---
title: "Project 2"
author: "Calvin Hoang"
date: "2/22/2023"
output:
  pdf_document: default
  html_document: default

Heart Disease Dataset

```{r, echo = FALSE}
heart_disease <- read.csv("C:/Users/hoang/Downloads/STA 106/heart_disease_health_indicators_BRFSS2015.csv", header = TRUE)
```

Part 1: Exploring beyond One-way ANOVA with simulated data.

What if there exist huge imbalances among sample sizes {n1,n2,..,nK }? For instance, n1 = n2 = .. = n_k-1 = 100, but n_k = 10^6.

- Creating 5 different samples with normal distribution so that we can compare the mean and the variances of each one of them:

```{r, echo = FALSE}
# generate normal random variable
x=rnorm(1000000,2,4)
print(c(mean(x),var(x)))
y=rnorm(1000,1,3)
print(c(mean(y),var(y)))
```

```
[1]  1.998463 15.989615
[1] 0.9171463 8.2485943
```

- After running many tests, there seems to be a significant difference between the variances of each sample, but not the mean. For example, in one of the tests, the means of the variables x and y have small difference. However, the variance of sample x is significantly different from the variances of y. In addition, I see that n = 1000 and n = 10^6 sometimes have significant difference in the mean. I conclude that as the sample sizes change significantly, there is a significant change between the variances of each sample; however, the mean doesn't seem to change much.

- Suppose there are three levels and simulate the data such that μ_i=mean1[i], σ_i=sd1[i], n_i=samples[i]

```{r, echo = FALSE}
K=3
# mean1=(mu1,mu2,mu3)
mean1=c(1,1,1)
# sd1=(sigma1,sigma2,sigma3)
sd1=c(1,1,1)
# samples=(n1,n2,n3)
samples=c(10,10,10)

# initialize mydata
mydata=data.frame()

for(i in 1:K){
  Yi=rnorm(samples[i],mean=mean1[i],sd=sd1[i])
  data_i=cbind(Yi,i)
  mydata=rbind(mydata,data_i)
}
colnames(mydata)=c('y','level')
# convert to factor variable
mydata$level=as.factor(mydata$level)
mydata
```

Description: df [30 × 2]

y <dbl>	level <fctr>
-0.736315082	1
1.653829811	1
-0.524445805	1
1.362333749	1
0.668768237	1
1.308640037	1
0.861837035	1
-1.124437677	1
0.284578178	1
1.308885858	1

1-10 of 30 rows

Previous 1 2 3 Next

```{r, echo = FALSE}
Yi=rnorm(samples[i],mean=mean1[i],sd=sd1[i])
Yi
```

```
[1]  0.81284100 -0.48946579  1.89282994  1.75195692  0.31836335  2.11366032 -0.12228935  0.42272354  0.01347734  2.36599727
```

```{r, echo = FALSE}
data_i=cbind(Yi,i)
data_i
```

```
               Yi i
 [1,]  0.81284100 3
 [2,] -0.48946579 3
 [3,]  1.89282994 3
 [4,]  1.75195692 3
 [5,]  0.31836335 3
 [6,]  2.11366032 3
 [7,] -0.12228935 3
 [8,]  0.42272354 3
 [9,]  0.01347734 3
[10,]  2.36599727 3
```

```{r, echo = FALSE}
fit1=aov(y~level,data=mydata)
anova(fit1)
```

```
Analysis of Variance Table

Response: y
           Df  Sum Sq Mean Sq F value Pr(>F)
level       2  0.9561 0.47805   0.476 0.6264
Residuals 27 27.1167 1.00432
```

```{r, echo = FALSE}
result1=anova(fit1)
result1$`Pr(>F)`[1]
```

```
[1] 0.6263854
```

- Extracting only the p-value, we get 0.6263854.

- Now we want to repeat this process by, for example, 10000 times since one result of the experiment won't be enough.

```{r, echo = FALSE}
# B number of replicates
B=10000
# save all p-values in the vector pvalues1
pvalues1=numeric(B)
for(b in 1:B){
  K=3
  # mean1=(mu1,mu2,mu3)
  mean1=c(1,1,1)
  # sd1=(sigma1,sigma2,sigma3)
  sd1=c(1,1,1)
  # samples=(n1,n2,n3)
  samples=c(10,10,10)

  # initialize mydata
  mydata=data.frame()

  for(i in 1:K){
    Yi=mean1[i]+rnorm(samples[i],mean=0,sd=sd1[i])
    data_i=cbind(Yi,i)
    mydata=rbind(mydata,data_i)
  }
  colnames(mydata)=c('y','level')
  # convert to factor variable
  mydata$level=as.factor(mydata$level)
  # fit anova model
  fit1=aov(y~level,data=mydata)

  # extract p-value
  result1=anova(fit1)
  pvalues1[b]=result1$`Pr(>F)`[1]
}
```

- Let's compare each times that we get the p-value with the significance level of 0.05. Then the ratio that we reject the null hypothesis:

```{r, echo = FALSE}
sum(pvalues1<0.05)/B
```

```
[1] 0.0497
```

- Here we can see that the ratio is below the significant level of 0.05, which means that the F-test works under this setting. Therefore, the assumption is not violated and the results are accurate.

- However, if we change the sample sizes, will the assumptions get violated? Let's find out!

- Here we simulate the data such that μ_i=mean2[i], σ_i=sd2[i], n_i=samples[i] (the samples are different)

```{r, echo = FALSE}
K=3
# mean2=(mu1,mu2,mu3)
mean2=c(1,1,1)
# sd2=(sigma1,sigma2,sigma3)
sd2=c(1,1,1)
# samples=(n1,n2,n3)
samples=c(10,50,100)

# initialize mydata
mydata=data.frame()

for(i in 1:K){
  Yi=rnorm(samples[i],mean=mean2[i],sd=sd2[i])
  data_i=cbind(Yi,i)
  mydata=rbind(mydata,data_i)
}
colnames(mydata)=c('y','level')
# convert to factor variable
mydata$level=as.factor(mydata$level)
mydata
```

Description: df [160 × 2]

y <dbl>	level <fctr>
1.624378042	1
2.391097108	1
-1.787896206	1
1.820441999	1
0.621876740	1
2.211845032	1
0.379279723	1
-0.067619631	1
-0.666321750	1
1.923704006	1

1-10 of 160 rows

Previous 1 2 3 4 5 6 ... 16 Next

```{r, echo = FALSE}
Yi=rnorm(samples[i],mean=mean2[i],sd=sd2[i])
Yi
```

```
 [1]  1.721424177  0.001215442  0.489241586  0.389589915  1.625455889  0.942516074 -0.012183259  1.763022328  0.058442008  2.263325163
[11]  1.645759761  2.282953398  1.137833686  1.538180919  1.269910588 -0.366989517  0.545477424  0.691300497  2.245524538  1.481471256
[21]  2.079537332 -1.051116017  0.865048150  0.701118608  0.783293426  1.473039444  0.082792131  2.435900921  0.234837782  0.063828730
[31] -0.718595227  1.954027300  0.166640109  1.814601425  0.623256658  0.156167747  0.434303187  0.343161014  2.476282125  0.063546857
[41]  1.407386857  1.008820427  0.768697047  0.605979941  0.517703535 -0.209257065  2.460368867 -1.684365692  1.610639086  2.482380850
[51]  1.074466283  2.956813120  0.308135605 -0.140721012  2.130594436  2.620999273  2.321643950  0.982339501  0.437411635  2.188888881
[61]  0.733563724  0.084380339  1.558476349 -0.290045793  1.462970499  2.433043633  1.582876395  2.962657663 -0.270585566  2.831575468
[71]  2.728462698  1.156412932  0.512336724  2.618964187  2.285221310  0.138185689  1.934295029  0.805056019  0.290891059 -0.288446413
[81]  0.571824331  3.732927692  0.760896395  0.758473997  0.483385366  2.234408710  2.863538957  1.895074794  0.745957584  2.444428161
[91]  1.067999877  0.608615271  1.244377908  1.650363864  1.865277126  0.829624340  1.428517056  2.333920577  0.766947787  0.178029118
```

```{r, echo = FALSE}
data_i=cbind(Yi,i)
data_i
```

```
                Yi i
  [1,]  1.721424177 3
  [2,]  0.001215442 3
  [3,]  0.489241586 3
  [4,]  0.389589915 3
  [5,]  1.625455889 3
  [6,]  0.942516074 3
  [7,] -0.012183259 3
  [8,]  1.763022328 3
  [9,]  0.058442008 3
 [10,]  2.263325163 3
 [11,]  1.645759761 3
 [12,]  2.282953398 3
 [13,]  1.137833686 3
 [14,]  1.538180919 3
 [15,]  1.269910588 3
 [16,] -0.366989517 3
 [17,]  0.545477424 3
 [18,]  0.691300497 3
 [19,]  2.245524538 3
 [20,]  1.481471256 3
 [21,]  2.079537332 3
 [22,] -1.051116017 3
 [23,]  0.865048150 3
 [24,]  0.701118608 3
 [25,]  0.783293426 3
 [26,]  1.473039444 3
 [27,]  0.082792131 3
 [28,]  2.435900921 3
 [29,]  0.234837782 3
 [30,]  0.063828730 3
 [31,] -0.718595227 3
 [32,]  1.954027300 3
 [33,]  0.166640109 3
 [34,]  1.814601425 3
 [35,]  0.623256658 3
 [36,]  0.156167747 3
 [37,]  0.434303187 3
 [38,]  0.343161014 3
 [39,]  2.476282125 3
 [40,]  0.063546857 3
 [41,]  1.407386857 3
 [42,]  1.008820427 3
 [43,]  0.768697047 3
 [44,]  0.605979941 3
 [45,]  0.517703535 3
 [46,] -0.209257065 3
 [47,]  2.460368867 3
 [48,] -1.684365692 3
 [49,]  1.610639086 3
 [50,]  2.482380850 3
 [51,]  1.074466283 3
 [52,]  2.956813120 3
 [53,]  0.308135605 3
 [54,] -0.140721012 3
 [55,]  2.130594436 3
 [56,]  2.620999273 3
 [57,]  2.321643950 3

 [58,]  0.982339501 3
 [59,]  0.437411635 3
 [60,]  2.188888881 3
 [61,]  0.733563724 3
 [62,]  0.084380339 3
 [63,]  1.558476349 3
 [64,] -0.290045793 3
 [65,]  1.462970499 3
 [66,]  2.433043633 3
 [67,]  1.582876395 3
 [68,]  2.962657663 3
 [69,] -0.270585566 3
 [70,]  2.831575468 3
 [71,]  2.728462698 3
 [72,]  1.156412932 3
 [73,]  0.512336724 3
 [74,]  2.618964187 3
 [75,]  2.285221310 3
 [76,]  0.138185689 3
 [77,]  1.934295029 3
 [78,]  0.805056019 3
 [79,]  0.290891059 3
 [80,] -0.288446413 3
 [81,]  0.571824331 3
 [82,]  3.732927692 3
 [83,]  0.760896395 3
 [84,]  0.758473997 3
 [85,]  0.483385366 3
 [86,]  2.234408710 3
 [87,]  2.863538957 3
 [88,]  1.895074794 3
 [89,]  0.745957584 3
 [90,]  2.444428161 3
 [91,]  1.067999877 3
 [92,]  0.608615271 3
 [93,]  1.244377908 3
 [94,]  1.650363864 3
 [95,]  1.865277126 3
 [96,]  0.829624340 3
 [97,]  1.428517056 3
 [98,]  2.333920577 3
 [99,]  0.766947787 3
[100,]  0.178029118 3
```

```{r, echo = FALSE}
fit2=aov(y~level,data=mydata)
anova(fit2)
```

```
Analysis of Variance Table

Response: y
           Df  Sum Sq Mean Sq F value Pr(>F)
level       2   2.488  1.2439  1.0967 0.3365
Residuals 157 178.084  1.1343
```

```{r, echo = FALSE}
result2=anova(fit2)
result2$`Pr(>F)`[1]
```

```
[1] 0.3365316
```

