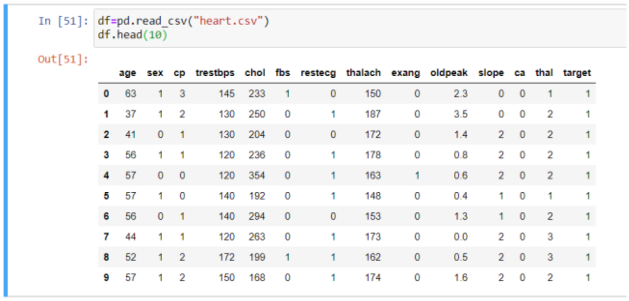
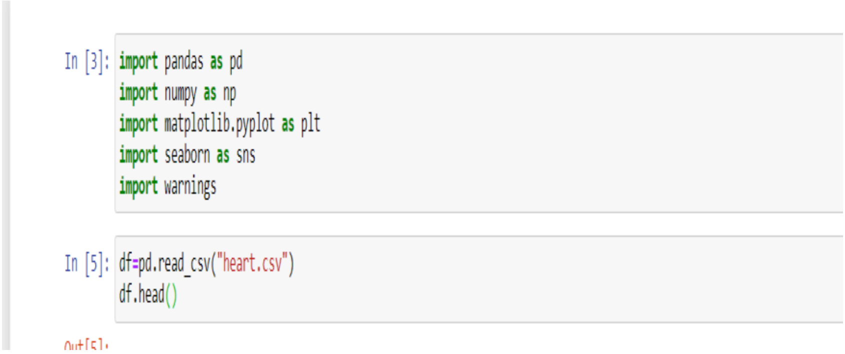
APPENDIX

1. Figure: Dataframe Python



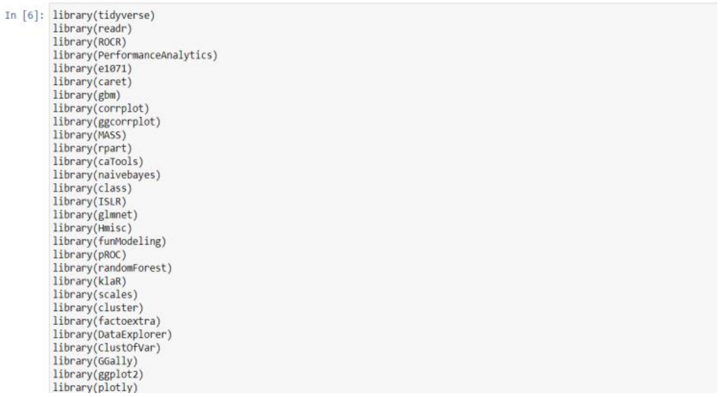
```
In [51]: df=pd.read_csv("heart.csv")
         df.head(10)

Out[51]:
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  ca  thal  target
0   63    1   3       145   233    1        0      150      0      2.3      0   0     1       1
1   37    1   2       130   250    0        1      187      0      3.5      0   0     2       1
2   41    0   1       130   204    0        0      172      0      1.4      2   0     2       1
3   56    1   1       120   236    0        1      178      0      0.8      2   0     2       1
4   57    0   0       120   354    0        1      163      1      0.6      2   0     2       1
5   57    1   0       140   192    0        1      148      0      0.4      1   0     1       1
6   56    0   1       140   294    0        0      153      0      1.3      1   0     2       1
7   44    1   1       120   263    0        1      173      0      0.0      2   0     3       1
8   52    1   2       172   199    1        1      162      0      0.5      2   0     3       1
9   57    1   2       150   168    0        1      174      0      1.6      2   0     2       1
```

