700007)" ...
```

I then removed columns that were missing more than half of the total observations:

```
#Removing columns that have missing values summing atleast half of the total amount of observations
colSums(is.na(df))
nrow(df)/2
df = df[-c(5,6,8,9,16,22,26)]
```

and the key description columns of other columns as listed on NYC OpenData website: https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Current-Year-To-Date-/5uac-w243

```
#Removing columns that are a description of another column
df = df[-c(11,14)]
view(df)
```

5 Principal Component Analysis

Regarding dimensionality reduction, I used PCA and relied on all of the numerical values.

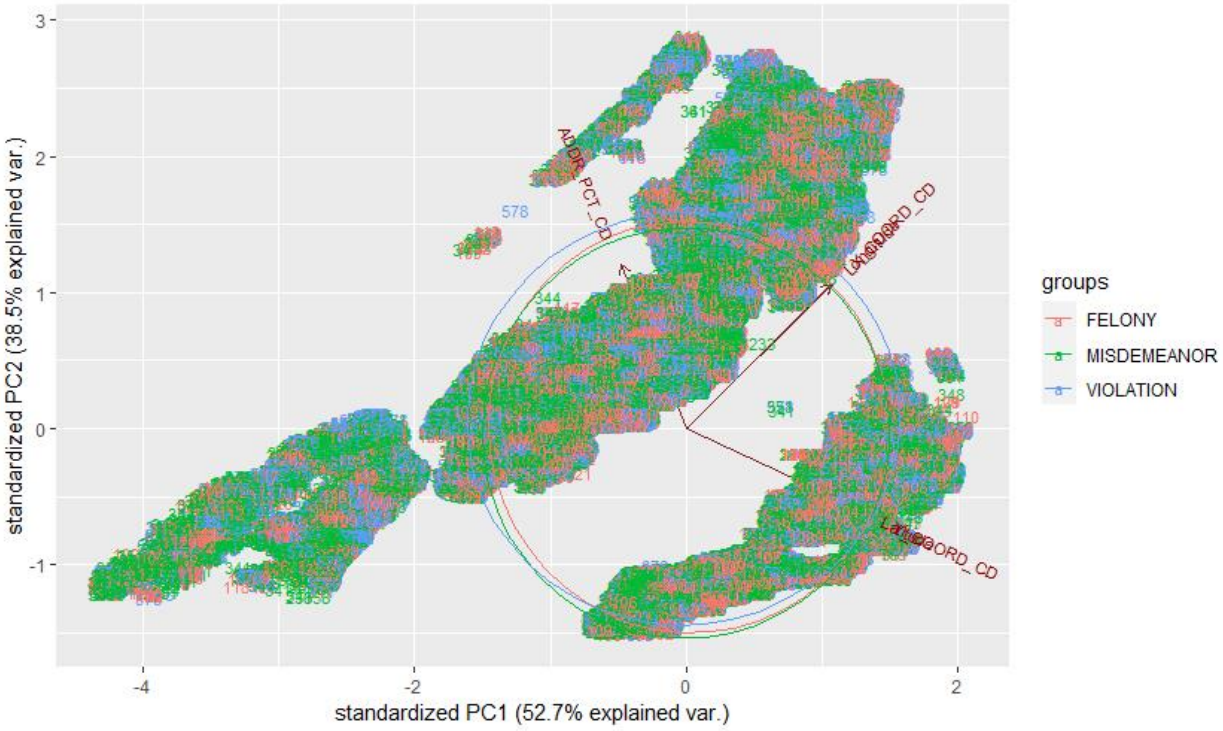
```
#Principle Component Analysis
pca<- prcomp(df[,c(1,21,22,23,24)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$KY_CD, ellipse=TRUE, groups=df$Dependent)
#91.2

pca<- prcomp(df[,c(1,21,22)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$KY_CD, ellipse=TRUE, groups=df$Dependent)
#89.5

pca<- prcomp(df[,c(1,23,24)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$KY_CD, ellipse=TRUE, groups=df$Dependent)
#89.5

pca<- prcomp(df[,c(1,8)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$KY_CD, ellipse=TRUE, groups=df$Dependent)
#Removing column 8 as that seems to be a 100% correlated to the dependent variable
```

I did multiple PCA tests and noted down the percentage or variance that's explainable and discovered that the column 8 at this point in the code had a 1-1 ratio with the dependent variable and therefore has to be removed as that's determined at the same time as the verdict of what type of offense has occurred. I also removed columns 24 and 25 as they are redundant being previously mentioned in the 4 columns before them. The problem with this is that the graphs made don't work in IBM's environment but I gave the code at the end of this document and the line that's needed for the graphs is line 8 and it is a comment but can be changed into a runnable line at any time.



This map is the plot of the PCA, as you can see the NYS XY plane coordinate system combined with the latitude and longitude global system has a 1.7% better chance at explaining the variance. This plot is the last PCA variable created.

6 Random Forest and Variable Importance

To do this next part, I had to make each categorical variable a factor and get rid of the ones that had more then 50 levels, which were all of the dates.