- Extracting only the p-value, we get 0.3365316.

```{r, echo = FALSE}
# B number of replicates
B=10000
# save all p-values in the vector pvalues2
pvalues2=numeric(B)
for(b in 1:B){
  K=3
  # mean2=(mu1,mu2,mu3)
  mean2=c(1,1,1)
  # sd2=(sigma1,sigma2,sigma3)
  sd2=c(1,1,1)
  # samples=(n1,n2,n3)
  samples=c(10,50,100)

  # initialize mydata
  mydata=data.frame()

  for(i in 1:K){
    Yi=mean2[i]+rnorm(samples[i],mean=0,sd=sd2[i])
```

```
  data_i=cbind(Yi,i)
  mydata=rbind(mydata,data_i)
 }
 colnames(mydata)=c('y','level')
 # convert to factor variable
 mydata$level=as.factor(mydata$level)
 # fit anova model
 fit2=aov(y~level,data=mydata)

 # extract p-value
 result2=anova(fit2)
 pvalues2[b]=result2$`Pr(>F)`[1]
}
```

- The ratio that we reject the null hypothesis:

```{r, echo = FALSE}
sum(pvalues2<0.05)/B
```

```
[1] 0.0506
```

- Here we can see that the ratio is roughly above the significant level of 0.05, which means that the F-test doesn't work under this setting. Therefore, the assumption is violated and the results are inaccurate.

- Now, what if the means are different? We simulate the data such that μ_i=mean3[i], σ_i=sd3[i], n_i=samples[i] (means are different)

```{r, echo = FALSE}
K=3
# mean3=(mu1,mu2,mu3)
mean3=c(1,4,5)
# sd3=(sigma1,sigma2,sigma3)
sd3=c(1,1,1)
# samples=(n1,n2,n3)
samples=c(10,10,10)

# initialize mydata
mydata=data.frame()
```

```
for(i in 1:K){
  Yi=rnorm(samples[i],mean=mean3[i],sd=sd3[i])
  data_i=cbind(Yi,i)
  mydata=rbind(mydata,data_i)
}
colnames(mydata)=c('y','level')
# convert to factor variable
mydata$level=as.factor(mydata$level)
mydata
```

Description: df [30 × 2]

y <dbl>	level <fctr>
-0.6384183	1
2.4364748	1
-1.0645276	1
1.1709280	1
1.9661512	1
-0.3838821	1
0.9516741	1
1.6770428	1
2.2008458	1
3.0347477	1

1-10 of 30 rows Previous 1 2 3 Next

```{r, echo = FALSE}
Yi=rnorm(samples[i],mean=mean3[i],sd=sd3[i])
Yi
```

```
 [1] 5.331848 5.301799 3.590345 5.742433 4.378606 4.974927 1.966111 3.303185 5.992292 4.433944
```

```{r, echo = FALSE}
data_i=cbind(Yi,i)
data_i
```

```
             Yi i
 [1,] 5.331848 3
 [2,] 5.301799 3
 [3,] 3.590345 3
 [4,] 5.742433 3
 [5,] 4.378606 3
 [6,] 4.974927 3
 [7,] 1.966111 3
 [8,] 3.303185 3
 [9,] 5.992292 3
[10,] 4.433944 3
```

```{r, echo = FALSE}
fit3=aov(y~level,data=mydata)
anova(fit3)
```

```
Analysis of Variance Table

Response: y
           Df Sum Sq Mean Sq F value    Pr(>F)
level       2 92.515  46.258   32.89 5.791e-08 ***
Residuals 27 37.974   1.406
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```{r, echo = FALSE}
result3=anova(fit3)
result3$`Pr(>F)`[1]
```

```
[1] 5.791424e-08
```

- Extracting only the p-value, we get 5.791424e-08.

```{r, echo = FALSE}
# B number of replicates
B=10000
# save all p-values in the vector pvalues2
pvalues3=numeric(B)
for(b in 1:B){
  K=3
  # mean3=(mu1,mu2,mu3)
  mean3=c(1,4,5)
  # sd3=(sigma1,sigma2,sigma3)
  sd3=c(1,1,1)
  # samples=(n1,n2,n3)
  samples=c(100,1000,10000)

  # initialize mydata
  mydata=data.frame()
```

```
  for(i in 1:K){
    Yi=mean3[i]+rnorm(samples[i],mean=0,sd=sd3[i])
    data_i=cbind(Yi,i)
    mydata=rbind(mydata,data_i)
  }
  colnames(mydata)=c('y','level')
  # convert to factor variable
  mydata$level=as.factor(mydata$level)
  # fit anova model
  fit3=aov(y~level,data=mydata)

  # extract p-value
  result3=anova(fit3)
  pvalues3[b]=result3$`Pr(>F)`[1]
}
```

- The ratio that we reject the null hypothesis:

```{r, echo = FALSE}
sum(pvalues3<0.05)/B
```

```
[1] 1
```

- Here we can see that the ratio is above the significant level of 0.05, which means that the F-test doesn't work under this setting. Therefore, the assumption is violated and the results are inaccurate.

What if the equal variance assumption is violated? That is, members of {(σ_1)^2 ,(σ_2)^2 ,..,(σ_k)^2 } are not all equal.

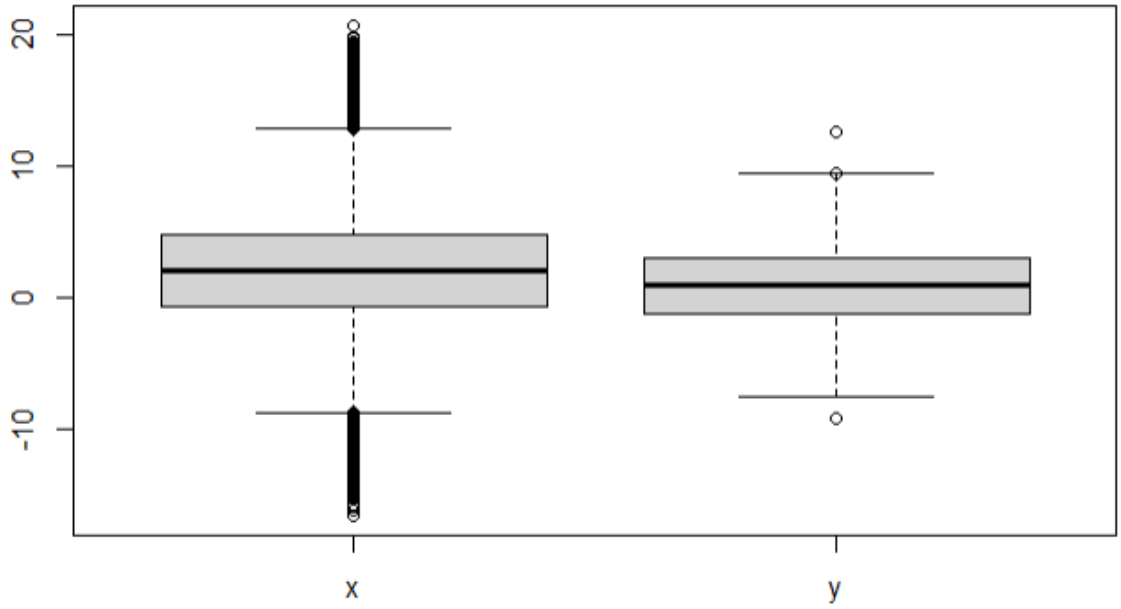
- In order to find the unequal sample sizes, we need to find the unequal variances between the samples which we will use t-test.
- By using t-test we are able to determine whether the samples have equal variances or not. One of the assumptions made in a t-test is when the two samples do not have equal variances, it is a real issue to identify the significant differences of the samples.

```{r, echo = FALSE}
t.test(x, y, var.equal = TRUE)
```

```
t_test_1 = boxplot(x,y, names=c("x","y"))
t_test_1
t.test(x, y, var.equal = FALSE)
```

```
        Two Sample t-test

data:  x and y
t = 9.1359, df = 1000998, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.9088703 1.4053495
sample estimates:
mean of x mean of y
1.9999044 0.8427945

$stats
           [,1]       [,2]
[1,] -8.7944394 -7.6431929
[2,] -0.7001064 -1.2753978
[3,]  2.0005483  0.8474057
[4,]  4.6961998  2.9887300
[5,] 12.7904982  9.3792565

$n
[1] 1e+06 1e+03

$conf
         [,1]      [,2]
[1,] 1.992022 0.6343529
[2,] 2.009074 1.0604585

$out
```

```
        Welch Two Sample t-test

data:  x and y
t = 12.146, df = 1002.5, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.9701624 1.3440574
sample estimates:
mean of x mean of y
1.9999044 0.8427945
```



- Based on the boxplot, we can see that the variances of samples x and y are nearly the same, but the p-values are significantly different. Now we compare the p-value with the significant level of 0.05. We will also use the t-test and the Welch's t-test to further prove our statements. Since the p-value of the t-test (when the variances of two samples are the same) is less than the significant level of 0.05, we reject the null hypothesis, which means that the variance of x is equal to the variance of y. However, the p-value of the Welch's t-test is greater than the significant level of 0.05, we fail to reject the null hypothesis, which means that the variance of x is not equal to the variance of y. Thus, we conclude that only the Welch's t-test is able to detect the significant difference because the two variances are unequal. The t-test might give out inaccurate information.