2. Figure: Python Libraries



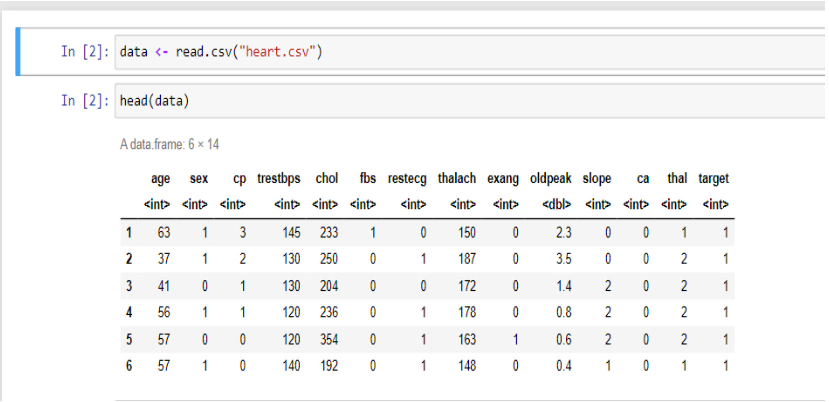
```
In [3]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        import warnings

In [5]: df=pd.read_csv("heart.csv")
        df.head()

Out[5]:
```

3. Figure: R Libraries:



```
In [6]: library(tidyverse)
        library(readr)
        library(ROCR)
        library(PerformanceAnalytics)
        library(e1071)
        library(caret)
        library(gbm)
        library(corrplot)
        library(ggcorrplot)
        library(MASS)
        library(rpart)
        library(caTools)
        library(naivebayes)
        library(class)
        library(ISLR)
        library(glmnet)
        library(Hmisc)
        library(funModeling)
        library(pROC)
        library(randomForest)
        library(klaR)
        library(scales)
        library(cluster)
        library(factoextra)
        library(DataExplorer)
        library(ClustOfVar)
        library(GGally)
        library(ggplot2)
        library(plotly)
```

4. Figure: Dataframe R



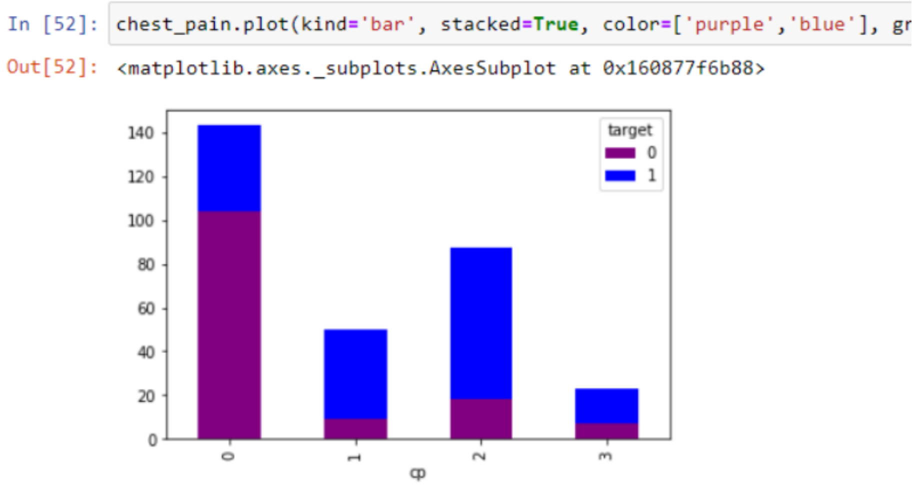
```
In [2]: data <- read.csv("heart.csv")
```

```
In [2]: head(data)
```

A data.frame: 6 × 14

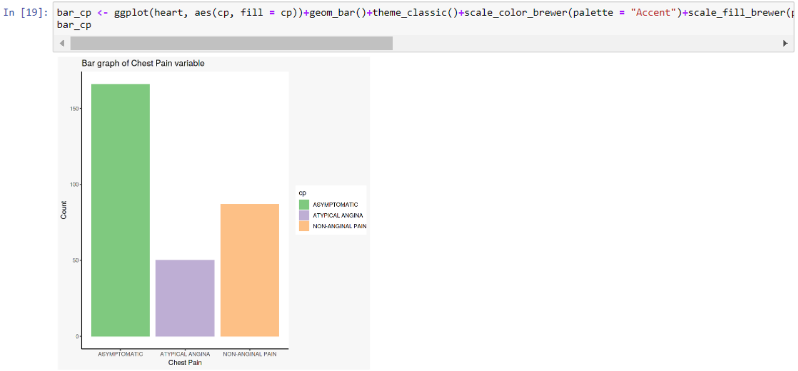
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<dbl>	<int>	<int>	<int>	<int>
1	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
2	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
3	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
4	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
5	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
6	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1

