FINAL GROUP PROJECT REPORT

Partial fulfilment for the award of the

Post Graduate Certification

in Data Analytics for Engineers

September 2021

SUBMITTED BY:

Anza

Gurunad

Karthik

Rashbin

Sachin

Under the Guidance of

Miss.Shalini Kumari

Technical Trainer

Edubridge

Post Graduate Certification in Data Analytics for Engineers

Edubridge India

September 2021

AKNOWLEDGMENT

Acknowledgement is not a mere obligation but the epitome of humility and ineptness to all those who have helped in the completion of this project. We are thankful to Miss Shalini Kumari, Edubridge India ASET for their constant guidance and encouragement provided in this endeavour. We also thank all our friends who helped us out in completing this project and helping us to solve various problems encountered during the progress of this project.

ABSTRACT

In this report, we completed the analysis two dataset using 5 tools .One is Amazon Prime Tv Shows and other is Pima Indians Diabetes. We got datasets from Kaggle . Tools used

- Python
- R
- Excel
- Tableau
- SAS

EduBridge

CONTENTS

1.INTRODUCTION

This data set was created so as to analyze the latest shows available on Amazon Prime as well as the shows with a high rating.

Content

The data set contains the name of the show or title, year of the release which is the year in which the show was released or went on-air, No.of seasons means the number of seasons of the show which are available on Prime, Language is for the audio language of the show and does not take into consideration the language of the subtitles, genre of the show like Kids, Drama, Action and so on, IMDB ratings of the show: though for many tv shows and kid shows the rating was not available, Age of Viewers is to specify the age of the target audience- All in age means that the content is not restricted to any particular age group and all audiences can view it.

2. ANALYSIS

2. 1. PYTHON

Jupyter notebook is used for analysis of data set. Firstly, import all libraries. Then read the data set and done EDA analysis. Then done Machine learning. We used logistic regression for ML approach.

Import Libraries

In [75]:

```
import pandas as pd
import numpy as np
from os import path
from PIL import Image
from wordcloud import WordCloud,STOPWORDS,ImageColorGenerator
#from wordcloud import WordCloud
#from wordcloud import STOPWORDS
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import warnings
warnings.filterwarnings('ignore')
```

Load and read the data

In [76]:

```
data=pd.read_csv("Prime TV Shows.csv")
data
```

Out[76]:

	S.no.	Name of the show	Year of release	No of seasons available	Language	Genre	IMDb rating	Age of viewers
0	1	Pataal Lok	2020.0	1.0	Hindi	Drama	7.5	18+
1	2	Upload	2020.0	1.0	English	Sci-fi comedy	8.1	16+
2	3	The Marvelous Mrs. Maisel	2017.0	3.0	English	Drama, Comedy	8.7	16+
3	4	Four More Shots Please	2019.0	2.0	Hindi	Drama, Comedy	5.3	18+
4	5	Fleabag	2016.0	2.0	English	Comedy	8.7	18+

Dropping Serial Number column

In [3]:
```
data.drop('S.no.',axis=1,inplace=True)
```

To check the rows and columns

In [4]:
```
data.shape
```

Out[4]:
```
(404, 7)
```

To print top 5 records

In [5]:
```
data.head()
```

Out[5]:

	Name of the show	Year of release	No of seasons available	Language	Genre	IMDb rating	Age of viewers
0	Pataal Lok	2020.0	1.0	Hindi	Drama	7.5	18+
1	Upload	2020.0	1.0	English	Sci-fi comedy	8.1	16+
2	The Marvelous Mrs. Maisel	2017.0	3.0	English	Drama, Comedy	8.7	16+
3	Four More Shots Please	2019.0	2.0	Hindi	Drama, Comedy	5.3	18+
4	Fleabag	2016.0	2.0	English	Comedy	8.7	18+

To print bottom 5 records

In [6]:
```
data.tail()
```

To print columns names

In [7]:
```
data.columns
```

Out[7]:
```
Index(['Name of the show', 'Year of release', 'No of seasons available',
       'Language', 'Genre', 'IMDb rating', 'Age of viewers'],
      dtype='object')
```

To check the datatype

In [8]:
```
data.dtypes
```

Out[8]:
```
Name of the show            object
Year of release            float64
No of seasons available    float64
Language                    object
Genre                       object
IMDb rating                float64
Age of viewers              object
dtype: object
```

Statistical Details

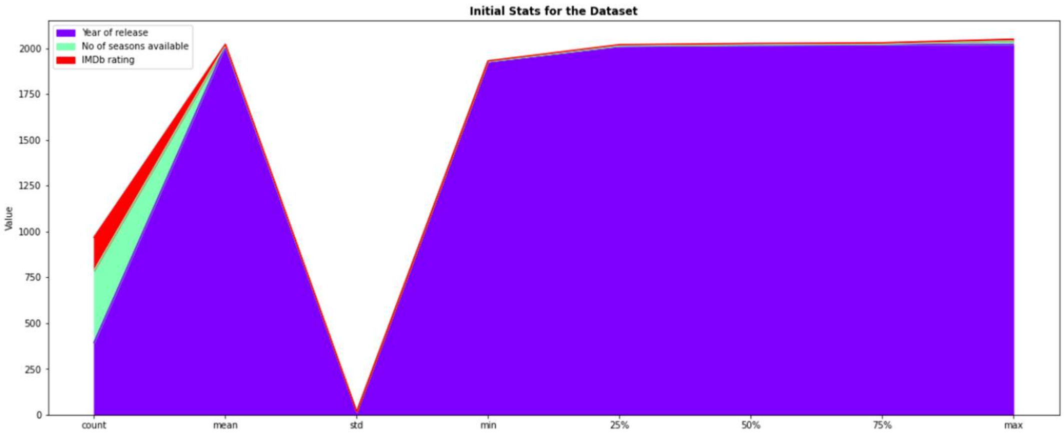
In [9]:
```
data.describe()
```

Out[9]:

	Year of release	No of seasons available	IMDb rating
count	393.000000	393.000000	182.000000
mean	2011.279898	2.608142	7.354396
std	12.944861	2.592008	0.959372

Plotting the statistical details

In [10]:
```
#Just trying to plot the above seen initial stats
data.describe().plot(kind='area',fontsize=10,figsize=(20,8),colormap='rainbow')
plt.title("Initial Stats for the Dataset",fontweight='bold')
plt.ylabel("Value")
plt.show()
```



What we can find from here:

- Min value for year of release - 1926? Seriously Dude? TV series from 1926? We will find out soon

Sum of null values

In [77]:
```
data.isna().sum()
```

Out[77]:
```
S.no.                       0
Name of the show           11
Year of release            11
No of seasons available    11
Language                   11
Genre                      11
IMDb rating               222
Age of viewers             11
dtype: int64
```

Fill the null values

In [78]:
```
data['Name of the show'].fillna(data['Name of the show'].value_counts().index[0],inplace=True)
data['Year of release'].fillna((data['Year of release'].mean()),inplace=True)
data['No of seasons available'].fillna((data['No of seasons available'].mean()),inplace=True)
data['Language'].fillna(data['Language'].value_counts().index[0],inplace=True)
data['Genre'].fillna(data['Genre'].value_counts().index[0],inplace=True)
data['IMDb rating'].fillna((data['IMDb rating'].mean()),inplace=True)
data['Age of viewers'].fillna(data['Age of viewers'].value_counts().index[0],inplace=True)

```

Check the null values

In [79]:
```
data.isna().sum()
```

Out[79]:
```
S.no.                     0
Name of the show          0
```

Print name of the shows and their counts

In [15]:
```
data['Name of the show'].unique()
```

Out[15]:
```
array(['Pataal Lok', 'Upload', 'The Marvelous Mrs. Maisel',
       'Four More Shots Please', 'Fleabag', 'Made in Heaven',
       'Homecoming', 'Mirzapur', 'The Family Man', 'Modern Love',
       'Comicstaan', 'Inside Edge', 'The Boys', 'Hanna', 'Hunters',
       'Good Omens', 'Breathe', 'The Forgotten Army- Azaadi ke Liye',
       "Tom Clancy's Jack Ryan", 'Tales from the Loop',
       "The Test: A New Era for Australia's Team",
       'The Man in the High Castle', 'One Mic Stand', 'Undone',
       'American Gods', 'The Tick', 'Jestination Unknown',
       'Man with a Plan', 'Suits', 'Doctor Who', "Grey's Anatomy",
       'The Mentalist', 'Afsos', 'Laakhon Mein Ek', 'House',
       'The Good Doctor', 'The Vampire Diaries', 'Hostel Daze',
       'Mr. Robot', 'Supernatural', 'Dexter', 'The Magicians',
       'Blindspot', 'The Good Wife', 'The Girlfriend Experience',
       'Chacha Vidhayak Hain Humare', 'Scorpion', 'Shameless', 'Reign',
       'Downtown Abbey', 'McMafia', 'This is Us',
       'Malgudi Days Swami & Friends 1', 'Nikita', 'Carnival Row',
       'Legacies', 'Bates Motel', 'The Exorcist', 'The Night Manager',
       'The Originals', 'The Purge', 'Deception', 'The Last Ship',
       'Treadstone', 'The Terror', 'Manifest', 'Pushpavalli', 'Fringe',
```

Wordcloud showing the most frequent "Name of the show" released by Amazon prime'.

In [16]:
```
stopwords = set(STOPWORDS)
wordcloudG=WordCloud(max_font_size=40, relative_scaling=.5,colormap="Dark2",stopwords=stopwords).generate(data['Name of the
plt.imshow(wordcloudG, interpolation="bilinear")
plt.axis('off')
plt.margins(x=0, y=0)
plt.show()
```

Wordcloud showing the most frequent "Name of the show" released by Amazon prime'.

