DIABETES AFFECT ANALYSIS

SC614 Statistical Methods with Lab R

"Statistics is a tool for enhancing intuition"

By: Kandarp Parmar(202118027)

Vidhi Shah(202118037)

Table of Contents

1. Loading and Displaying the data

```
> library(readr)
> dia<-
read.csv("https://raw.githubusercontent.com/Datamanim/datarepo/main/diabetes/train.csv")
> View(dia)
```

2. Displaying first 5 and last 5 records of the dataset

```
> head(dia,5)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
1           3     102           74             0       0 29.5
2           2     144           58            33     135 31.6
3           5     136           82             0       0  0.0
4          13     145           82            19     110 22.2
5           1     117           60            23     106 33.8
  DiabetesPedigreeFunction Age Outcome
1                    0.121  32       0
2                    0.422  25       1
3                    0.640  69       0
4                    0.245  57       0
5                    0.466  27       0

> tail(dia,5)
    Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
610           3     187           70            22     200 36.4
611          11     138           74            26     144 36.1
612           5     168           64             0       0 32.9
```

```
613         1     79          60          42     48 43.5
614         1    140          74          26    180 24.1
    DiabetesPedigreeFunction Age Outcome
610                    0.408  36       1
611                    0.557  50       1
612                    0.135  41       1
613                    0.678  23       0
614                    0.828  23       0
```

3. Displaying mean of every column

```
> install.packages("dplyr")
> dia %>% summarise_each(funs(mean))
  Pregnancies Glucose BloodPressure SkinThickness  Insulin    BMI
1    3.843648 120.614     69.34853     21.22638 79.97557 31.9816
  DiabetesPedigreeFunction     Age   Outcome
1                 0.462513 33.25733 0.3485342
```

4. Displaying median

```
> median(dia$Glucose)
[1] 117
> median(dia$Age)
[1] 29
```

5. Displaying mode

```
> require(modeest)
```

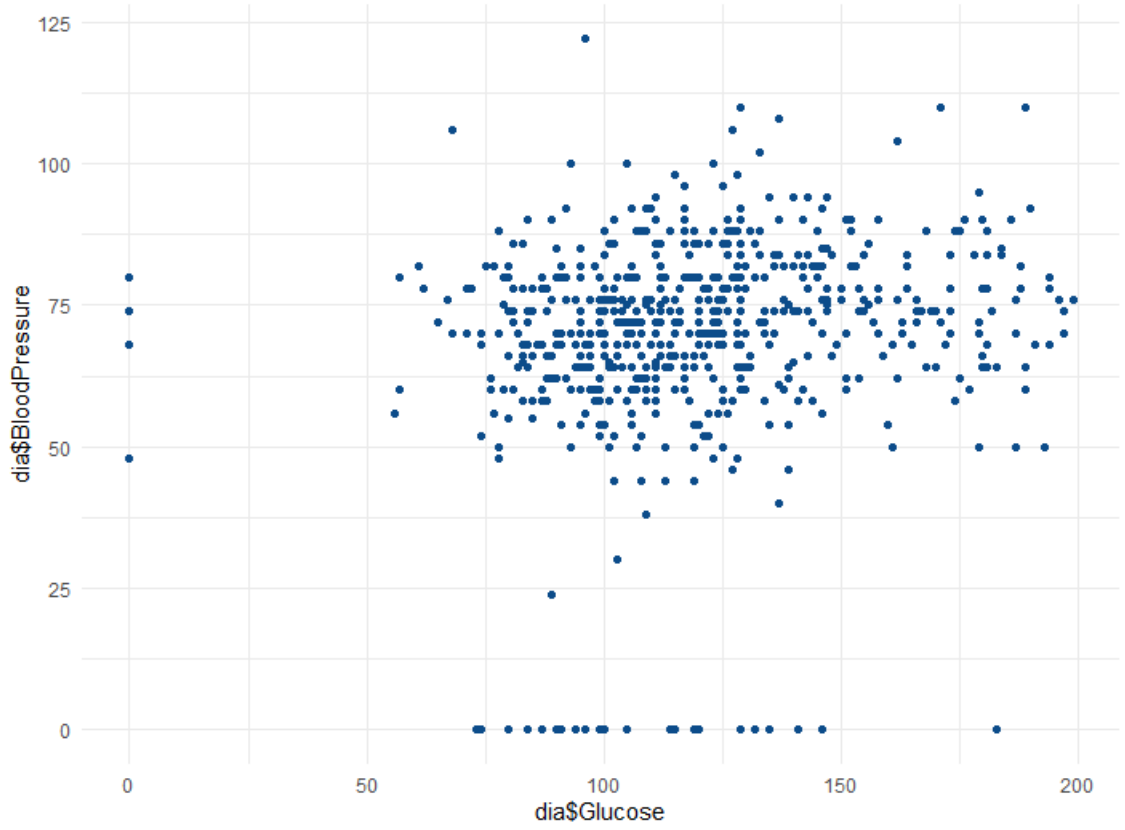
```
Loading required package: modeest
> mfv(dia$SkinThickness)
[1] 0
```

6. Correlation between Glucose and Blood Pressure

```
> cor(dia$Glucose,dia$BloodPressure)
[1] 0.1521056
```

7. Scatter plot between Blood Pressure and Glucose

```
> library(ggplot2)
> ggplot(dia)+
+     aes(x=dia$Glucose,y=dia$BloodPressure)+
+     geom_point(colour="#0c4c8a")+
+     theme_minimal()
```