5. Chest Pain bar chart Python



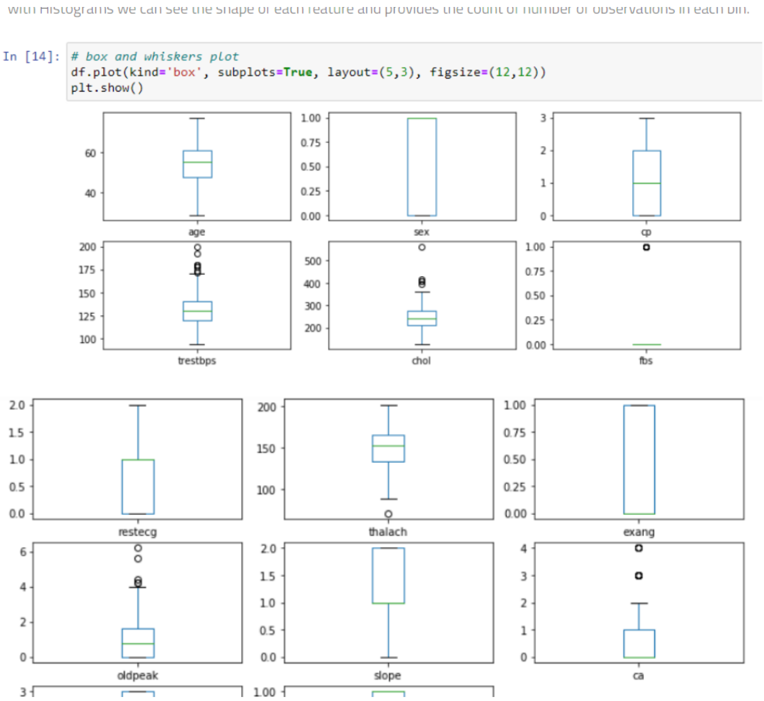
```
In [52]: chest_pain.plot(kind='bar', stacked=True, color=['purple','blue'], gr
Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x160877f6b88>
```

6. Figure: chest pain bar chart in R



```
In [19]: bar_cp <- ggplot(heart, aes(cp, fill = cp))+geom_bar()+theme_classic()+scale_color_brewer(palette = "Accent")+scale_fill_brewer(
         bar_cp
```

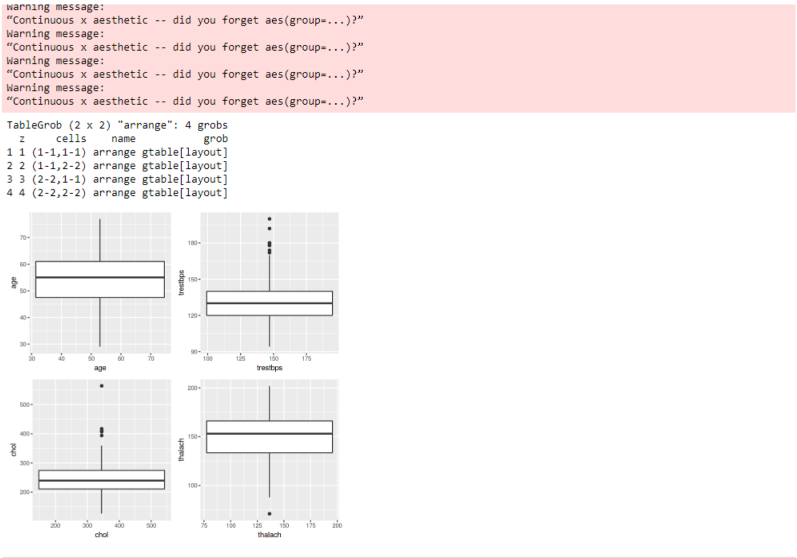
7. Figure: Boxplot python



with Histograms we can see the shape of each feature and provides the count of number of observations in each bin.

```
In [14]: # box and whiskers plot
         df.plot(kind='box', subplots=True, layout=(5,3), figsize=(12,12))
         plt.show()
```