- Next, we simulate the data such that μ_i=mean3[i], σ_i=sd3[i], n_i=samples[i] (the variances are not all equal)

```{r, echo = FALSE}
K=3
# mean4=(mu1,mu2,mu3)
mean4=c(1,1,1)
# sd4=(sigma1,sigma2,sigma3)
sd4=c(1,2,3)
# samples=(n1,n2,n3)
```

```
samples=c(10,10,10)

# initialize mydata
mydata=data.frame()

for(i in 1:K){
  Yi=rnorm(samples[i],mean=mean4[i],sd=sd4[i])
  data_i=cbind(Yi,i)
  mydata=rbind(mydata,data_i)
}
colnames(mydata)=c('y','level')
# convert to factor variable
mydata$level=as.factor(mydata$level)
mydata
```

Description: df [30 × 2]

y <dbl>	level <fctr>
2.3106359	1
-0.2881330	1
-0.3752931	1
0.4171258	1
2.1135489	1
1.7945325	1
1.2555977	1
1.3169368	1
0.8702643	1
-1.0282300	1

1-10 of 30 rows Previous 1 2 3 Next

```{r, echo = FALSE}
Yi=rnorm(samples[i],mean=mean4[i],sd=sd4[i])
Yi
```

```
[1] -3.4217443  1.7699466  3.7144962 -2.7022061 -5.1513586 -6.3237604 -3.2460668 -1.5907242  0.1004859 -1.0260492
```

```{r, echo = FALSE}
data_i=cbind(Yi,i)
data_i
```

```
              Yi i
 [1,] -3.4217443 3
 [2,]  1.7699466 3
 [3,]  3.7144962 3
 [4,] -2.7022061 3
 [5,] -5.1513586 3
 [6,] -6.3237604 3
 [7,] -3.2460668 3
 [8,] -1.5907242 3
 [9,]  0.1004859 3
[10,] -1.0260492 3
```

```{r, echo = FALSE}
fit4=aov(y~level,data=mydata)
anova(fit4)
```

```
Analysis of Variance Table

Response: y
           Df  Sum Sq Mean Sq F value Pr(>F)
level       2   4.047  2.0235  0.3715 0.6932
Residuals 27 147.061  5.4467
```

```{r, echo = FALSE}
result4=anova(fit4)
result4$`Pr(>F)`[1]
```

```
[1] 0.6931681
```

- Extracting only the p-value, we get 0.6931681.

```{r, echo = FALSE}
# B number of replicates
B=10000
# save all p-values in the vector pvalues3
pvalues4=numeric(B)
for(b in 1:B){
  K=3
  # mean4=(mu1,mu2,mu3)
  mean4=c(1,1,1)
  # sd4=(sigma1,sigma2,sigma3)
  sd4=c(1,2,3)
```

```
  # samples=(n1,n2,n3)
  samples=c(10,10,10)

  # initialize mydata
  mydata=data.frame()

  for(i in 1:K){
    Yi=mean4[i]+rnorm(samples[i],mean=0,sd=sd4[i])
    data_i=cbind(Yi,i)
    mydata=rbind(mydata,data_i)
  }
  colnames(mydata)=c('y','level')
  # convert to factor variable
  mydata$level=as.factor(mydata$level)
  # fit anova model
  fit4=aov(y~level,data=mydata)

  # extract p-value
  result4=anova(fit4)
  pvalues4[b]=result4$`Pr(>F)`[1]
}
```

- After repeating the process 10000 times, the ratio that we reject the null hypothesis:

```{r, echo = FALSE}
sum(pvalues4<0.05)/B
```

```
  [1] 0.0627
```

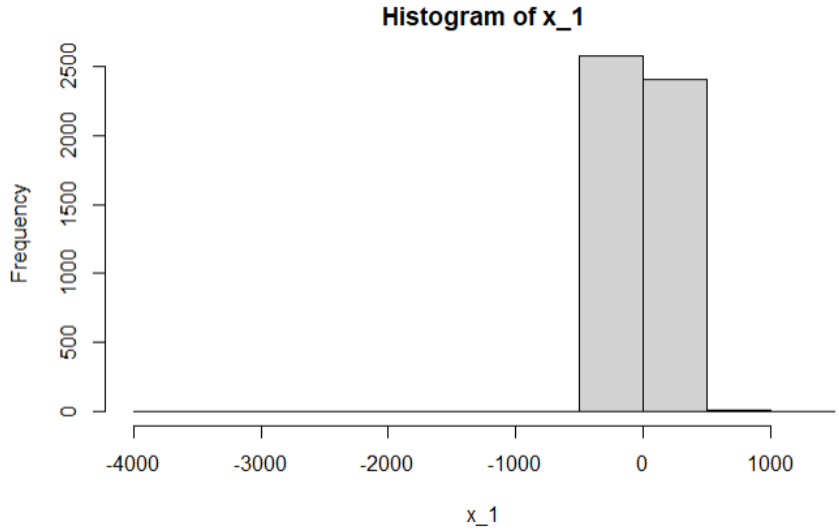
- Since the ratio is roughly above the significance level of 0.05, the F-test doesn't work under this setting. Therefore, the assumption is violated. It also means the results of the tests are unaccurate.

What if the Normality assumption is violated?

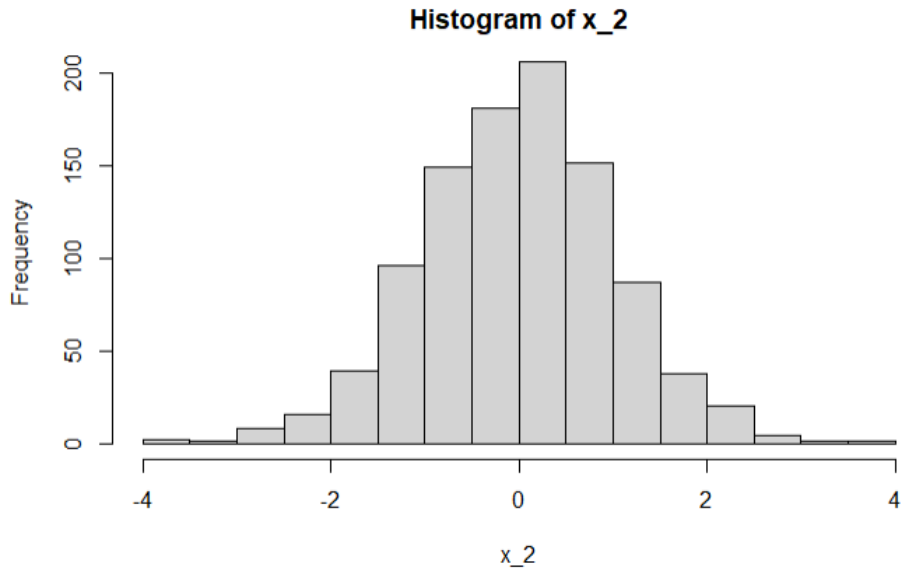
```{r, echo = FALSE}
x_1 = rt(5000, 1)
x_2 = rt(1000, 100)
```

- If the histogram for a data set is roughly bell-shaped, then it's likely that the data is normally distributed.

```{r, echo = FALSE}
hist(x_1)
```




```{r, echo = FALSE}
hist(x_2)
```



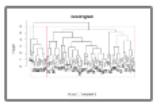
```{r, echo = FALSE}
shapiro.test(x_1)
shapiro.test(x_2)
```

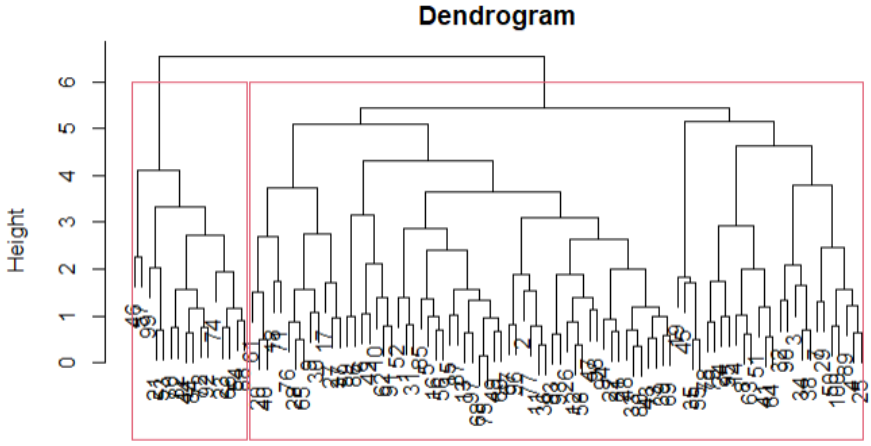
```
        Shapiro-Wilk normality test

data:  x_1
W = 0.047904, p-value < 2.2e-16


        Shapiro-Wilk normality test

data:  x_2
W = 0.99853, p-value = 0.5741
```

- Based on the histogram of each samples, we can see that sample x_2 is roughly bell-shaped, which mean that it is normally distributed and it doesn't violate the Normality assumption. However, in sample x_1, the histogram is not bell-shaped, which means that it is not normally distributed and it violates the Normality assumption.
- Based on the shapiro test, we can see that the p-value of x_1 is less than the significance level of 0.05 which means that the data is not normal and it violates the Normality assumption, but the p-value of x_2 is greater than 0.05 meaning that the data is normal and the Normality assumption is not violated.

What if you want to discover the community structures among the K samples?

```{r, echo = FALSE}
x_1=rnorm(100,0,1)
print(c(mean(x_1),var(x_1)))
x_2=rnorm(100,0,1)
print(c(mean(x_2),var(x_2)))
x_3=rnorm(100,0,1)
print(c(mean(x_3),var(x_3)))
x_4=rnorm(100,4,1)
print(c(mean(x_4),var(x_4)))

samples = data.frame(x_1,x_2,x_3,x_4)

clusters <- hclust(dist(samples))
plot(clusters,xlab='',main='Dendrogram')

# cut off the tree at the desired number of clusters using cutree.
clusterCut <- cutree(clusters, 1)
# plot the tree
rect.hclust(clusters, k=2)
```





- Based on the dendrogram tree, we can see that the community structures among samples x_1, x_2, x_3 are in the same category (the same cluster) as the means and variances of the samples are nearly the same. However, when we change the mean of the sample x_4, the tree indicates that its community structure is separated into its own cluster which datas are also in a different box.