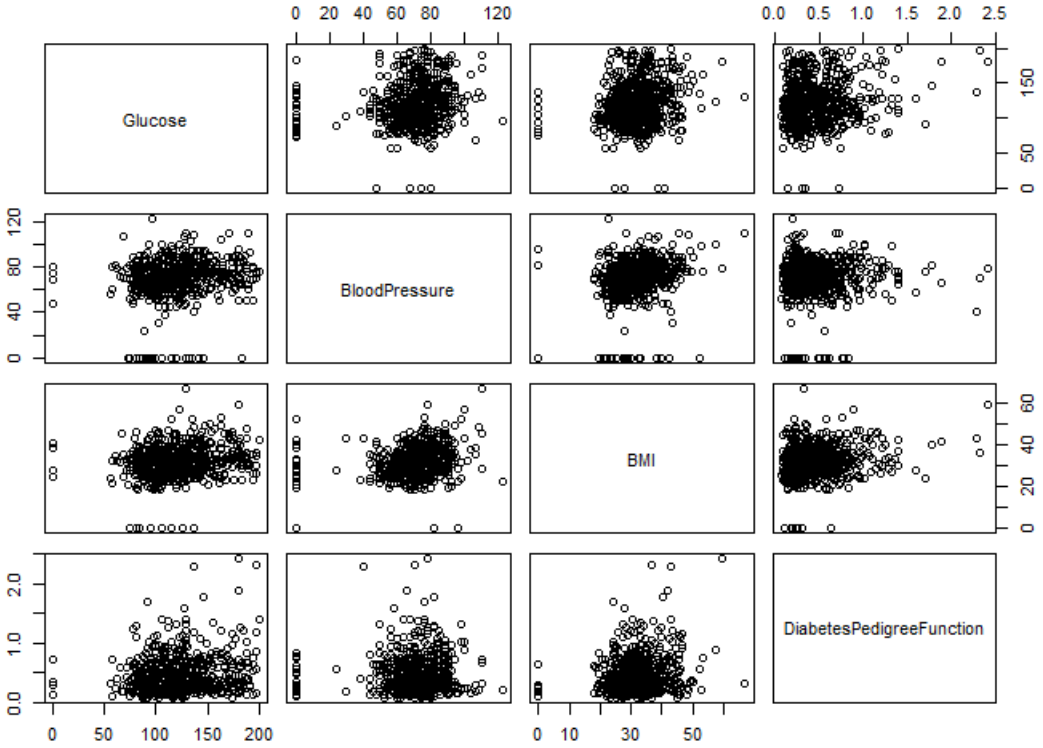


Inference:

A Scatter plot is often used to study the correlation between two variables and how they are connected. Here, from the above calculated correlation we can see that there is no correlation between Glucose and Blood Pressure.

8. Plotting a pair plot

> pairs(dia[,c(2,3,6,7)])



Inference:

A pairwise plot is used for the study of the correlation of multiple scatter plots for different variables at a time. Here, we can see that all the plots are scattered randomly, so none of the variables are having any correlation between them.

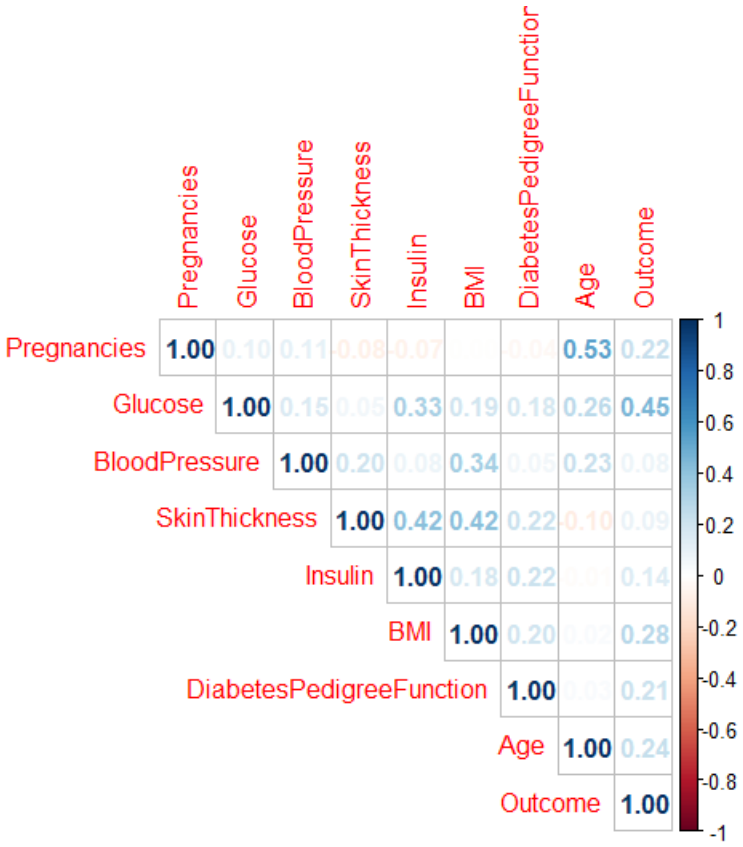
9. Plotting the corrplot

> require(corrplot)