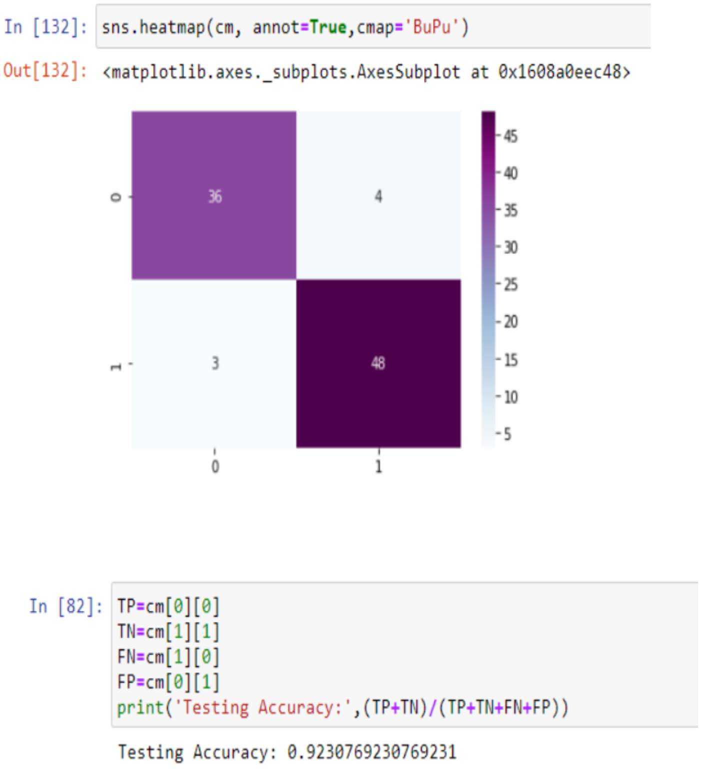
8. Figure: Boxplot R



```
Warning message:
“Continuous x aesthetic -- did you forget aes(group=...)?”
Warning message:
“Continuous x aesthetic -- did you forget aes(group=...)?”
Warning message:
“Continuous x aesthetic -- did you forget aes(group=...)?”
Warning message:
“Continuous x aesthetic -- did you forget aes(group=...)?”
```

```
TableGrob (2 x 2) "arrange": 4 grobs
  z     cells    name           grob
1 1 (1-1,1-1) arrange gtable[layout]
2 2 (1-1,2-2) arrange gtable[layout]
3 3 (2-2,1-1) arrange gtable[layout]
4 4 (2-2,2-2) arrange gtable[layout]
```

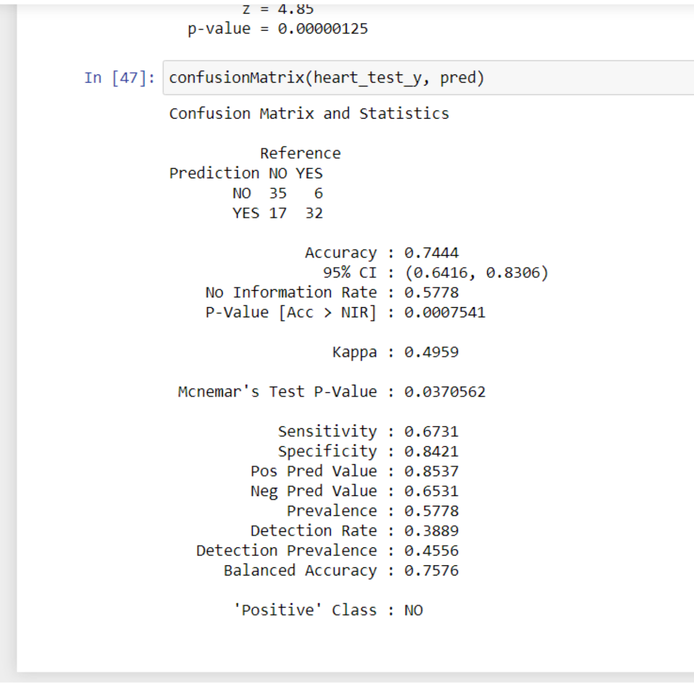
9. Figure: Python confusion matrix



```
In [132]: sns.heatmap(cm, annot=True,cmap='BuPu')
Out[132]: <matplotlib.axes._subplots.AxesSubplot at 0x1608a0eec48>
```