In [16]:
```
stopwords = set(STOPWORDS)
wordcloudG=WordCloud(max_font_size=40, relative_scaling=.5,colormap="Dark2",stopwords=stopwords).generate(data['Name of the
plt.imshow(wordcloudG, interpolation="bilinear")
plt.axis('off')
plt.margins(x=0, y=0)
plt.show()
plt.savefig("donaldwc.png")

```


```
<Figure size 432x288 with 0 Axes>
```

In [17]:
```
data.value_counts("Name of the show")
```

Out[17]:
```
Name of the show
The Last Ship            13
The Missing               2
#IMomSoHard Live          1
Silver and Gold           1
Splitting Up Together     1
```

Print Years of release of the show and their counts

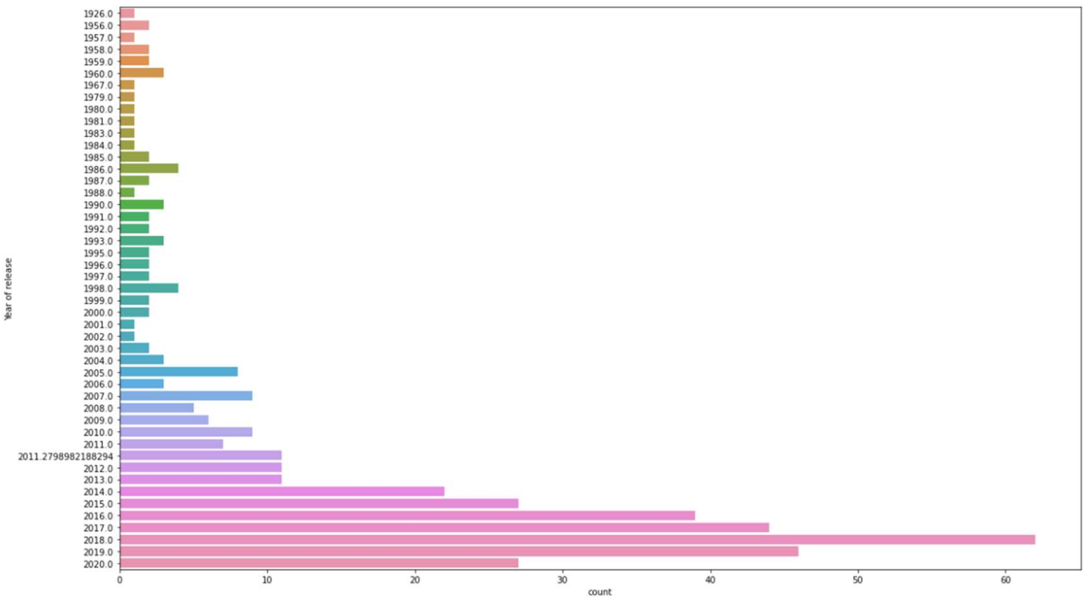
In [18]:
```
data["Year of release"].unique()
```

Out[18]:
```
array([2020.        , 2017.        , 2019.        , 2016.        ,
       2018.        , 2015.        , 2011.        , 2005.        ,
       2009.        , 2010.        , 2006.        , 2014.        ,
       1987.        , 2013.        , 2012.        , 2004.        ,
       2007.        , 2008.        , 1998.        , 1985.        ,
       1992.        , 2002.        , 1956.        , 1996.        ,
       1986.        , 1983.        , 1926.        , 1959.        ,
       1990.        , 1960.        , 1997.        , 1993.        ,
       2000.        , 1991.        , 2001.        , 1979.        ,
       1957.        , 1981.        , 1995.        , 2003.        ,
       1984.        , 1999.        , 1980.        , 1967.        ,
       1988.        , 1958.        , 2011.27989822])
```

In [19]:
```
data.value_counts("Year of release")
```

Out[19]:
```
Year of release
2018.000000    62
2019.000000    46
2017.000000    44
2016.000000    39
2020.000000    27
2015.000000    27
2014.000000    22
2013.000000    11
2012.000000    11
2011.279898    11
2010.000000     9
2007.000000     9
2005.000000     8
2011.000000     7
2009.000000     6
2008.000000     5
```

In [20]:
```
plt.figure(figsize = (20,12))
sns.countplot(y= 'Year of release', data = data);
```



The ratings have not changed much over the years while the number of shows offered on Amzon Prime have increased at a fast rate. The number of shows before the year 2010 are below 10. The data for 2020 cannot be considered for the analysis as it does not take all the months into account .

Print the number of seasons available for the shows and their counts

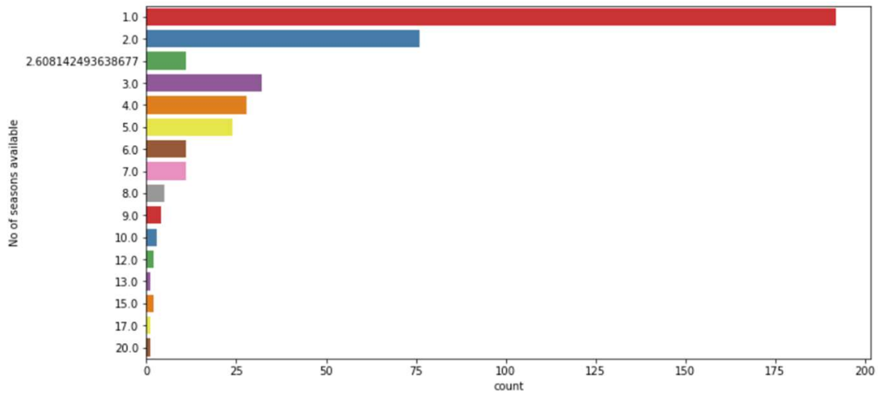
In [21]:
```
data["No of seasons available"].unique()
```

Out[21]:
```
array([ 1.        ,  3.        ,  2.        ,  4.        ,  8.        ,
       13.        , 15.        ,  7.        ,  5.        , 10.        ,
        6.        , 17.        ,  9.        , 20.        , 12.        ,
        2.60814249])
```

In [22]:
```
data.value_counts("No of seasons available")
```

Out[22]:
```
No of seasons available
1.000000     192
2.000000      76
3.000000      32
4.000000      28
5.000000      24
2.608142      11
6.000000      11
7.000000      11
8.000000       5
9.000000       4
10.000000      3
12.000000      2
15.000000      2
13.000000      1
17.000000      1
20.000000      1
dtype: int64
```

In [23]:
```
plt.figure(figsize = (12,6))
sns.countplot(y = 'No of seasons available', data = data,palette='Set1');
```



Shows with less number of seasons, mostly 1 or 2 seasons are the highest. Shows with more than 12 seasons, are very less in number.