Loading required package: corrplot

corrplot 0.90 loaded

> corrplot(cor(dia[,c(1,2,3,4,5,6,7,8,9)]),method= "number",type="upper")

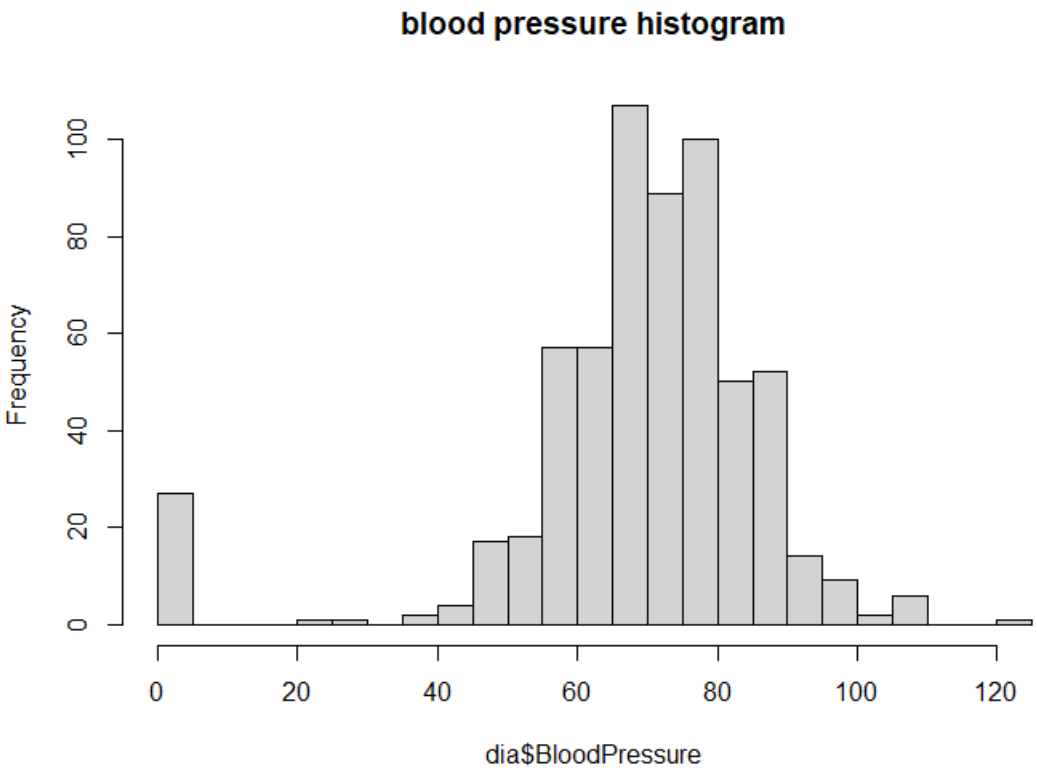


Inference:

Corrplot which is also known as the heat map. By this plot, it becomes easy to find the exact value of correlation between different variables. This can be done with the r function, but for better visualization, we have created a plot that makes it interesting.

10. Plotting histogram of blood pressure

```
> hist(dia$BloodPressure,breaks = 20,main = "blood pressure histogram")
```



Inference:

Here, we are plotting the histogram for the blood pressure column of our dataset which gives the required output.

11. Plotting a histogram and comparing it with the probability density function of the chi-square distribution

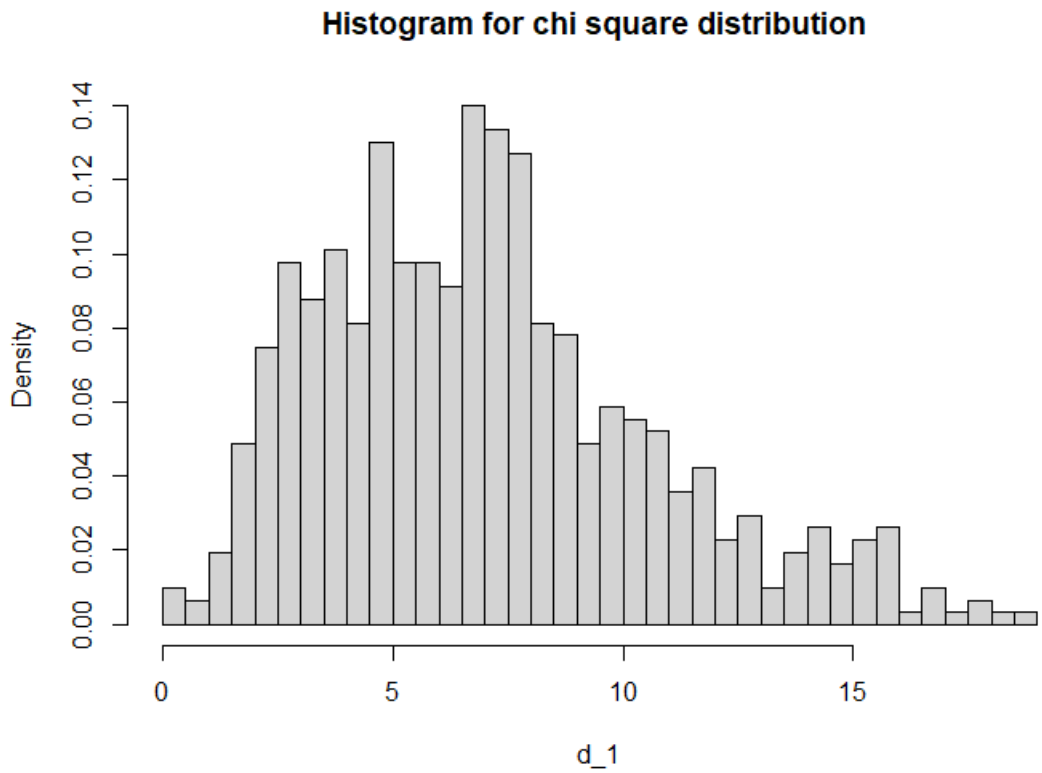
> d_1<-rchisq(dia$BloodPressure,df=7) #we plot a histogram and compare it to the probability density function of the χ2-distribution with df=7