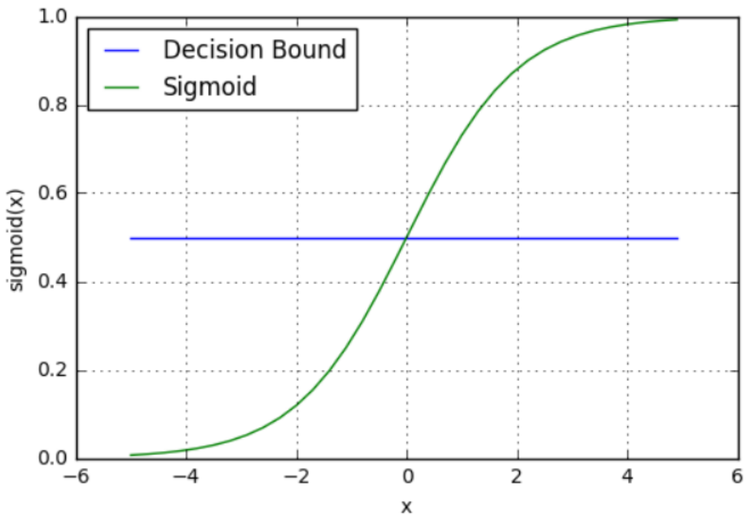
```
In [82]: TP=cm[0][0]
         TN=cm[1][1]
         FN=cm[1][0]
         FP=cm[0][1]
         print('Testing Accuracy:',(TP+TN)/(TP+TN+FN+FP))

         Testing Accuracy: 0.9230769230769231
```

10. Figure: R confusion Matrix



```
                z = 4.85
          p-value = 0.00000125

In [47]: confusionMatrix(heart_test_y, pred)

         Confusion Matrix and Statistics

                   Reference
         Prediction NO YES
                NO  35   6
                YES 17  32

                        Accuracy : 0.7444
                          95% CI : (0.6416, 0.8306)
             No Information Rate : 0.5778
             P-Value [Acc > NIR] : 0.0007541

                           Kappa : 0.4959

          Mcnemar's Test P-Value : 0.0370562

                     Sensitivity : 0.6731
                     Specificity : 0.8421
                  Pos Pred Value : 0.8537
                  Neg Pred Value : 0.6531
                      Prevalence : 0.5778
                  Detection Rate : 0.3889
            Detection Prevalence : 0.4556
               Balanced Accuracy : 0.7576

                'Positive' Class : NO
```

11. Figure Logistic Regression



12. Code Python:

```
import pandas as PD
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

df=pd.read_csv("heart.csv")
df.head(10)

#checkingtheNulValues
df.isnull().sum()

print(df.info())

#checkingtheCorrelationAmongtheAttributes
plt.figure(figsize=(30,15))
sns.heatmap(df.corr(), annot=True, cmap='terrain')
sns.pairplot(data=df)
df.hist(figsize=(12,12), layout=(5,3));
#box and whiskers plot
df.plot(kind='box', subplots=True, layout=(5,3), figsize=(12,12))
plt.show()
#visualizethefeatures and their relation with target(heart disease or no heart disease)
sns.catplot(data=df, x='sex', y='age', hue='target', palette='husl')
df['sex'].value_counts() #207males and 96females
df['cp'].value_counts() #chestPainType
sns.countplot(x='cp', hue='target', data=df, palette='rocket')
#crossTables
gen=pd.crosstab(df['sex'], df['target'])
print(gen)
gen.plot(kind='bar', stacked=True, color=['Skyblue', 'yellow'], grid=False)
chest_pain=pd.crosstab(df['cp'], df['target'])
chest_pain
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
StandardScaler=StandardScaler()
columns_to_scale=['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
df[columns_to_scale]= StandardScaler.fit_transform(df[columns_to_scale])
df.head()
x=df.drop(['target'], axis=1)
y=df['target']
x_train, x_test, y_train, y_test= train_test_split(x,y, test_size=0.3, random_state=40)
print('x_train-', x_train.size)
print('x_test-', x_test.size)
print('y_train-', y_train.size)
print('y_test-', y_test.size)

#appyingLogisticRegression

from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
```

```
modell=lr.fit(x_train, y_train)
prediction1=modell.predict(x_test)

from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test, prediction1)
cm

TP=cm[0][0]
TN=cm[1][1]
FN=cm[1][0]
FP=cm[0][1]
print('Testing Accuracy:', (TP+TN)/ (TP+TN+FN+FP))
sns.heatmap(cm, annot=True, cmap="BuPu")
TP=cm[0][0]
TN=cm[1][1]
FN=cm[1][0]
FP=cm[0][1]
print('Testing Accuracy: '. (TP+TN)/ (TP+TN+FN+FP))
from sklearn.metrics import accuracy_score
accuracy_score(y_test, prediction1)
from sklearn.metrics import classification_report
print(classification_report(y_test, prediction1))
```