Different languages in which movies are released on Amazon prime.

In [24]:
```
data["Language"].unique()
```

LOGISTIC REGRESSION:

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

Model Building

In [65]:
```
columns=['Name of the show',"Language"]
dr=data.drop(columns,axis=1)
dr
```

Out[65]:

	Year of release	No of seasons available	Genre	IMDb rating	Age of viewers
0	2020.000000	1.000000	Drama	7.500000	18+
1	2020.000000	1.000000	Sci-fi comedy	8.100000	16+
2	2017.000000	3.000000	Drama, Comedy	8.700000	16+
3	2019.000000	2.000000	Drama, Comedy	5.300000	18+
4	2016.000000	2.000000	Comedy	8.700000	18+
...	...	...	...	...	...
399	2011.279898	2.608142	Drama	7.354396	16+
400	2011.279898	2.608142	Drama	7.354396	16+
401	2011.279898	2.608142	Drama	7.354396	16+
402	2011.279898	2.608142	Drama	7.354396	16+
403	2011.279898	2.608142	Drama	7.354396	16+

404 rows × 5 columns

In [66]:
```
x=pd.get_dummies(dr)
x
```

Out[66]:

	Year of release	No of seasons available	IMDb rating	Genre_Action	Genre_Action, Comedy	Genre_Adventure	Genre_Animation	Genre_Animation, Drama	Genre_Arts, Entertainment, Culture	Genre_Comedy	...
0	2020.000000	1.000000	7.500000	0	0	0	0	0	0	0	...
1	2020.000000	1.000000	8.100000	0	0	0	0	0	0	0	...
2	2017.000000	3.000000	8.700000	0	0	0	0	0	0	0	...
3	2019.000000	2.000000	5.300000	0	0	0	0	0	0	0	...
4	2016.000000	2.000000	8.700000	0	0	0	0	0	0	1	...
...	...	...	...	...	...	...	...	...	...	...	...
399	2011.279898	2.608142	7.354396	0	0	0	0	0	0	0	...
400	2011.279898	2.608142	7.354396	0	0	0	0	0	0	0	...
401	2011.279898	2.608142	7.354396	0	0	0	0	0	0	0	...
402	2011.279898	2.608142	7.354396	0	0	0	0	0	0	0	...
403	2011.279898	2.608142	7.354396	0	0	0	0	0	0	0	...

404 rows × 57 columns

In [67]:
```
y=data['Language']
y
```

Out[67]:
```
0        Hindi
1      English
2      English
3        Hindi
4      English
        ...
399    English
400    English
401    English
402    English
403    English
Name: Language, Length: 404, dtype: object
```

Split the dataset in training set and test data

In [68]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

Train the model on training set

In [69]:
```
l=LogisticRegression()
l.fit(x_train,y_train)
print("train test complete")
```

```
train test complete
```

Predict the test result

In [70]:
```
y_pred=l.predict(x_test)
y_pred
```

Out[70]:
```
array(['English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English', 'English', 'English', 'English',
       'English', 'English', 'English'], dtype=object)
```

In [71]:
```
d1=pd.DataFrame({"Actual":y_test,"Predicted":y_pred})
d1
```

Out[71]:

	Actual	Predicted
360	Marathi	English
62	English	English
374	Japanese	English
92	English	English
146	English	English
...	...	...
11	Hindi	English
363	English	English
300	English	English
41	English	English
161	English	English

81 rows × 2 columns

Evaluate the model

In [72]:
```
accuracy_score(y_test,y_pred)
```

Out[72]: 0.7901234567901234

Accuracy of logistic regression is 0.79

Conclusion

```
By using the logistic regression model we were able to predict the languages in which the shows were released with an accuracy
of 79%.So with this model we can predict the language in which the shows would be released in future as well for a set of x
parameters.
```

2.2. R

Exploratory Data Analysis (EDA) in R is the process of analyzing and visualizing the data to get a better understanding of the data and glean insight from it. There are various steps involved when doing EDA but the following are the common steps that a data analyst can take when performing EDA:

1. Import the data
2. Clean the data

3. Process the data
4. Visualize the data

We have done our R analysis in Pima Indians Diabetes Dataset.

#Pima Indians Diabetes

Who is Pima Indians ?

#"The Pima Indians ("River People")are a group of Native Americans living in an area

#consisting of what is now central and southern Arizona.

Understanding the data

#The datasets consist of several medical predictor (independent) variables

#and one target (dependent) variable, Outcome. Independent variables include

#the number of pregnancies the patient has had, their BMI, insulin level,

#age, and so on.