> hist(d_1,

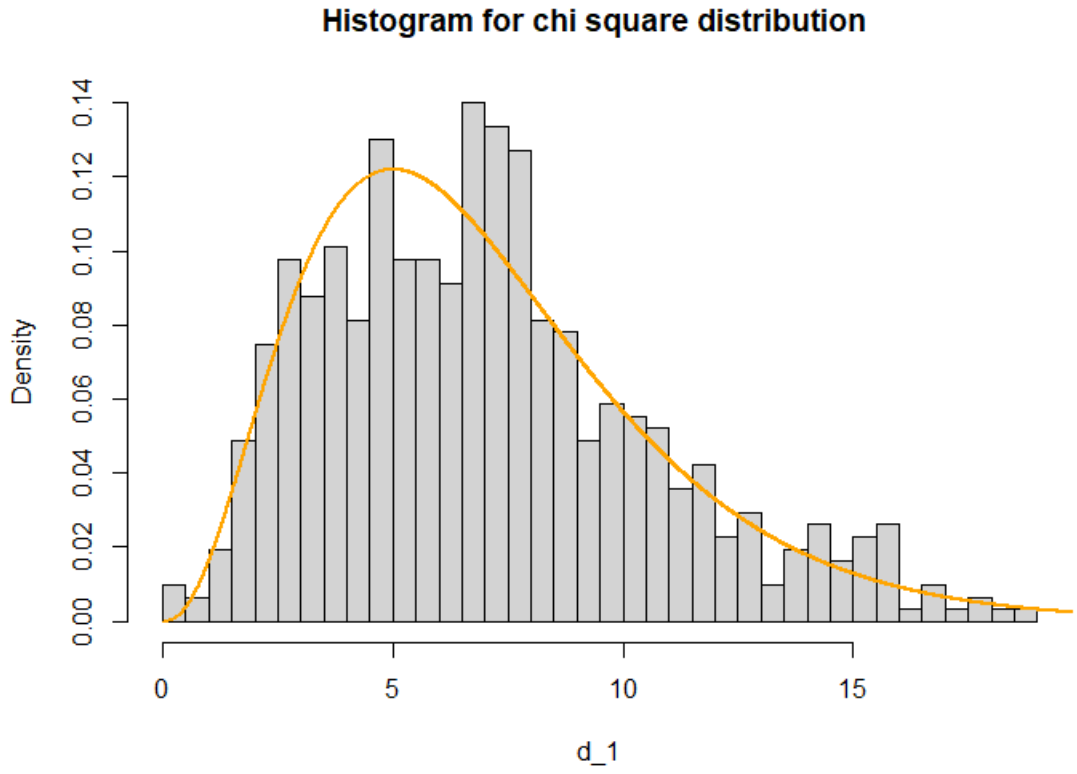
+ breaks = 50,

+ freq = FALSE,

+ main = ('Histogram for chi square distribution '))



> curve(dchisq(x, df = 7), from = 0, to = 25, n = 5000, col= 'orange', lwd=2, add = T)