13. Code R

```
data <- read.csv("heart.csv")
head(data)
str(data)
summary(data)
colnames(data)
library(tidyverse)
library(readr)
library(ROCR)
library(PerformanceAnalytics)
library(e1071)
library(caret)
library(gbm)
library(corrplot)
library(ggcorrplot)
library(MASS)
library(rpart)
library(caTools)
library(naivebayes)
library(class)
library(ISLR)
library(glmnet)
library(Hmisc)
library(funModeling)
library(pROC)
library(randomForest)
library(klaR)
library(scales)
library(cluster)
library(factoextra)
library(DataExplorer)
library(ClustOfVar)
library(GGally)
library(ggplot2)
library(plotly)

heart <- data %>%
  mutate(sex = if_else(sex == 1, "MALE", "FEMALE"),
         fbs = if_else(fbs == 1, ">120", "<=120"),
         exang = if_else(exang == 1, "YES" ,"NO"),
         cp = if_else(cp == 1, "ATYPICAL ANGINA",
                      if_else(cp == 2, "NON-ANGINAL PAIN", "ASYMPTOMATIC")),
         restecg = if_else(restecg == 0, "NORMAL",
                           if_else(restecg == 1, "ABNORMALITY", "PROBABLE OR DEFINITE")),
        slope = as.factor(slope),
        ca = as.factor(ca),
        thal = as.factor(thal),
        target = if_else(target == 1, "YES", "NO")
        ) %>%
  mutate_if(is.character, as.factor) %>%
  dplyr::select(target, sex, fbs, exang, cp, restecg, slope, ca, thal, everything())
colnames(data)
colnames(heart)

head(data)
head(heart)

summary(heart)

boxplot(heart)
boxplot(heart[,10:13])
library(gridExtra)
box_plot <- grid.arrange(ggplot(heart, aes(age, age))+geom_boxplot(),
            ggplot(heart, aes(trestbps, trestbps))+geom_boxplot(),
            ggplot(heart, aes(chol, chol))+geom_boxplot(),
            ggplot(heart, aes(thalach, thalach))+geom_boxplot()
)
box_plot

bar_graph <- grid.arrange(ggplot(heart, aes(x = sex, fill = target))+geom_bar(position = "fill"),
                        ggplot(heart, aes(x = fbs, fill = target))+geom_bar(position = "fill"),
                        ggplot(heart, aes(x = exang, fill = target))+geom_bar(position = "fill")
                       )
bar_graph

grid_bar <- grid.arrange(ggplot(heart, aes(x = cp, fill = target))+geom_bar(position = "fill")+ theme(axis.text.x = element_text(angle = 90, hjust = 1)),
            ggplot(heart, aes(x = restecg, fill = target))+geom_bar(position = "fill")+
theme(axis.text.x = element_text(angle = 90, hjust = 1))

           )

   grid_bar


   bar_target <- ggplot(heart, aes(target, fill =
target))+geom_bar()+theme_classic()+scale_color_brewer(palette =
"Accent")+scale_fill_brewer(palette = "Accent")+theme(plot.background = element_rect(fill
= "grey97"))+labs(title = "Bar graph of Target variable", x = "Heart Disease", y = "Count")

   bar_target


   ggplotly(bar_target)

   bar_cp <- ggplot(heart, aes(cp, fill =
cp))+geom_bar()+theme_classic()+scale_color_brewer(palette =
"Accent")+scale_fill_brewer(palette = "Accent")+theme(plot.background = element_rect(fill
= "grey97"))+labs(title = "Bar graph of Chest Pain variable", x = "Chest Pain", y = "Count")

   bar_cp

   ggplotly(bar_cp)