Part 2: Exploring beyond One-way ANOVA with Kaggle data.

```{r, echo = FALSE}
# combine several binary HeartDiseaseorAttack, HighBP, HighChol, Stroke, Smoker
# into one categorical variable
BMI <- data.frame(BMI = heart_disease$BMI)
HeartDiseaseorAttack <- data.frame(HeartDiseaseorAttack = heart_disease$HeartDiseaseorAttack)
HighBP <- data.frame(HighBP = heart_disease$HighBP)
HighChol <- data.frame(HighChol = heart_disease$HighChol)
Stroke <- data.frame(Stroke = heart_disease$Stroke)
Smoker <- data.frame(Smoker = heart_disease$Smoker)
HD = cbind(BMI, HeartDiseaseorAttack, HighBP, HighChol, Stroke, Smoker)

# as.factor(): covert to factor variable
HD$HeartDiseaseorAttack=as.factor(HD$HeartDiseaseorAttack)
HD$HighBP=as.factor(HD$HighBP)
HD$HighChol=as.factor(HD$HighChol)
HD$Stroke=as.factor(HD$Stroke)
HD$Smoker=as.factor(HD$Smoker)

# Combine specified categorical variables by
# concatenating their values into one character
# function combineCatVars(.data,vars,sep = ".")
# .data: a dataframe with the columns to be combined
#  vars: a character vector of the categorical variables to be combined
#  sep: the separator to combine the values of the variables in var
library(iNZightTools)
HD = combineCatVars(HD, vars = c('HeartDiseaseorAttack', 'HighBP', 'HighChol', 'Stroke', 'Smoker'), sep = "_")
colnames(HD)[7]  <- "Combined"
HD
# check the last column of the data combined
```

Description: df [253,680 × 7]

BMI <dbl>	HeartDiseaseorAttack <fctr>	HighBP <fctr>	HighChol <fctr>	Stroke <fctr>	Smoker <fctr>	Combined <fctr>
40	0	1	1	0	1	0_1_1_0_1
25	0	0	0	0	1	0_0_0_0_1
28	0	1	1	0	0	0_1_1_0_0
27	0	1	0	0	0	0_1_0_0_0
24	0	1	1	0	0	0_1_1_0_0
25	0	1	1	0	1	0_1_1_0_1
30	0	1	0	0	1	0_1_0_0_1
25	0	1	1	0	1	0_1_1_0_1
30	1	1	1	0	1	1_1_1_0_1
24	0	0	0	0	0	0_0_0_0_0

1-10 of 253,680 rows

Previous 1 2 3 4 5 6 ... 100 Next

```{r, echo = FALSE}
New_HD = data.frame(xtabs(~BMI + Combined ,data= HD))
New_HD
```

Description: df [2,688 × 3]

BMI <fctr>	Combined <fctr>	Freq <int>
12	0_0_0_0_0	3
13	0_0_0_0_0	6
14	0_0_0_0_0	9
15	0_0_0_0_0	29
16	0_0_0_0_0	79
17	0_0_0_0_0	202
18	0_0_0_0_0	603
19	0_0_0_0_0	1524
20	0_0_0_0_0	2496
21	0_0_0_0_0	3715

1-10 of 2,688 rows

Previous 1 2 3 4 5 6 ... 100 Next

- Creating multiple datas such that BMI is in respect to different variables of the Combined.

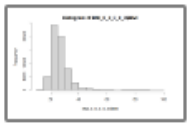
```{r, echo = FALSE}
BMI_0_0_0_0_0 = data.frame(HD[HD$Combined == '0_0_0_0_0',])


BMI_mean1 = mean(BMI_0_0_0_0_0$BMI)
BMI_variances1 = var(BMI_0_0_0_0_0$BMI)
BMI_mean1
BMI_variances1


hist(BMI_0_0_0_0_0$BMI)
```

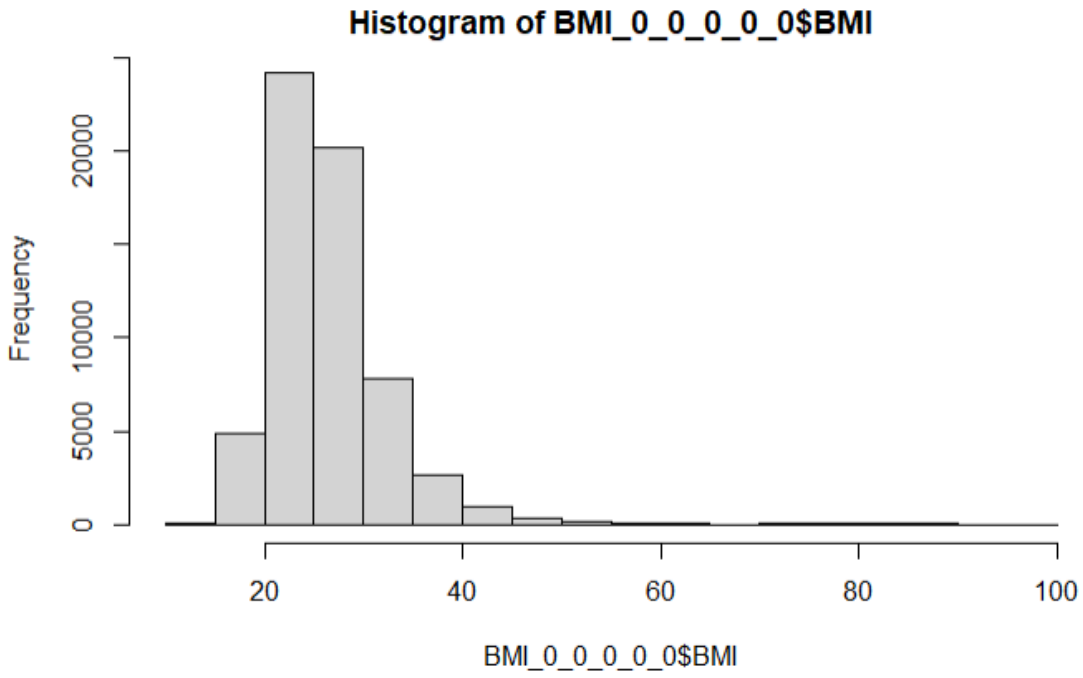
Description: df [61,489 × 7]

	BMI <dbl>	HeartDiseaseorAttack <fctr>	HighBP <fctr>	HighChol <fctr>	Stroke <fctr>	Smoker <fctr>	Combined <fctr>
10	24	0	0	0	0	0	0_0_0_0_0
19	23	0	0	0	0	0	0_0_0_0_0
26	32	0	0	0	0	0	0_0_0_0_0
33	23	0	0	0	0	0	0_0_0_0_0
38	22	0	0	0	0	0	0_0_0_0_0
41	26	0	0	0	0	0	0_0_0_0_0
47	31	0	0	0	0	0	0_0_0_0_0
54	31	0	0	0	0	0	0_0_0_0_0
56	29	0	0	0	0	0	0_0_0_0_0
57	26	0	0	0	0	0	0_0_0_0_0

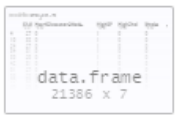
1-10 of 61,489 rows

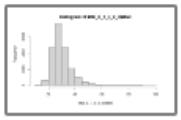
Previous 1 2 3 4 5 6 ... 100 Next

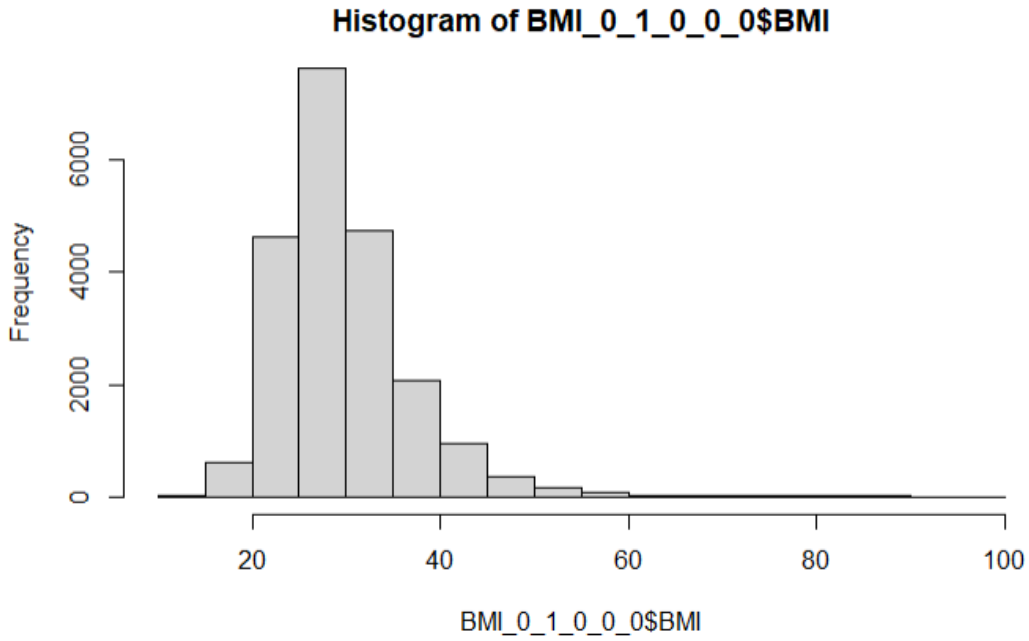




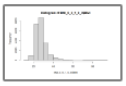
```{r, echo = FALSE}

BMI_0_1_0_0_0 = data.frame(HD[HD$Combined == '0_1_0_0_0',])

BMI_0_1_0_0_0


BMI_mean2 = mean(BMI_0_1_0_0_0$BMI)
BMI_variances2 = var(BMI_0_1_0_0_0$BMI)
BMI_mean2
BMI_variances2


hist(BMI_0_1_0_0_0$BMI)
```

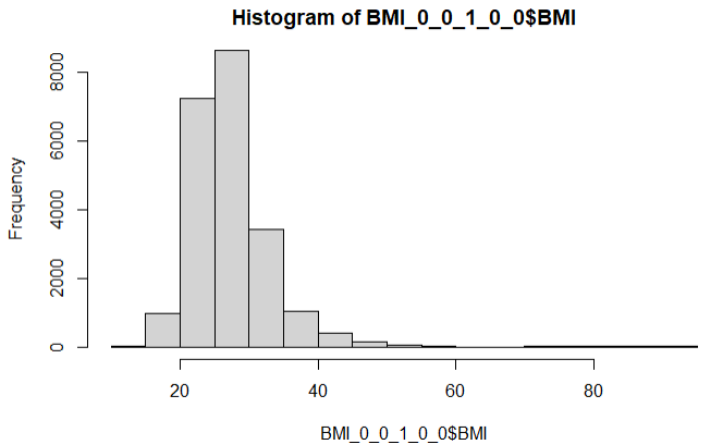




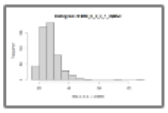


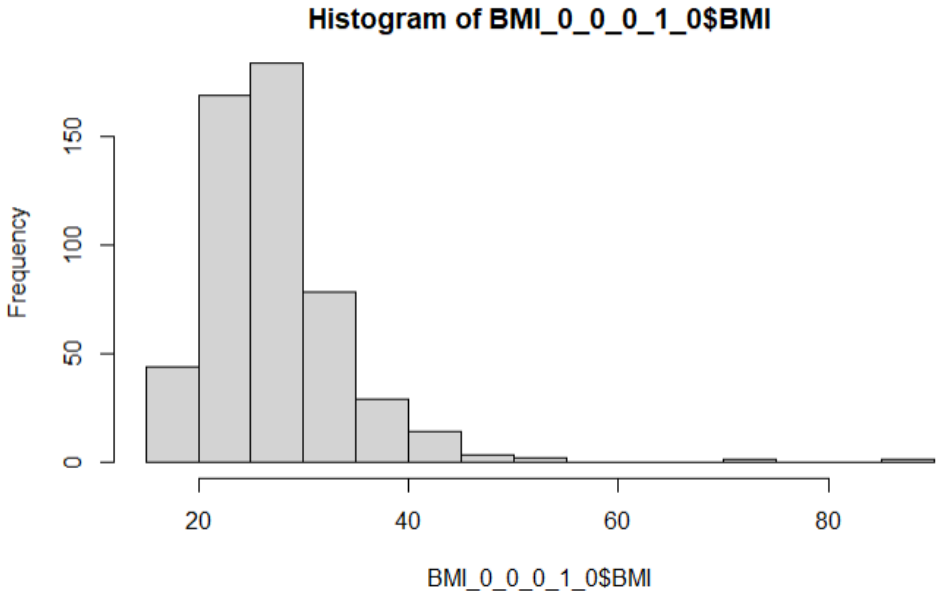
```r
BMI_0_0_1_0_0 = data.frame(HD[HD$Combined == '0_0_1_0_0',])

BMI_0_0_1_0_0


BMI_mean3 = mean(BMI_0_0_1_0_0$BMI)
BMI_variances3 = var(BMI_0_0_1_0_0$BMI)
BMI_mean3
BMI_variances3


hist(BMI_0_0_1_0_0$BMI)
```