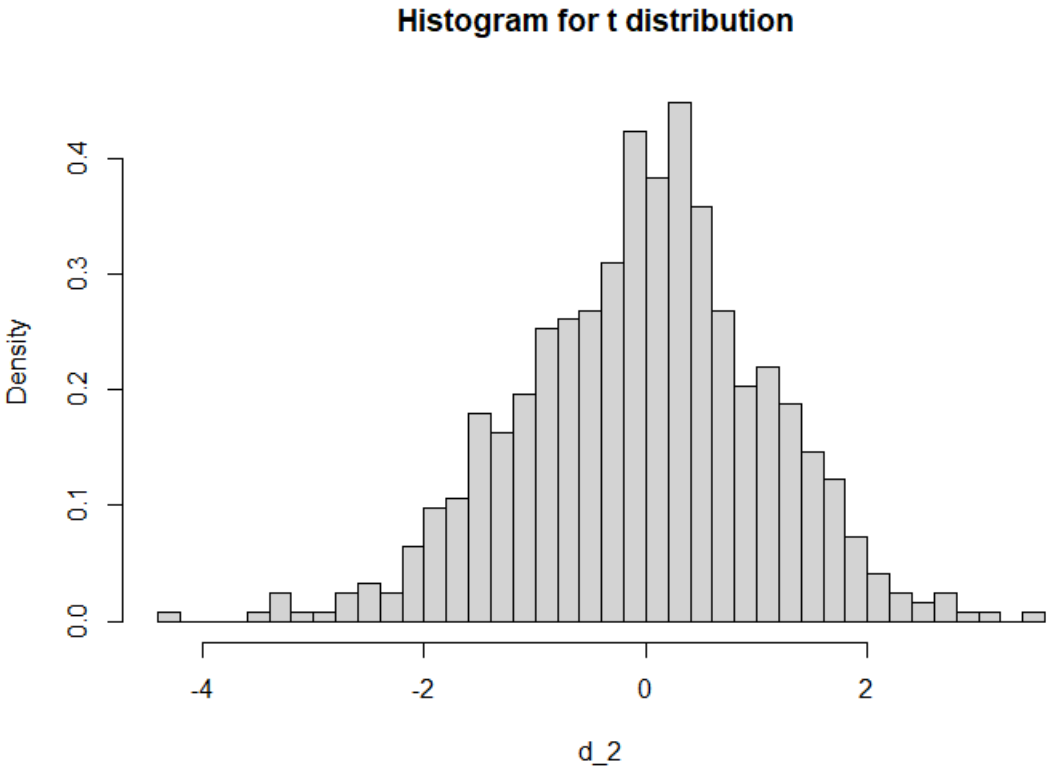


Inference:

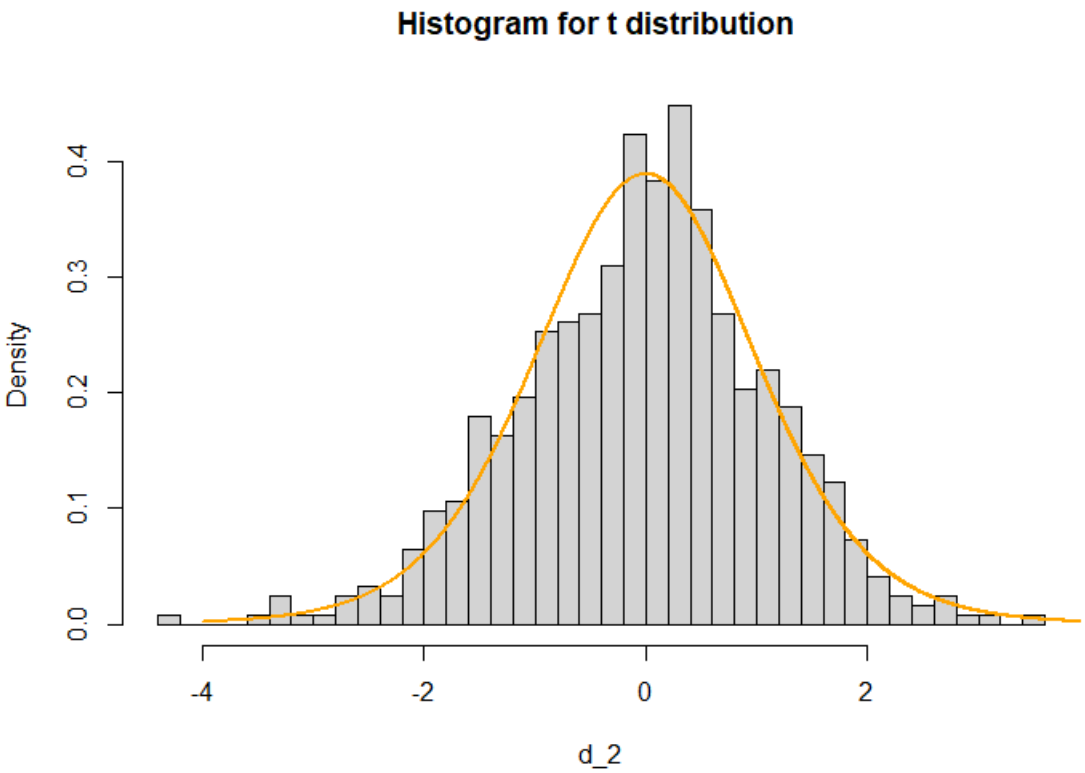
Here, in this plot, firstly we are plotting a histogram of the blood pressure column with 7 degrees of freedom and then we are plotting the curve for the same. They nearly match, so we can assume that for the large value taken in our dataset it will lead to a normal curve.

12. Plotting a histogram and comparing it with the probability density function of the t-distribution

```
> d_2<-rt(dia$Glucose,df=10)  #we plot a histogram and compare it to the probability density function of the t-distribution with df=10
> hist(d_2,freq = FALSE,breaks=50)
>   hist(d_2,
+    breaks = 50,
+    freq = FALSE,
+    main = ('Histogram for t distribution '))
```




```
>   curve(dt(x, df = 10), from = -4, to = 4, n = 5000, col= 'orange', lwd=2, add = T)
```