Load required libraries

library(ggplot2)

library(ggthemes)

library(psych)

library(dplyr)

```
library(caret)

library(reshape2)

library(corrplot)




# Load the dataset

dataset=read.csv("C:/Users/Sachin/Programs_EduBridge/diabetes.csv"
)



# Print first 5 row

print(head(dataset,5))



# Print last 5 row

print(tail(dataset,5))



# To View the contents in the dataet

View(dataset)



# Print column names
```

```
print(names(dataset))


# Dimention of data
print(dim(dataset))


# Print Statistical summary
describe(dataset)


# Internal structure of data
print(str(dataset))


# Display columns  and display some portions of the data
#print(glimpse(dataset))


# Statistical values
print(is.na(dataset))
print(ncol(dataset))
print(nrow(dataset))
print(max(dataset$Outcome))
```

```
print(min(dataset$Outcome))

print(sort(dataset$Outcome))

print(which.max(dataset$Outcome))# Return the index of the first
maximum value

print(which.min(dataset$Outcome))# Return the index of the first
minimum value

print(mean(dataset$Outcome))

print(mean(dataset$Outcome,trim=0.10))

print(var(dataset$Outcome))

print(median(dataset$Outcome))

print(mad(dataset$Outcome))# mean absolute division

print(sd(dataset$Outcome))

print(range(dataset$Outcome))

print(quantile(dataset$Outcome))

print(IQR(dataset$Outcome))

print(t.test(dataset$Outcome))



# Data visualisation



# Create a 2 x 2 plotting matrix

par(mfrow = c(2, 2))
```

The $ notation can be used to subset the variable you're interested in.

```
# Histogram of numerical data

print(hist(dataset$Pregnancies,col="red"))

print(hist(dataset$Age,col="blue3"))

print(hist(dataset$Glucose,col="cyan"))

print(hist(dataset$BMI,col="pink"))

#Age and number of times pregnant are not normal distributions as expected since the underlying population should not be

#normally distributed either. This 392 observations are just a sample of the original population. On the other hand, the glucose

#level and BMI seem to follow a normal distribution. When performing any analysis, it is always good to know what is the

#distribution of the data so all the assumptions for different tests or models can be met."""



# Age distribution


age<-
ggplot(dataset,aes(x=Age))+geom_histogram(binwidth=10,col="blue",fill="brown")+
  labs(title="Age column",x="Age","Count")

print(age)



#Pregnancy distribution
```

```
print(str(dataset$Pregnancies))

print(table(dataset$Pregnancies))# Create a table for pregnancies



dataset$Outcome <- as.factor(dataset$Outcome)

#All 8 independent variables are numeric. There are two outcomes, this
data is good for classifciation.

#Lets change Outcome to categorical Variable



pd<-ggplot(dataset,aes(x = Pregnancies)) +

  geom_histogram(binwidth = 0.5,aes(fill = Outcome,position =
"dodge")) +

  ggtitle("Pregnancies Data Distribution") + ylab("OutCome Counts") +

  theme_light() +

  theme_update(plot.title = element_text(hjust = ))

print(pd)


#Pregnancies data is right skewed.


opm<-ggplot(data = dataset,aes(x = Outcome, y = Pregnancies)) +

  geom_boxplot( aes(fill= Outcome)) +

  scale_y_continuous(breaks = seq(1,12,1),limits = c(0,12)) +

  ggtitle("Pregnancies boxplot") +

  stat_summary(fun.y=mean, colour="darkred", geom="point",
```

```
            shape=18, size=3,show.legend = TRUE) +
  theme_gray() +
  theme_update(plot.title = element_text(hjust = 0.5))
print(opm)
```

#Box plot shows, woman who had more pregnancies are more prone to diabetes.This may be important variable for model.

```
ogm<-ggplot(data = dataset,aes(x = Outcome, y = Glucose)) +
  geom_boxplot( aes(fill= Outcome)) +
  scale_y_continuous(breaks = seq(80,200,10),limits = c(80,200)) +
  ggtitle("Glucose") +
  stat_summary(fun=mean, colour="darkred", geom="point",
            shape=18, size=3,show.legend = TRUE) +
  theme_gray() +
  theme_update(plot.title = element_text(hjust = 0.5))
print(ogm)
```

#Diabetics woman have high Plasma glucose concentration.