```
#randomForest and Variable Importance
library(randomForest)
df$Dependent = as.factor(df$Dependent)
table(df$Dependent)
colnames(df) = c('A','B','C','D','E','F','G','Dependent','H','I','J','K','L','M','N','O','P','Q','R','S','T','U'

df$B = as.factor(df$B)
df$C = as.factor(df$C)
df$E = as.factor(df$E)
df$G = as.factor(df$G)
df$H = as.factor(df$H)
df$I = as.factor(df$I)
df$K = as.factor(df$K)
df$L = as.factor(df$L)
df$M = as.factor(df$M)
df$N = as.factor(df$N)
df$O = as.factor(df$O)
df$P = as.factor(df$P)
df$Q = as.factor(df$Q)
df$R = as.factor(df$R)

df$C = NULL
df$D = NULL
df$K = NULL
df$L = NULL

set.seed(222)
index = sample(2, nrow(df), replace = TRUE, prob = c(0.7,0.3))
train = df[index==1,]
test = df[index==2,]
str(train)
summary(train)

rf = randomForest(Dependent~., train, na.action=na.omit)
rf
```

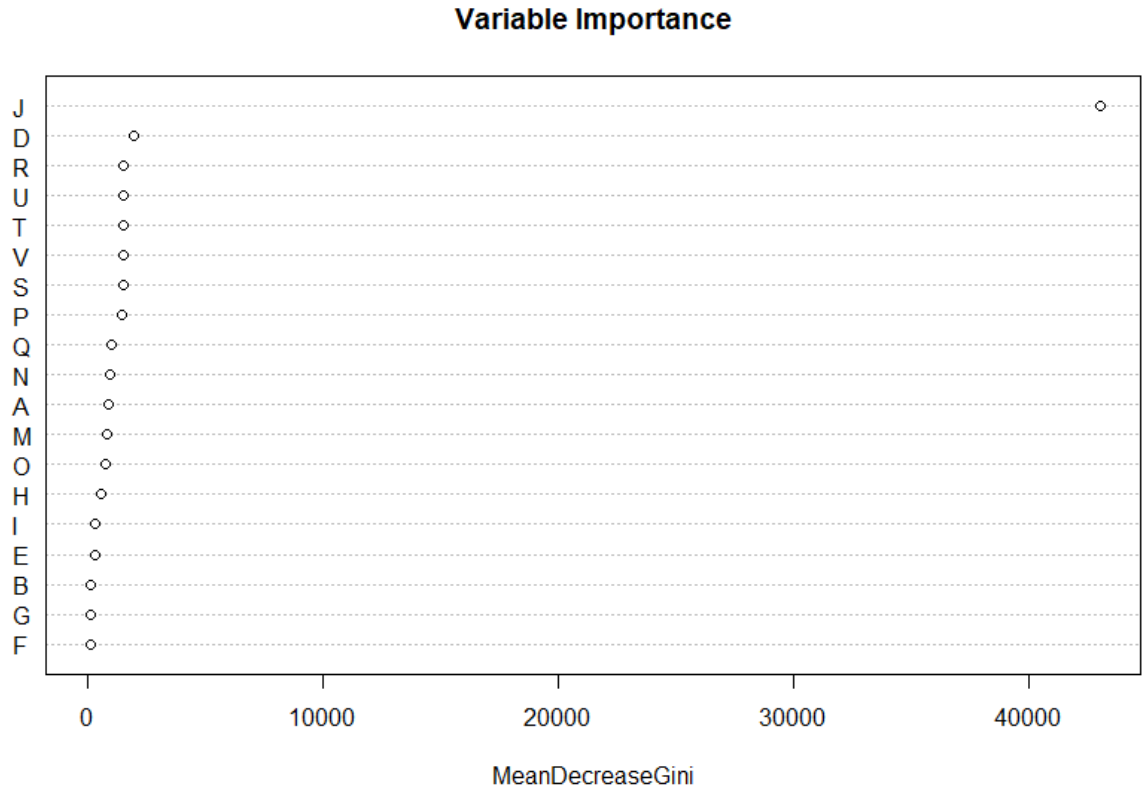
So, I decided to do a Random Forest algorithm on the data by first making the dependent variable a factor, so that categorical data is chosen, and changing all the column names to letters besides the dependent variable as the algorithm doesn't work well when the columns have numbers as names. I then made a train and test data and used the training data to make the forest.