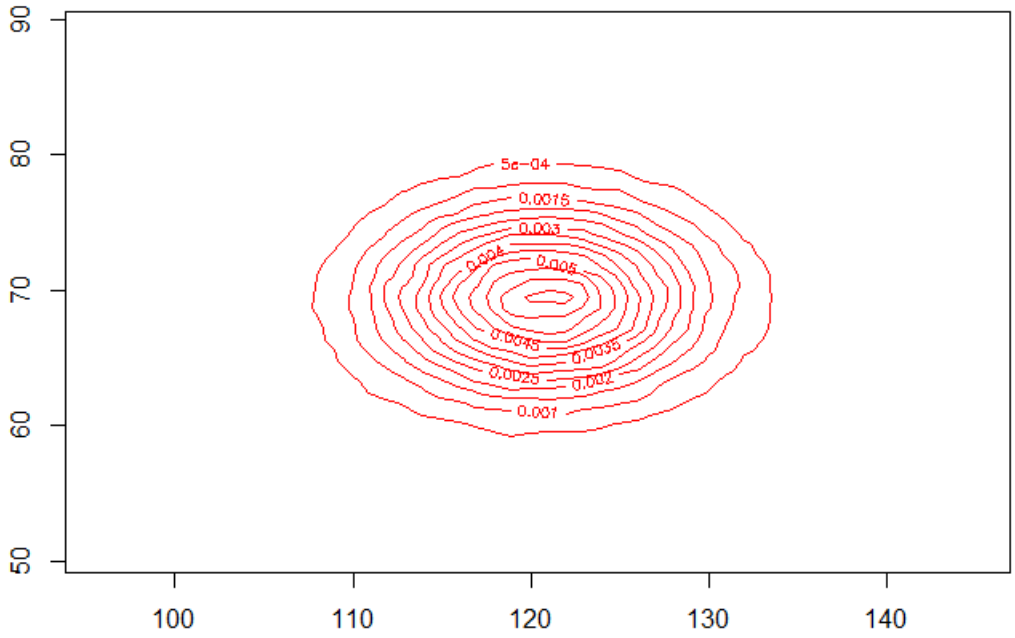
Inference:

This plot is for the t distribution. Here also, we get the same inference that as the number of observations will rise we will get the normal curve.

13. Function for Contour Plot

```
> contour_plot<-function(x,y)
+ {
+   mu1<-mean(x)
+   mu2<-mean(y)
+   mu3<-c(mu1,mu2)
+   c1<-cor(x,y)
+   s1<-sqrt(var(x))
+   s2<-sqrt(var(y))
+   sigma1<-matrix(c(s1,c1,c1,s2),ncol = 2)
```

```
+    library(MASS)
+    bivn<-mvrnorm(100000,mu=mu3,Sigma = sigma1)
+    head(bivn)
+    bivn.kde<-kde2d(bivn[,1],bivn[,2],n=50)
+    contour(bivn.kde,col="red")
+ }
> contour_plot(dia$Glucose, dia$BloodPressure)
```



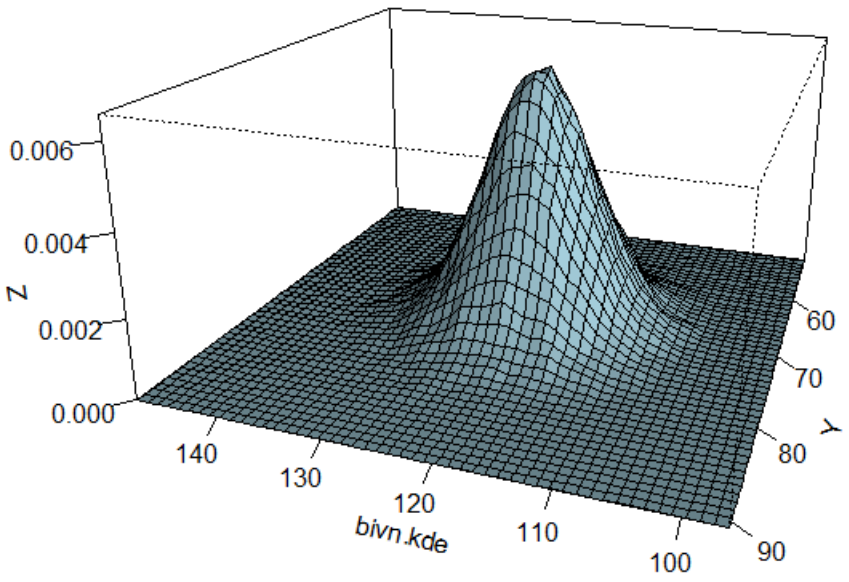
Inference:

When a 3d plot is sliced from the z-axis, at that time we get a contour plot.

14. Function for Perspective plot

```
> perspec_plot<-function(x,y)
+ {
+    mu1<-mean(x)
```

```
+    mu2<-mean(y)
+    mu3<-c(mu1,mu2)
+    c1<-cor(x,y)
+    s1<-sqrt(var(x))
+    s2<-sqrt(var(y))
+    sigma1<-matrix(c(s1,c1,c1,s2),ncol = 2)
+    library(MASS)
+    bivn<-mvrnorm(100000,mu=mu3,Sigma = sigma1)
+    head(bivn)
+    bivn.kde<-kde2d(bivn[,1],bivn[,2],n=50)
+    persp(bivn.kde, theta = 200, phi = 20,
+          shade = 0.75, col = "light blue", expand = 0.5, r = 2,
+          ltheta = 240, ticktype = "detailed")
+ }
> perspec_plot(dia$Glucose, dia$BloodPressure)
```



Inference:

This is a 3d model of the dataset, by which we can get to know that, what is the spread of our data.