   hist_age <- ggplot(heart, aes(age, fill = sex))+geom_histogram(bins =
30)+theme_classic()+scale_color_brewer(palette = "Accent")+scale_fill_brewer(palette =
"Accent")+theme(plot.background = element_rect(fill = "grey97"))+labs(title = "Histogram
of age variable with sex", x = "age", y = "count")

   hist_age

   ggplotly(hist_age)

   age_point <- ggplot(heart, aes(age, chol, color = sex, size =
chol))+geom_point()+geom_smooth()+theme_classic()+theme(plot.background =
element_rect(fill = "grey97"))+ggtitle("Age by Chol")

   age_point

   ggplotly(age_point)

   bp_box <- ggplot(heart, aes(x=sex,y=trestbps))+geom_boxplot(fill =
"pink")+facet_grid(~cp)+geom_smooth()+theme_classic()+theme(plot.background =
element_rect(fill = "grey97"))+labs(title = "Comparison of Blood pressure across pain type",
x = "Sex", y = "Blood Pressure")

   bp_box


   ggplotly(bp_box)

   chol_box <- ggplot(heart, aes(x=sex, y=chol))+geom_boxplot(fill =
"turquoise")+facet_grid(~cp)+geom_smooth()+theme_classic()+theme(plot.background =
element_rect(fill = "grey97"))+labs(title = "Comparison of Cholestoral across pain type ", x = "Sex", y = "Chol")
    chol_box
    ggplotly(chol_box)
    cor_heart <- cor(heart[,10:14])
    cor_heart

    ggcorrplot(cor_heart,lab = TRUE)
    split <- createDataPartition(heart$target, time = 1, list = FALSE, p = 0.70)

    heart_train <- heart[split,]
    heart_test <- heart[-split,]
    dim(heart)
    dim(heart_train)
    dim(heart_test)
    heart_test_x <- heart_test %>% dplyr::select(-target)
    heart_test_y <- heart_test$target
    head(heart_train)
    head(heart_test_x)
    plot_num(heart)
    freq(heart)
    heart_mod <- glm(target~., data = heart_train, family = "binomial")
    summary(heart_mod)
    heart_mod
    options(scipen = 999)
    summary(heart_mod)
    test_pred <- predict(heart_mod, type = "response", newdata = heart_test_x)
    head(test_pred)
    head(heart_test_y)
    library(pROC)
    roc <- roc(heart_test_y ~ test_pred, plot = TRUE, print.auc = TRUE,  col="red")
    roc
    pred <- ifelse(test_pred >= 0.8218, 'YES', 'NO')
    head(pred)
head(heart_test_y)
str(heart_test_y)
str(pred)
heart_test_y <- as.factor(heart_test_y)
pred <- as.factor(pred)
library(irr)
kappa2(data.frame(heart_test_y, pred))
confusionMatrix(heart_test_y, pred)
```

ACKNOWLEDGMENT

I have completed this task under the mentorship of HM Mostafizur Rahman (Lecturer and department head) school of IT, Bac International Study Centre. I am doing a Bachelor's in computer science. In this fall semester, I have completed the course Data science for business, where I have learned the fundamental of data science, machine learning algorithms from my mentors as course instructors. This work has been assigned as a project assignment. I would like to express my special thanks to my mentor for inspiring us to complete the working paper. Without his active guidance, help, cooperation & encouragement, I would not have been able to write this paper. I am very thankful for his guidance and help in the completion of this paper.

Thanking You.