#On average this value is 140 for diabetics woman while this is quite low for non-diabetics.

```
#Blood Pressure
table(dataset$BloodPressure)

obm<-ggplot(data = dataset,aes(x = Outcome, y = BloodPressure)) +
```

```
  geom_boxplot( aes(fill= Outcome)) +
  scale_y_continuous(breaks = seq(60,110,10),limits = c(60,110)) +
  ylab("Blood Pressure") +
  ggtitle("Blood Pressure Histogram") +
  stat_summary(fun=mean, colour="darkred", geom="point",
               shape=18, size=3,show.legend = TRUE) +
  theme_gray() +
  theme_update(plot.title = element_text(hjust = 0.5))
print(obm)
#Diastolic blood pressure for diabetic woman is higher compare to non-diabetics.


#Triceps skin fold thickness
#Triceps skin-fold thickness normal value for female 23
table(dataset$SkinThickness)
# Let's replace zero with the median value.
dataset$SkinThickness <- ifelse(
  dataset$SkinThickness == 0 ,
  median(dataset$SkinThickness,na.rm = TRUE),
  dataset$SkinThickness)



osm<-ggplot(data = dataset,aes(x = Outcome, y = SkinThickness)) +
  geom_boxplot( aes(fill= Outcome),outlier.colour = "red", outlier.size = 5) +
```

```
  scale_y_continuous(breaks = seq(0,100,10),limits = c(0,100)) +
  ylab("Triceps skin fold thickness") +
  ggtitle("Skin Thickness Histogram") +
  stat_summary(fun=mean, colour="darkred", geom="point",
               shape=18, size=3,show.legend = TRUE) +
  theme_gray() +
  theme_update(plot.title = element_text(hjust = 0.5))
print(osm)
#Boxplot shows that diabetics woman normally has high skin thickness.
#Red big dots are outlier but ignoring this outlier to consider the
extreme case.



table(dataset$BMI)


obm<-ggplot(data = dataset,aes(x = Outcome, y = BMI)) +
  geom_boxplot( aes(fill= Outcome),outlier.colour = "red", outlier.size =
5) +
  scale_y_continuous(breaks = seq(20,70,5),limits = c(20,70)) +
  ylab("BMI") +
  ggtitle("Body mass index Histogram") +
  stat_summary(fun=mean, colour="darkred", geom="point",
               shape=18, size=3,show.legend = TRUE) +
  theme_gray() +
  theme_update(plot.title = element_text(hjust = 0.5))
```

print(obm)

#BMI for diabetics woman is high compare to non-diabetics.

#There are few outlier, let not treat them to consider the extreme cases of BMI.

#Diabetes pedigree function

```
odpf<-ggplot(data = dataset,aes(x = Outcome, y = DiabetesPedigreeFunction)) +
  geom_boxplot( aes(fill= Outcome),outlier.colour = "red", outlier.size = 5) +
  scale_y_continuous(breaks = seq(0,2,0.2),limits = c(0,2)) +
  ylab("Diabetes Pedigree Function") +
  ggtitle("Diabetes Pedigree Function") +
  stat_summary(fun=mean, colour="darkred", geom="point",
               shape=18, size=3,show.legend = TRUE) +
  theme_gray() +
  theme_update(plot.title = element_text(hjust = 0.5))
print(odpf)
```

#Check the balancing of data

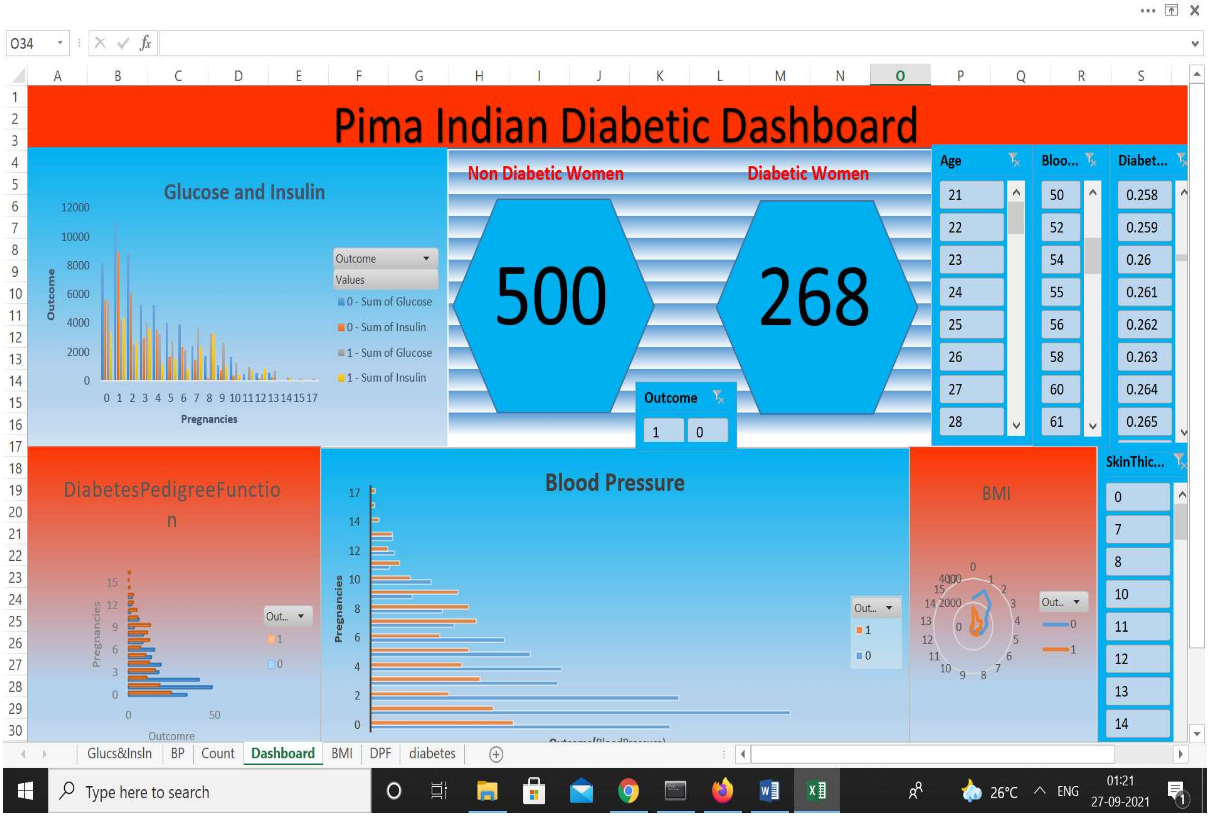
```
table(dataset$Outcome)
prop.table(table(dataset$Outcome))
ggplot(dataset,aes(Outcome))+
  geom_bar(fill=c("red","green"))+
```

```
 geom_text(stat = "count",aes(label=stat(count),vjust=0.5))


# it seems to be unbalanced



# correlation matrix

cor_melt <- melt(cor(dataset[, 1:8]))

cor_melt <- cor_melt[which(cor_melt$value > 0.5 & cor_melt$value !=
1), ]

cor_melt <- cor_melt[1:3, ]

print(cor_melt)

#correlation values higher than 0.5.



#Let's see the correlation between numerical variables. There are
variables which are highly correlated.

#That is the case of Age for example.

correlat <- cor(dataset[, setdiff(names(dataset), 'Outcome')])

print(correlat)

print(corrplot(correlat,method="ellipse"))


#In this analysis, we used the diabetic patient health management
follow-up data

#We have combined feature selection and imbalanced processing
techniques.
```