15. Two-sided confidence interval

```
> cd_normalsigma_unknown<-function(n,alpha)
+    {
+      mu<-mean(n)
+      s=sqrt(var(n))
+      len1<-length(n)
+      z1<-qt(1-(alpha/2),df=((len1)-1))
+      f1<-(mu-(z1*(s/sqrt(len1))))
+      f2<-(mu+(z1*(s/sqrt(len1))))
+      f<-c(f1,f2)
+      return(f)
+      }
> cd_normalsigma_unknown(dia$Glucose, 0.05)
[1] 118.1189 123.1091
```

Inference:

Implementing the above function would lead us to the intervals of the particular column. For this dataset column glucose is having a confidence interval between (118.1189, 123.1091).

16. Hypothesis testing

```
> null_hypothesis<-function(samp,a,alpha)
+ {
```

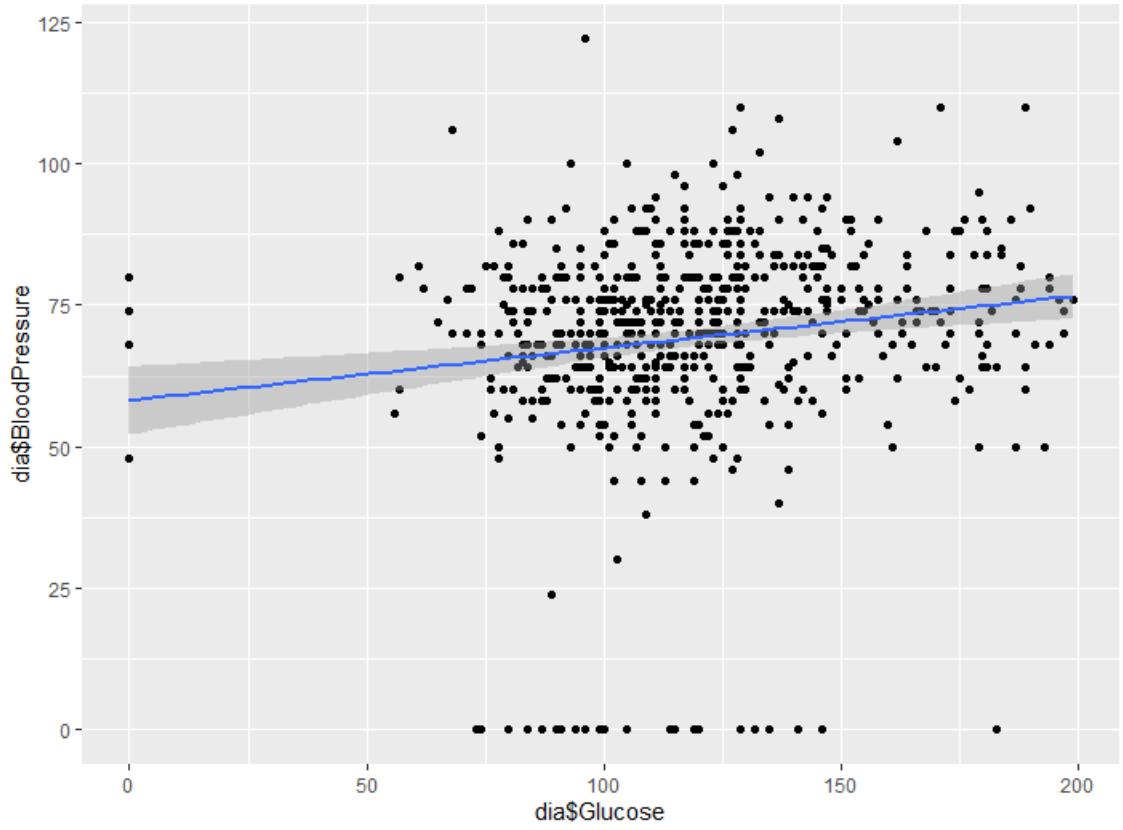
```
+   xbar<-mean(samp)
+   s<-sqrt(var(samp))
+   len1<-length(samp)
+   z_stat<-qt(1-(alpha/2),df=((len1)-1))
+   z1<-(xbar-a)*sqrt(len1)/s
+   if(abs(z1)<=z_stat){
+     print("Hyptothesis is in acceptance region")
+   }else{
+     print("Hypothesis is in rejectance region")
+   }
+ }
> null_hypothesis(dia$Glucose, 94.93, 0.05)
[1] "Hypothesis is in rejectance region"
```

Inference:

By this function, firstly we create a hypothesis which means an assumption, secondly we perform various tests to get the output and make a conclusion that whether the assumption is right or not. The dataset column named glucose gave the output that the null hypothesis was not accepted for the entered value.

17. Linear regression Plot

```
> ggplot(dia,aes(x=dia$Glucose,y=dia$BloodPressure))+
+   geom_point()+
+   stat_smooth(method = lm)
```



Inference:

From linear regression, we get to know that how the values are depending on each other and how we can get the value of the response variable from the predictor variable.

18. Linear regression model

```
> model<-lm(BloodPressure~Glucose,data=dia)
> model

Call:
lm(formula = BloodPressure ~ Glucose, data = dia)

Coefficients:
(Intercept)     Glucose
   58.18335     0.09257
```

```
> new.glu<-data.frame(Glucose=c(180,105,126))
> predict(model,newdata = new.glu)
    1      2      3
74.84587 67.90315 69.84711
```

Inference:

From the above analysis, we get to know the value of the intercept and slope of the line which is plotted. And then we are taking any random values from which we get the output of the response variable.