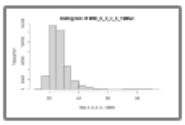


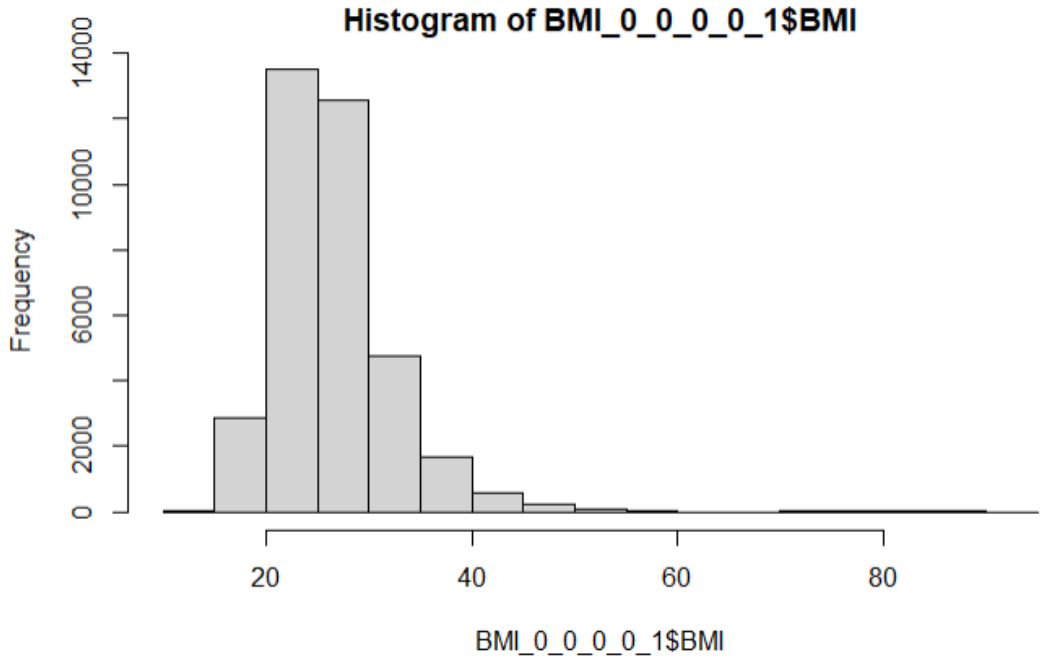
```{r, echo = FALSE}

BMI_0_0_0_1_0 = data.frame(HD[HD$Combined == '0_0_0_1_0',])

BMI_0_0_0_1_0


BMI_mean4 = mean(BMI_0_0_0_1_0$BMI)
BMI_variances4 = var(BMI_0_0_0_1_0$BMI)
BMI_mean4
BMI_variances4


hist(BMI_0_0_0_1_0$BMI)
```



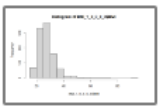


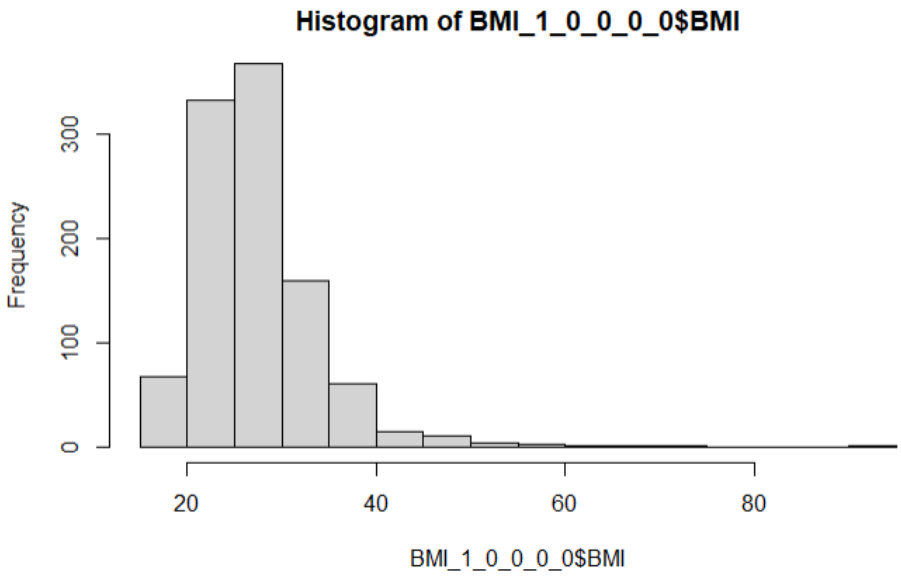
```{r, echo = FALSE}

BMI_0_0_0_0_1 = data.frame(HD[HD$Combined == '0_0_0_0_1',])

BMI_0_0_0_0_1


BMI_mean5 = mean(BMI_0_0_0_0_1$BMI)
BMI_variances5 = var(BMI_0_0_0_0_1$BMI)
BMI_mean5
BMI_variances5


hist(BMI_0_0_0_0_1$BMI)
```



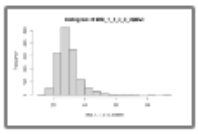


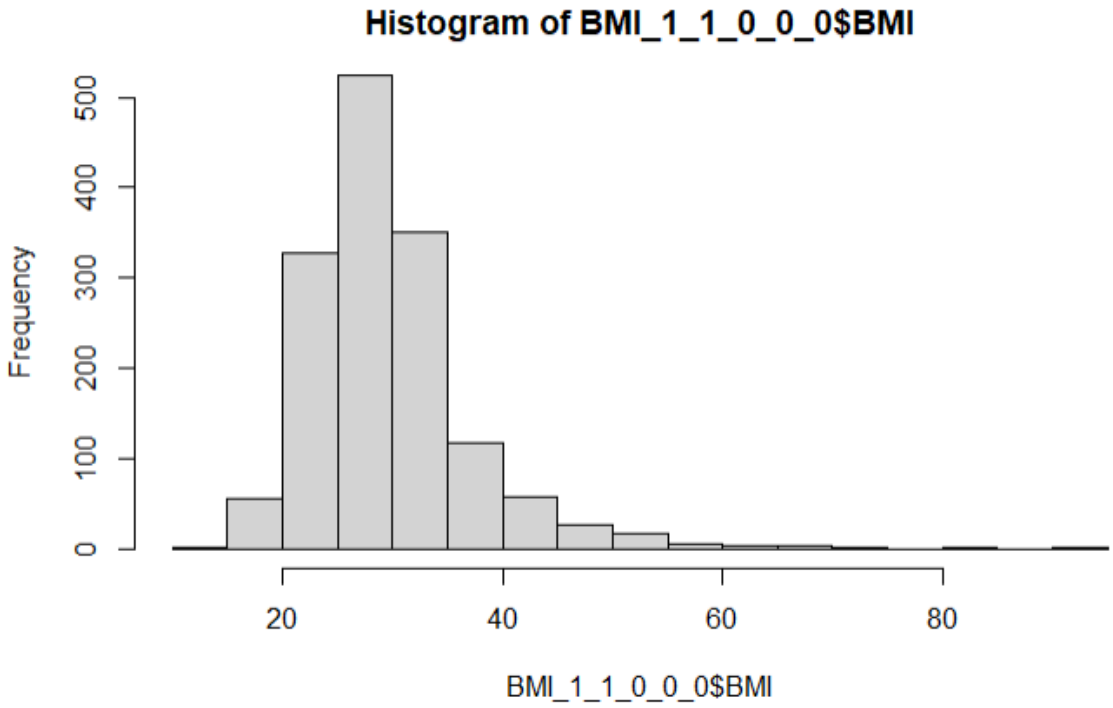
```{r, echo = FALSE}

BMI_1_0_0_0_0 = data.frame(HD[HD$Combined == '1_0_0_0_0',])

BMI_1_0_0_0_0


BMI_mean6 = mean(BMI_1_0_0_0_0$BMI)
BMI_variances6 = var(BMI_1_0_0_0_0$BMI)
BMI_mean6
BMI_variances6


hist(BMI_1_0_0_0_0$BMI)
```



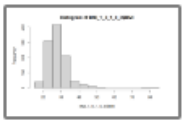


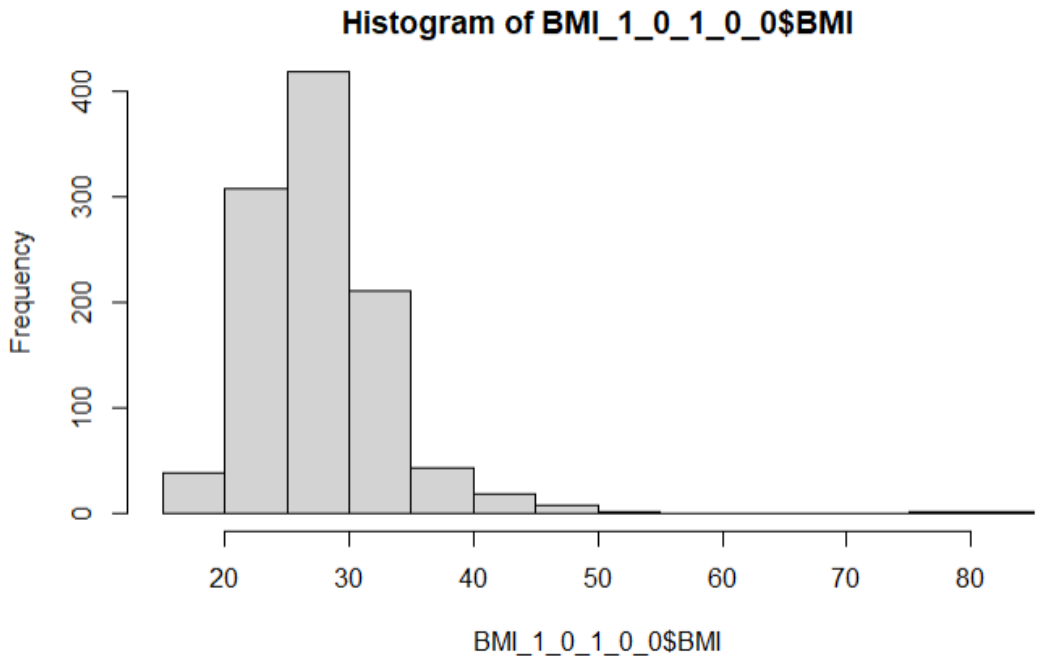
```{r, echo = FALSE}

BMI_1_1_0_0_0 = data.frame(HD[HD$Combined == '1_1_0_0_0',])

BMI_1_1_0_0_0


BMI_mean7 = mean(BMI_1_1_0_0_0$BMI)
BMI_variances7 = var(BMI_1_1_0_0_0$BMI)
BMI_mean7
BMI_variances7


hist(BMI_1_1_0_0_0$BMI)
```