2.3 Excel

A dashboard is a visual representation of key metrics that allow you to quickly view and analyze your data in one place. Dashboards not only provide consolidated data views, but a self-service business intelligence opportunity, where users are able to filter the data to display just what's important to them.

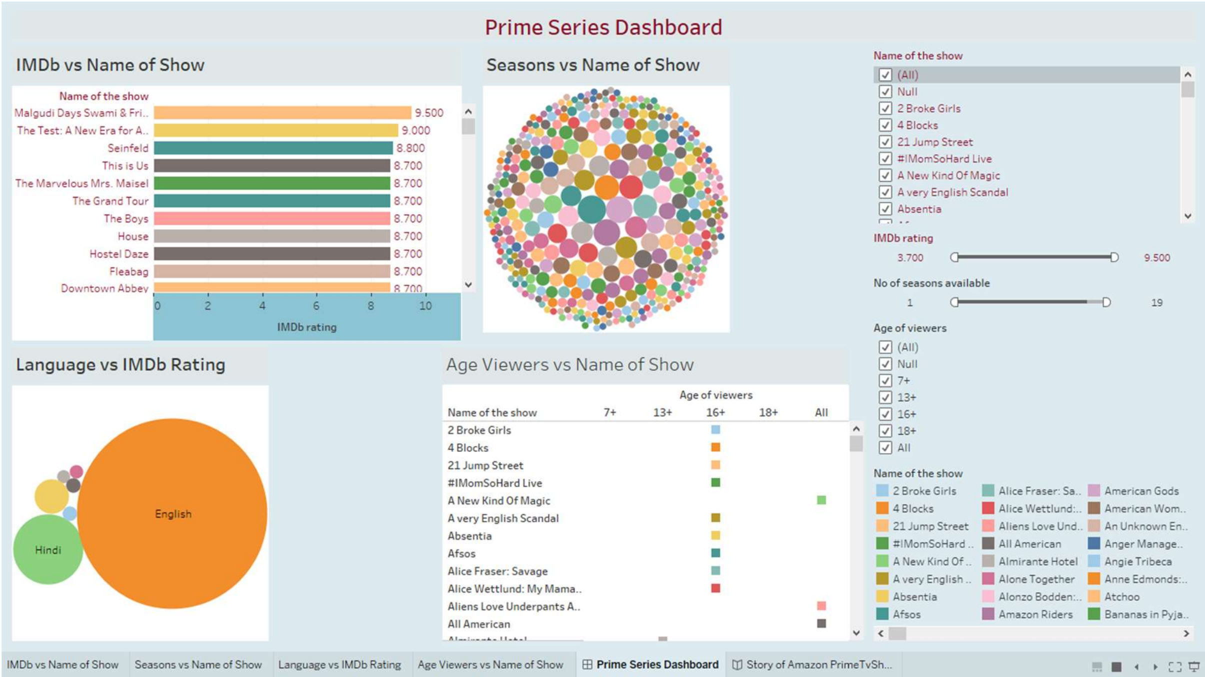
We have created this excel dashboard on Pima Indian Diabetic dataset.



2.4 Tableau

Tableau is a powerful and fastest growing data visualization tool used in the Business Intelligence Industry. It helps in simplifying raw data in a very easily understandable format. Tableau helps create the data that can

be understood by professionals at any level in an organization. It also allows non-technical users to create customized dashboards.
Data analysis is very fast with Tableau tool and the visualizations created are in the form of dashboards and worksheets.