19. Conf interval for Linear Regression model

```
> confint(model)
                2.5 %     97.5 %
(Intercept) 52.23140041 64.1352926
Glucose     0.04481971  0.1403194
```

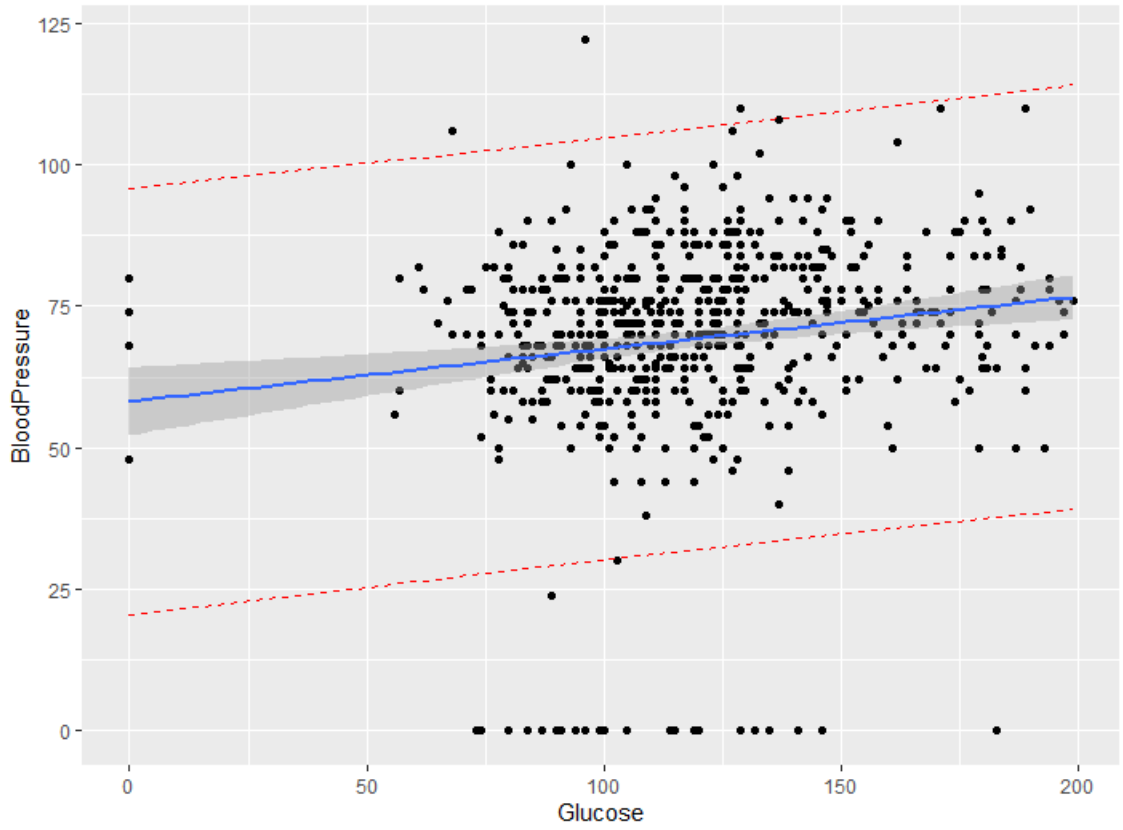
Inference:

This is the confidence interval for the regression model which we have created.

20. Plot with a prediction interval

```
> pred.int<-predict(model,interval="prediction")
> mydata<-cbind(dia,pred.int)
> p<-ggplot(mydata,aes(Glucose,BloodPressure))+
+    geom_point()+
+    stat_smooth(method=lm)
> p+geom_line(aes(y=lwr),color="red", linetype="dashed")+
```

```
+    geom_line(aes(y=upr),color="red", linetype="dashed")
```



Inference:

From this plot, we are creating the prediction interval which shows us the interval in which our data will lie and what will be the output of the response variable if we test for any value of the predictor variable.

21. Conclusion and Learning

In this project, We distinguished data objects from functions and showed how to create and change scalar things in R. We also learned that different objects are designed and altered by applying functions.

We got to know that there are many pre-packaged graphical commands — like hist(), barplot(), or boxplot() — that combine several aspects and provide options for quickly generating some particular type of visualization. On the other hand, there are many low-level plotting functions for designing new visualizations from scratch, or for modifying existing plots. Rather than using a range of specialized tools, we could design a toolbox that provides many different functions in a systematic fashion (e.g., by sharing the same arguments and command syntax for different visualizations). Such a toolbox is provided by the ggplot2 package which we used in our Project.

In terms of tasks, we also used the dplyr functions which is mainly aim to explicate and summarize the data contained in a table. We also used many different libraries like readr for reading datasets, tidyverse for whole package of all libraries and modeest for finding mode.

Summing up the learning part, we understood the application of RStudio and how it can be helpful to analyse data and find valuable inferences. Also, how we can do prediction modelling and regression models in RStudio. The Dynamicity of programming in this language adds an immense value of Knowledge and Data-Analysing techniques which can be handy for us in future.