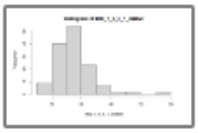


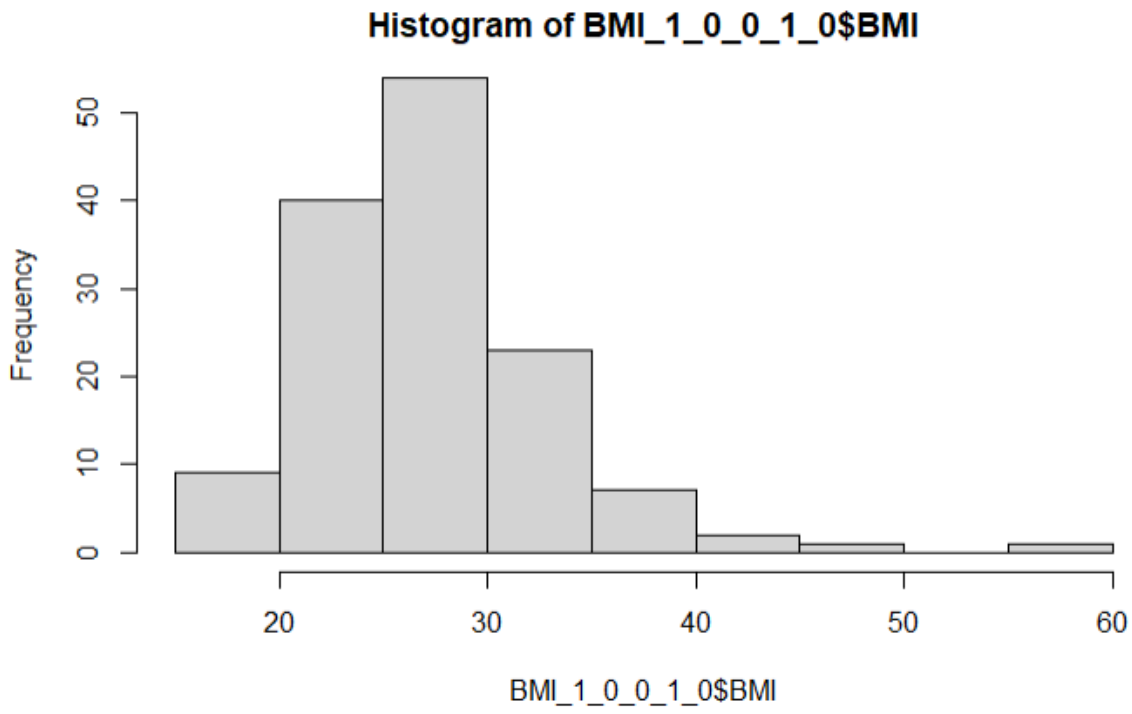
```{r, echo = FALSE}

BMI_1_0_1_0_0 = data.frame(HD[HD$Combined == '1_0_1_0_0',])

BMI_1_0_1_0_0


BMI_mean8 = mean(BMI_1_0_1_0_0$BMI)
BMI_variances8 = var(BMI_1_0_1_0_0$BMI)
BMI_mean8
BMI_variances8


hist(BMI_1_0_1_0_0$BMI)
```



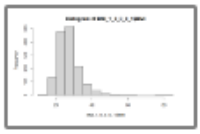


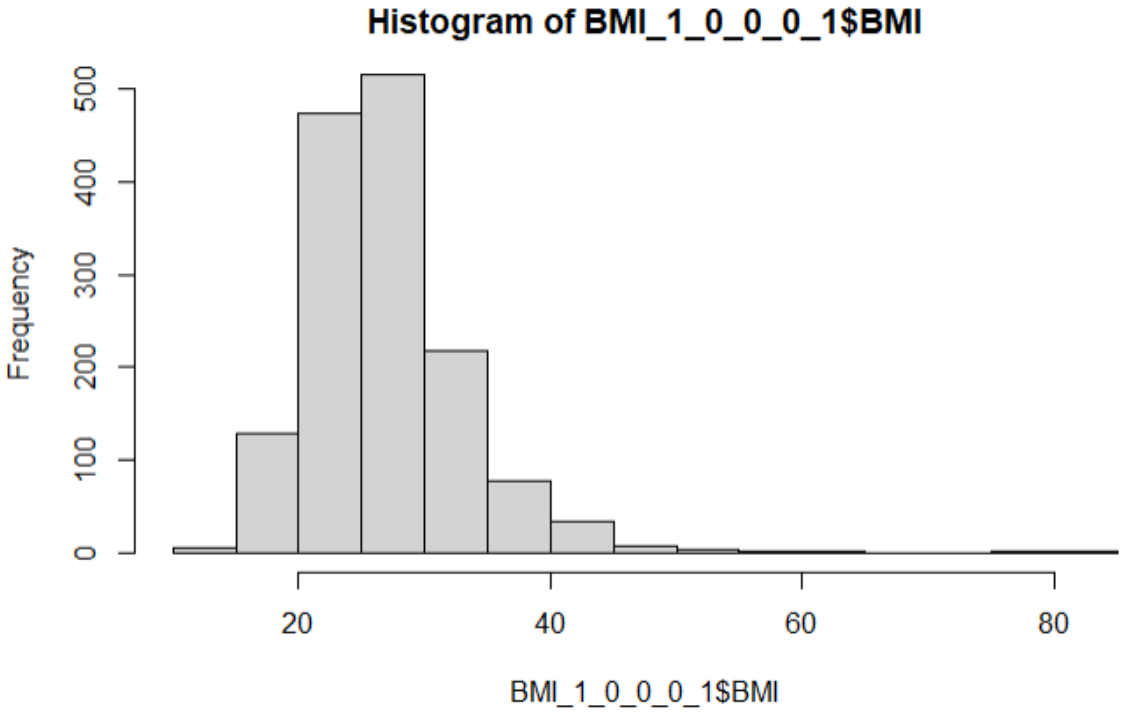
```{r, echo = FALSE}

BMI_1_0_0_1_0 = data.frame(HD[HD$Combined == '1_0_0_1_0',])

BMI_1_0_0_1_0


BMI_mean9 = mean(BMI_1_0_0_1_0$BMI)
BMI_variances9 = var(BMI_1_0_0_1_0$BMI)
BMI_mean9
BMI_variances9


hist(BMI_1_0_0_1_0$BMI)
```



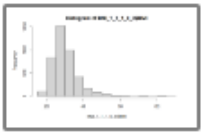


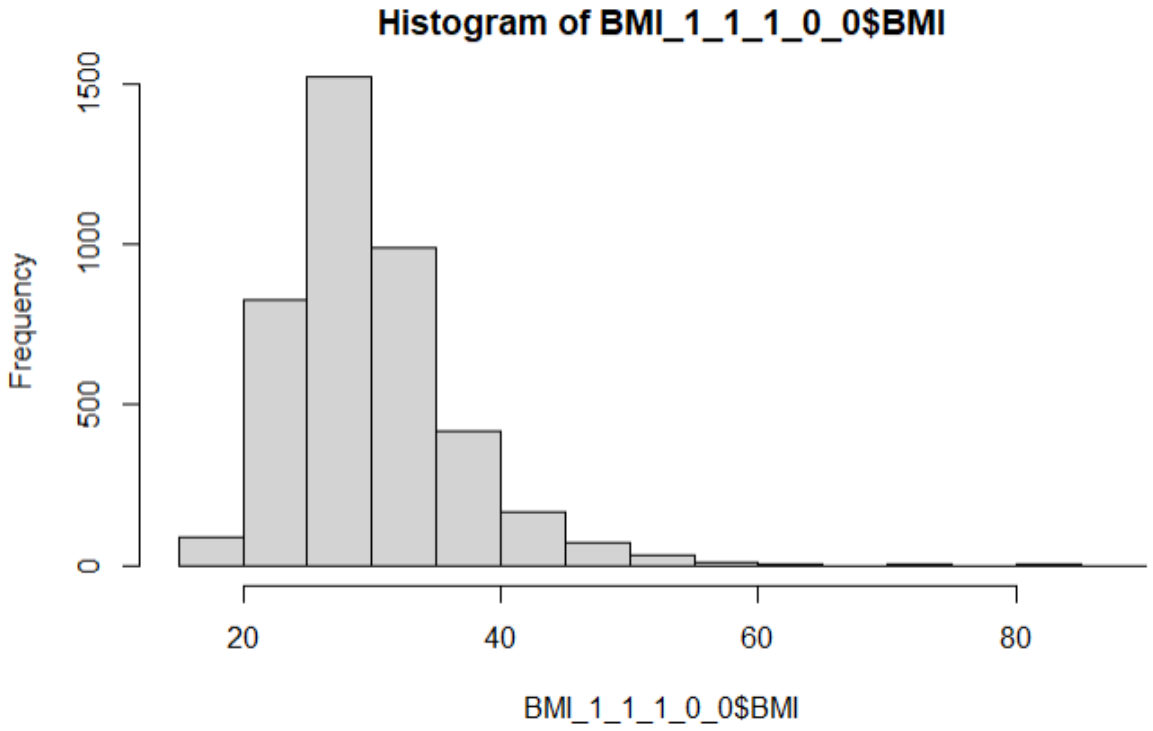
```r
BMI_1_0_0_0_1 = data.frame(HD[HD$Combined == '1_0_0_0_1',])

BMI_1_0_0_0_1


BMI_mean10 = mean(BMI_1_0_0_0_1$BMI)
BMI_variances10 = var(BMI_1_0_0_0_1$BMI)
BMI_mean10
BMI_variances10


hist(BMI_1_0_0_0_1$BMI)
```