```
> rf = randomForest(Dependent~., train, na.action=na.omit)
> rf

Call:
 randomForest(formula = Dependent ~ ., data = train, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 2.44%
Confusion matrix:
            FELONY MISDEMEANOR VIOLATION class.error
FELONY       43837        1831         1 0.040114739
MISDEMEANOR   1488       71651         2 0.020371611
VIOLATION       72          89     23947 0.006678281
>
```

After this, I used the variable importance function to check how much the model uses each variable.



7 VIF

Next thing I did was take the VIF, but since this is a classification data, I don't believe it'sa s useful as other methods.

```
116  str(df)
117  #VIF
118  library(car)
119  m = lm(A~F+J+S+T+U+V, data=df)
120  vif(m)
```

```
> vif(m)
           F            J            S            T            U            V
1.000507e+00 1.003262e+00 1.050093e+06 5.351406e+06 5.345899e+06 1.052566e+06
>
```

The variables above 5 are the latitude and longitude, and since the variable A is referring to the precinct the incident occurred in. It makes sense that these are correlated but not enough for me to remove them as they aren't redundant.

8 Code

#Load the data

library(readr)

```
library(plyr)

library(dplyr)

library(tidyverse)

library(usethis)

library(devtools)

#install_github("vqv/ggbiplot", force=TRUE)

library(grid)

library(ggbiplot)

library(ggplot2)

library(lattice)

library(caret)

#From IBM Terminal

df <-
read_csv("/home/2021/nyu/fall/ap5254/hw01/NYPD_Complaint_Data_Current__Year_To_Date_.csv")

#If you have the file

#df <- read_csv("C:/Users/poona/Downloads/NYPD_Complaint_Data_Current__Year_To_Date_.csv")

#If you have the link

#df = read.csv(url("https://data.cityofnewyork.us/api/views/5uac-
w243/rows.csv?accessType=DOWNLOAD"))

#Check dimensionsdim(df)


#Change names to numbers to help reduce bias

names(df) = c(1:36)

names(df)


#Label the dependent variable

names(df)[14] = 'Dependent'

names(df)
```

```
head(df)

#Imbalance check
sum(df$Dependent=='FELONY')/nrow(df)
sum(df$Dependent=='MISDEMEANOR')/nrow(df)
sum(df$Dependent=='VIOLATION')/nrow(df)

ggplot(data = df) +
  geom_bar(mapping = aes(Dependent))

#Get rid of Identifier
df = df[-c(1)]

#Data Types of each column
str(df)

#Removing columns that have missing values summing at least half of the total amount of observations
colSums(is.na(df))
nrow(df)/2
df = df[-c(5,6,8,9,16,22,26)]

#Removing columns that are a description of another column
df = df[-c(11,14)]
str(df)

#Principle Component Analysis
pca<- prcomp(df[,c(1,21,22,23,24)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$`13`, ellipse=TRUE, groups=df$Dependent)
```

```
#91.2

pca<- prcomp(df[,c(1,21,22)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$`13`, ellipse=TRUE, groups=df$Dependent)
#89.5

pca<- prcomp(df[,c(1,23,24)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$`13`, ellipse=TRUE, groups=df$Dependent)
#89.5

pca<- prcomp(df[,c(1,8)], center = TRUE,scale. = TRUE)
str(pca)
ggbiplot(pca,labels = df$`13`, ellipse=TRUE, groups=df$Dependent)
#Removing column 8 as that seems to be a 100% correlated to the dependent variable
df = df[-c(8)]
df = df[-c(24,25)]

#randomForest and Variable Importance
library(randomForest)
df$Dependent = as.factor(df$Dependent)
table(df$Dependent)
colnames(df) = c('A','B','C','D','E','F','G','Dependent','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V')

df$B = as.factor(df$B)
df$C = as.factor(df$C)
df$E = as.factor(df$E)
df$G = as.factor(df$G)
```

```
df$H = as.factor(df$H)
df$I = as.factor(df$I)
df$K = as.factor(df$K)
df$L = as.factor(df$L)
df$M = as.factor(df$M)
df$N = as.factor(df$N)
df$O = as.factor(df$O)
df$P = as.factor(df$P)
df$Q = as.factor(df$Q)
df$R = as.factor(df$R)

df$C = NULL
df$D = NULL
df$K = NULL
df$L = NULL

set.seed(222)
index = sample(2, nrow(df), replace = TRUE, prob = c(0.7,0.3))
train = df[index==1,]
test = df[index==2,]
str(train)
summary(train)

rf = randomForest(Dependent~., train, na.action=na.omit)
rf

VI =varImp(rf)
varImpPlot(rf,main="Variable Importance")
```

```
str(df)
#VIF
library(car)
m = lm(A~F+J+S+T+U+V, data=df)
vif(m)
```