2.5. SAS

Exploratory Data Analysis in SAS

```
data diabetes;

infile "diabetes.csv" dlm="," firstobs=2 dsd;

input Pregnancy Glucose BloodPressure
SkinThickness Insulin BMI DPF Age Outcome;
```

```
run;

proc print data=diabetes;

run;


/* To check is there any missing values present in table*/

proc means data=diabetes nmiss;

run;


/* To check the datatypes of data*/

proc contents data=diabetes;

run;


/* To check the summary  of the data*/

proc summary data=diabetes print n mean median mode stddev min max;

var Pregnancy Glucose BloodPressure SkinThickness Insulin BMI DPF ;

run;


/* Pie diagram of Diabetic women and Non Diabetic women */

proc sgpie data=diabetes ;

pie Outcome/  datalabelloc=outside;  ;

run;
```

```
/* To check the correlation between columns */

proc corr data=diabetes;

run;

/* Result :

   The chance of Diabetes patients  mainly correlates
with the values of Glucose, BMI, Age, Pregnancy*/




/* Histogram of Glucose level in diabetic dataset*/

title"Glucose level in Diabetic and Non Diabetic
women";

proc sgplot  data=diabetes;

histogram Glucose/group=Outcome transparency=0.5
fillattrs=(color=olive);

density Glucose /type=normal group=Outcome;

keylegend /location=inside position=topright
across=1;

run;


/* Histogram of Preganancy count in diabetic dataset
*/

title"Number of Pregancy in Diabetic and Non
Diabetic women";
```

```
proc sgplot  data=diabetes;

histogram Pregnancy/group=Outcome
transparency=0.5;

density Pregnancy /type=normal group=Outcome;

keylegend /location=inside position=topright
across=1;

run;


/* Box plot of Pregnancy count */

proc sgplot data=diabetes;

vbox Pregnancy/group=Outcome;

run;



/* Histogram ofBMI in Diabetes dataset */

title" BMI in Diabetic and Non Diabetic Women";

proc sgplot  data=diabetes;

histogram BMI/group=Outcome transparency=0.5
fillattrs=(color=teal);

density BMI /type=normal group=Outcome ;

keylegend /location=inside position=topright
across=1;

run;


/* To find information of data */
```

```
proc contents data=diabetes ;

run;


/* To compare the Median and Maximum values of
both diabetes and non diabetes patients */

proc means data=diabetes(where=(Outcome=1)) print
median  max ;

var Pregnancy Glucose BloodPressure SkinThickness
Insulin BMI DPF Age;

title "Diabetes Patients";

proc means data=diabetes(where=(Outcome=0)) print
median  max;

var Pregnancy Glucose BloodPressure SkinThickness
Insulin BMI DPF Age;

title "Non Diabetes Patients";

run;

/*.............*/


/* To add one column to the table */

proc sql;

alter table diabetes add Groups char(20);

quit;

run;
```

```
/* update the data by adding age group with
conditions */

proc sql;

update diabetes

set Groups=

CASE WHEN age <= 16 THEN 'Child'

WHEN age <= 30 and age>16 THEN 'Young Adult'

WHEN age <= 45 and age>30 THEN 'Middle-Aged
Adult'

ELSE 'Old-Aged Adult'

END;

QUIT;

run;


/* To show the updated data */

proc print data=diabetes;

run;


/* To create Table with Diabtes patients only*/

proc sql;

create table DPatient as

select
Pregnancy,Glucose,BloodPressure,SkinThickness,Ins
ulin,BMI,DPF,Age,Groups from diabetes where
Outcome=1;
```

```
quit;

proc print data=DPatient;

run;


/* Percentage of Diabetes patients by Age
Categories*/

proc sql;

select Groups, ((COUNT( * ) / ( SELECT COUNT( * )
FROM DPatient)) * 100 ) AS Percentage from DPatient
group by Groups order by Groups;

quit;

run;

/* Around 45% Diabetes patients were Middle-Aged
Adults and 33% were Young Adults.*/


/* To compare the correlation between glucose, BMI
and Insulin within all age groups*/

proc corr data=DPatient(where=(Groups='Young
Adult'));

var Glucose BMI Insulin;

title"Young Adult";

proc corr data=DPatient(where=(Groups='Middle-Aged
Adult'));

var Glucose BMI Insulin;

title"Middle-Aged Adult";
```

```
proc corr data=DPatient(where=(Groups='Old-Aged
Adult'));

var Glucose BMI Insulin;

title"Old-Aged Adult";

run;


/* Bar graph showing Insulin level by Different age
groups */

proc sgplot data=DPatient;

    hbar Groups/response=Insulin stat=mean

    datalabel datalabelattrs=(weight=bold);

    title 'Insulin level in Different Age Groups';


/* Bar graph showing Glucose level by different Age
groups */

proc sgplot  data=DPatient;

    hbar Groups/response=Glucose stat=mean

    datalabel datalabelattrs=(weight=bold)
fillattrs=(color=cadetblue);

    title 'Glucose level in Different Age Groups';

run;


/*Scatter plot showing relationship between Glucose
level and Blood Pressure in Middle Aged women*/
```

```
proc sgplot data=DPatient(where=(Groups='Middle-Aged Adult'));
   scatter x=Glucose y=BloodPressure;
   title 'Relationship between Glucose and Blood Pressure';
run;

/*
```

Conclusion:

Although diabetes affects men and women equally, women are more severely impacted by its consequences.

There are currently over 199 million women living with diabetes, and this is projected to increase to 313

million by 20401. Diabetes is the ninth leading direct cause of death in women globally, causing 2.1 million

deaths each year, most of them were pre-mature1. The issue of women and diabetes is important for several reasons.

This data showing increase of Glucose level, BMI, Number of Pregnancies and Age were reasons to become

diabetic patient. The blood Pressure, Skin Thicknes were not involving greatly to become diabetic.

Diabetes mainly seen on Middle Aged women compared to other age groups. The mean of Insulin level

is relatively low compared to other age groups as well.

***/**