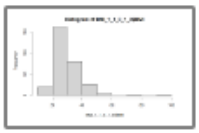


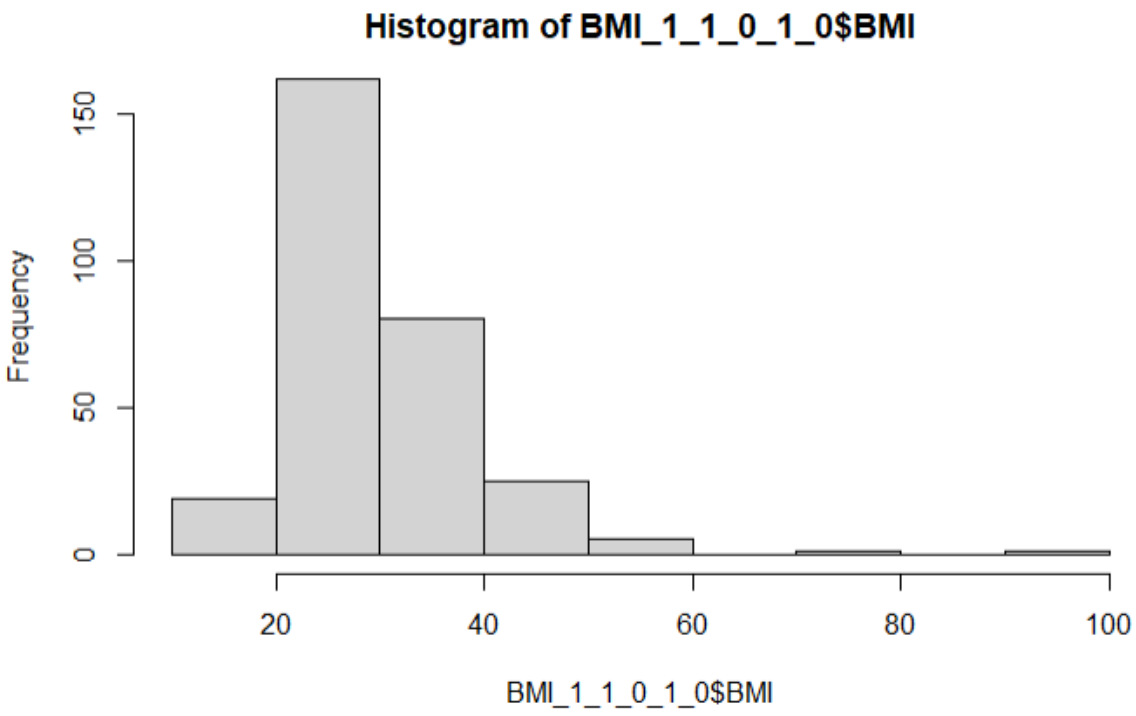
```{r, echo = FALSE}

BMI_1_1_1_0_0 = data.frame(HD[HD$Combined == '1_1_1_0_0',])

BMI_1_1_1_0_0


BMI_mean11 = mean(BMI_1_1_1_0_0$BMI)
BMI_variances11 = var(BMI_1_1_1_0_0$BMI)
BMI_mean11
BMI_variances11


hist(BMI_1_1_1_0_0$BMI)
```



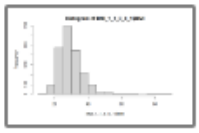


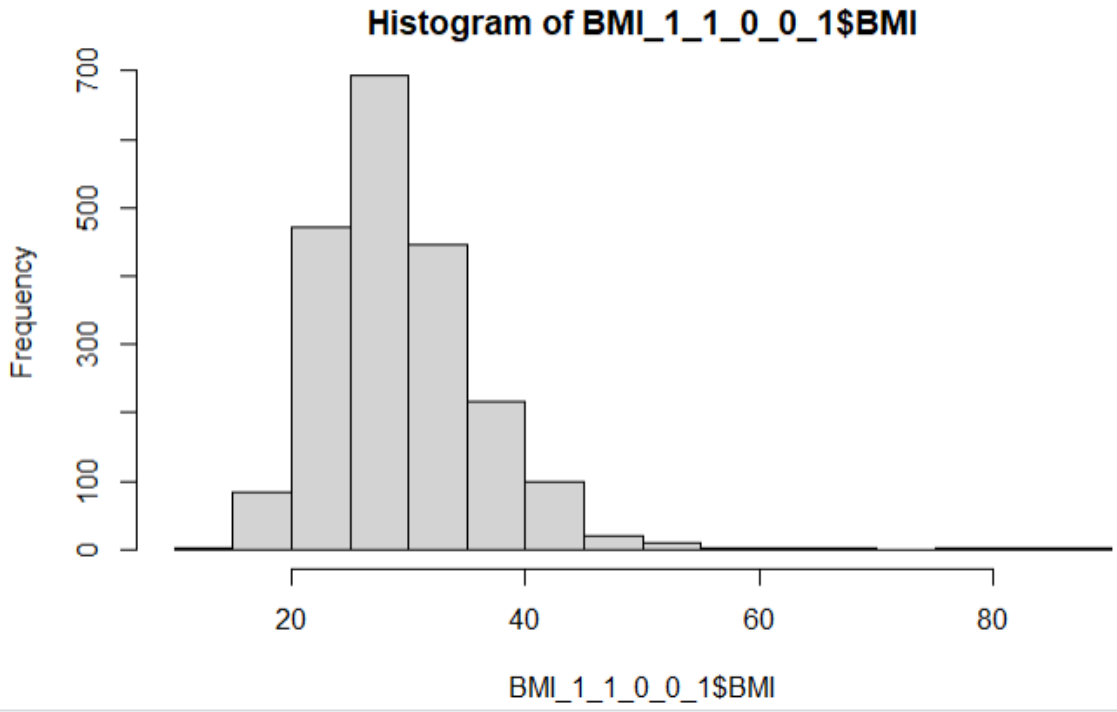
```{r, echo = FALSE}

BMI_1_1_0_1_0 = data.frame(HD[HD$Combined == '1_1_0_1_0',])

BMI_1_1_0_1_0


BMI_mean12 = mean(BMI_1_1_0_1_0$BMI)
BMI_variances12 = var(BMI_1_1_0_1_0$BMI)
BMI_mean12
BMI_variances12


hist(BMI_1_1_0_1_0$BMI)
```



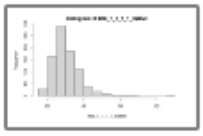


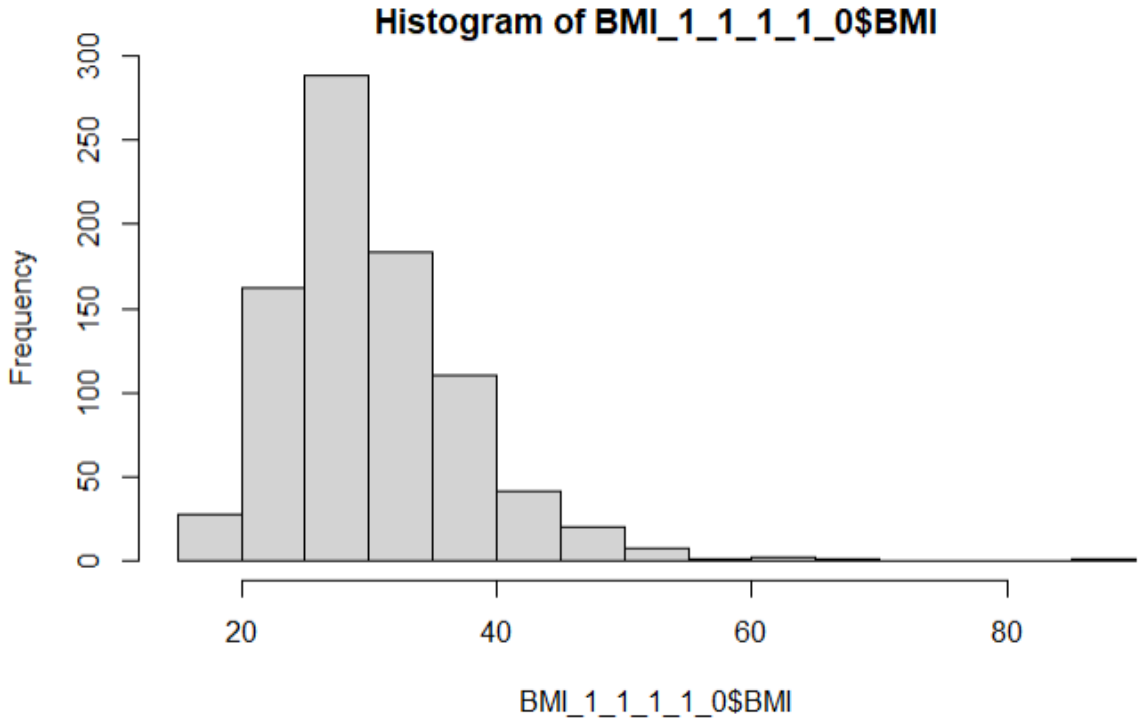
```{r, echo = FALSE}

BMI_1_1_0_0_1 = data.frame(HD[HD$Combined == '1_1_0_0_1',])

BMI_1_1_0_0_1


BMI_mean13 = mean(BMI_1_1_0_0_1$BMI)
BMI_variances13 = var(BMI_1_1_0_0_1$BMI)
BMI_mean13
BMI_variances13


hist(BMI_1_1_0_0_1$BMI)
```



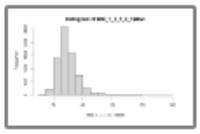


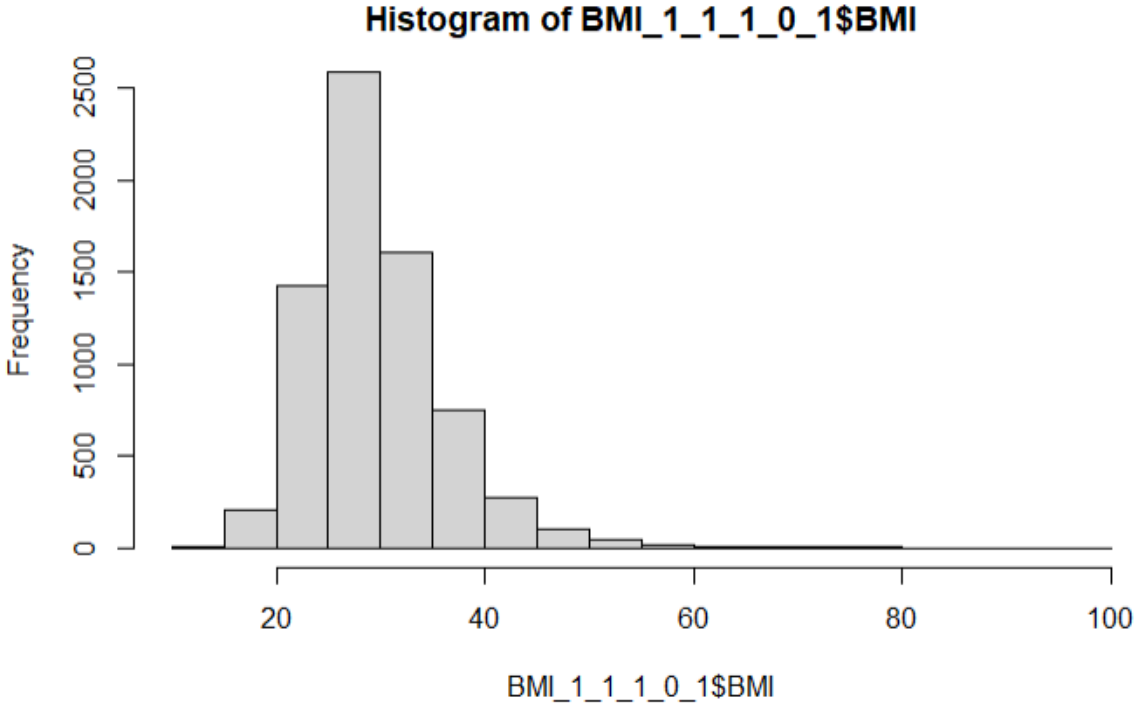
```{r, echo = FALSE}

BMI_1_1_1_1_0 = data.frame(HD[HD$Combined == '1_1_1_1_0',])

BMI_1_1_1_1_0


BMI_mean14 = mean(BMI_1_1_1_1_0$BMI)
BMI_variances14 = var(BMI_1_1_1_1_0$BMI)
BMI_mean14
BMI_variances14


hist(BMI_1_1_1_1_0$BMI)
```



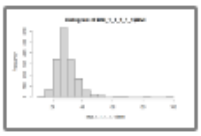


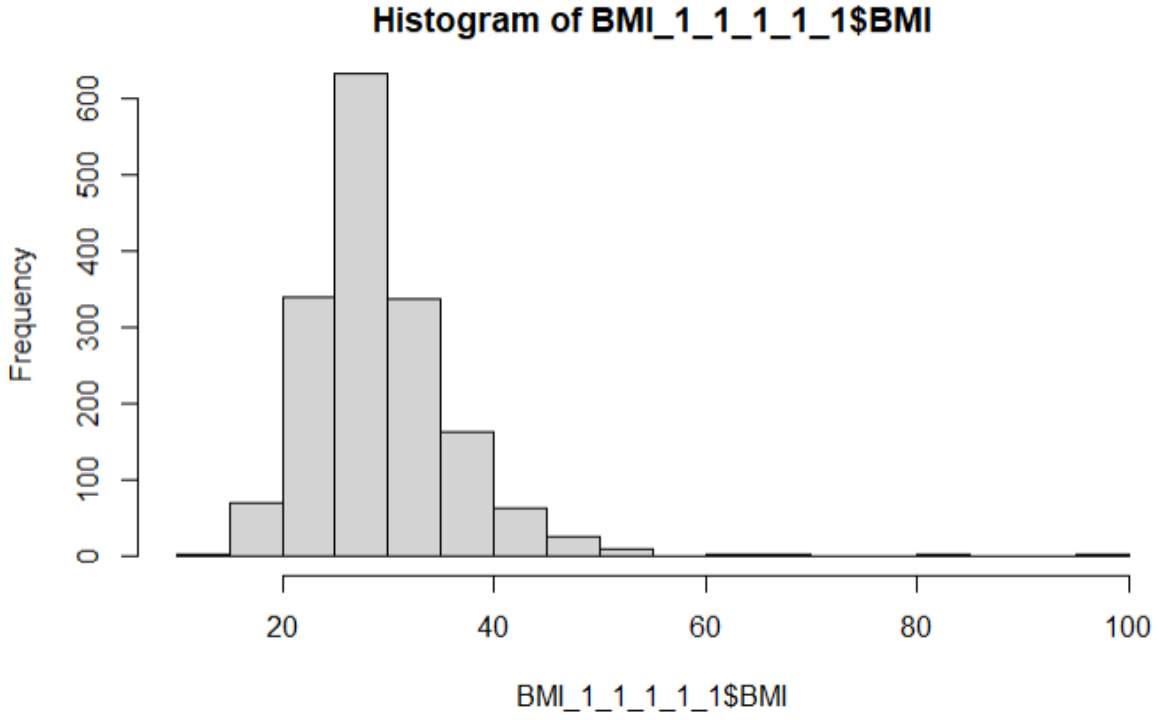
```{r, echo = FALSE}

BMI_1_1_1_0_1 = data.frame(HD[HD$Combined == '1_1_1_0_1',])

BMI_1_1_1_0_1


BMI_mean15 = mean(BMI_1_1_1_0_1$BMI)
BMI_variances15 = var(BMI_1_1_1_0_1$BMI)
BMI_mean15
BMI_variances15


hist(BMI_1_1_1_0_1$BMI)
```



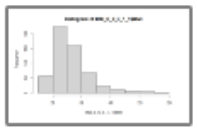


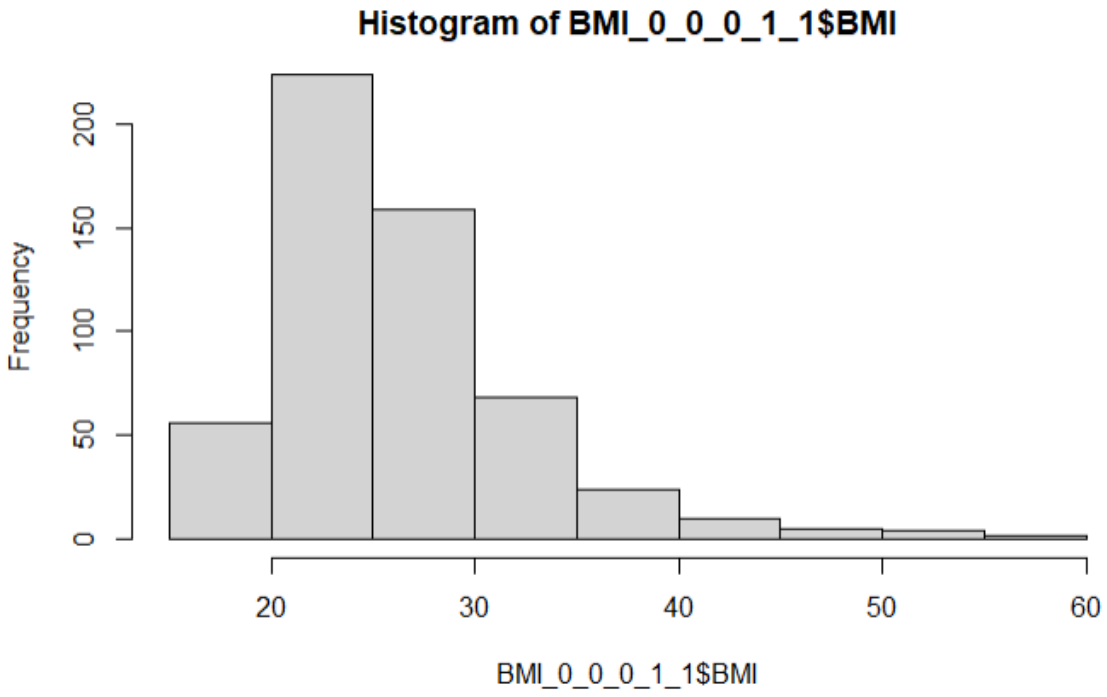
```{r, echo = FALSE}

BMI_1_1_1_1_1 = data.frame(HD[HD$Combined == '1_1_1_1_1',])

BMI_1_1_1_1_1


BMI_mean16 = mean(BMI_1_1_1_1_1$BMI)
BMI_variances16 = var(BMI_1_1_1_1_1$BMI)
BMI_mean16
BMI_variances16


hist(BMI_1_1_1_1_1$BMI)
```



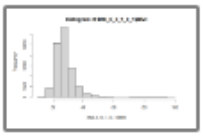


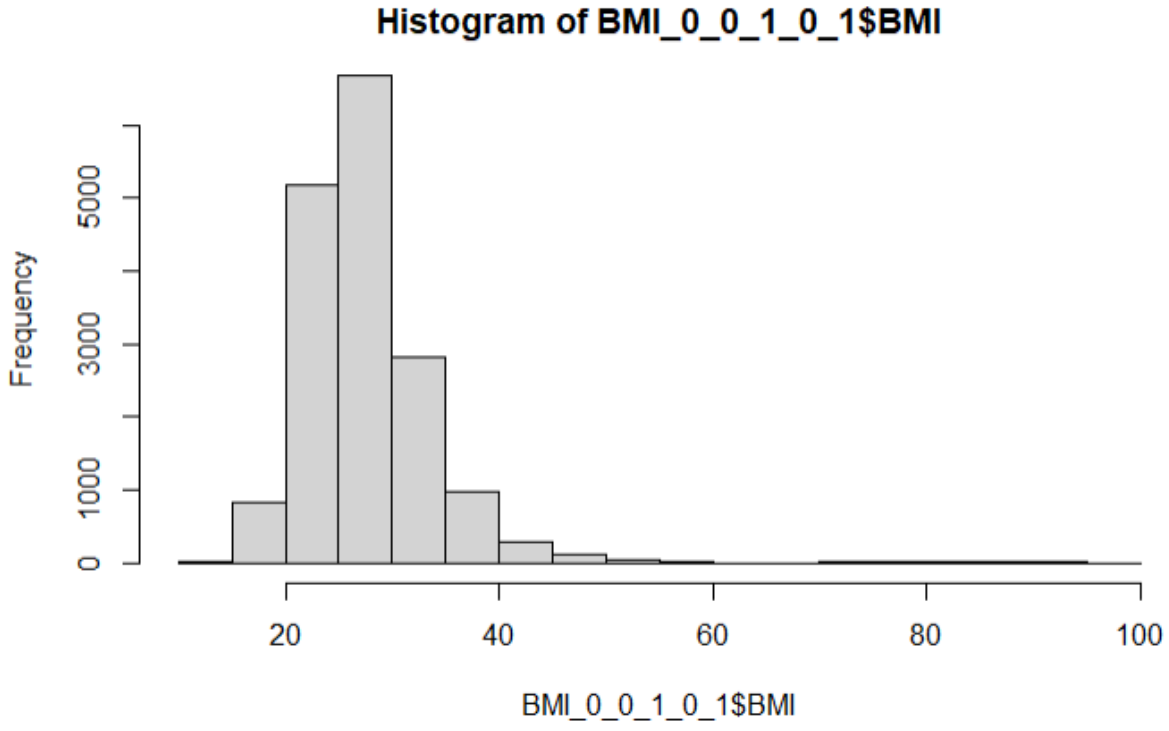
```{r, echo = FALSE}

BMI_0_0_0_1_1 = data.frame(HD[HD$Combined == '0_0_0_1_1',])

BMI_0_0_0_1_1


BMI_mean17 = mean(BMI_0_0_0_1_1$BMI)
BMI_variances17 = var(BMI_0_0_0_1_1$BMI)
BMI_mean17
BMI_variances17


hist(BMI_0_0_0_1_1$BMI)
```



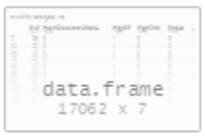


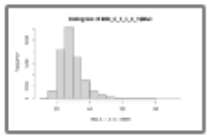
```{r, echo = FALSE}

BMI_0_0_1_0_1 = data.frame(HD[HD$Combined == '0_0_1_0_1',])

BMI_0_0_1_0_1


BMI_mean18 = mean(BMI_0_0_1_0_1$BMI)
BMI_variances18 = var(BMI_0_0_1_0_1$BMI)
BMI_mean18
BMI_variances18


hist(BMI_0_0_1_0_1$BMI)
```

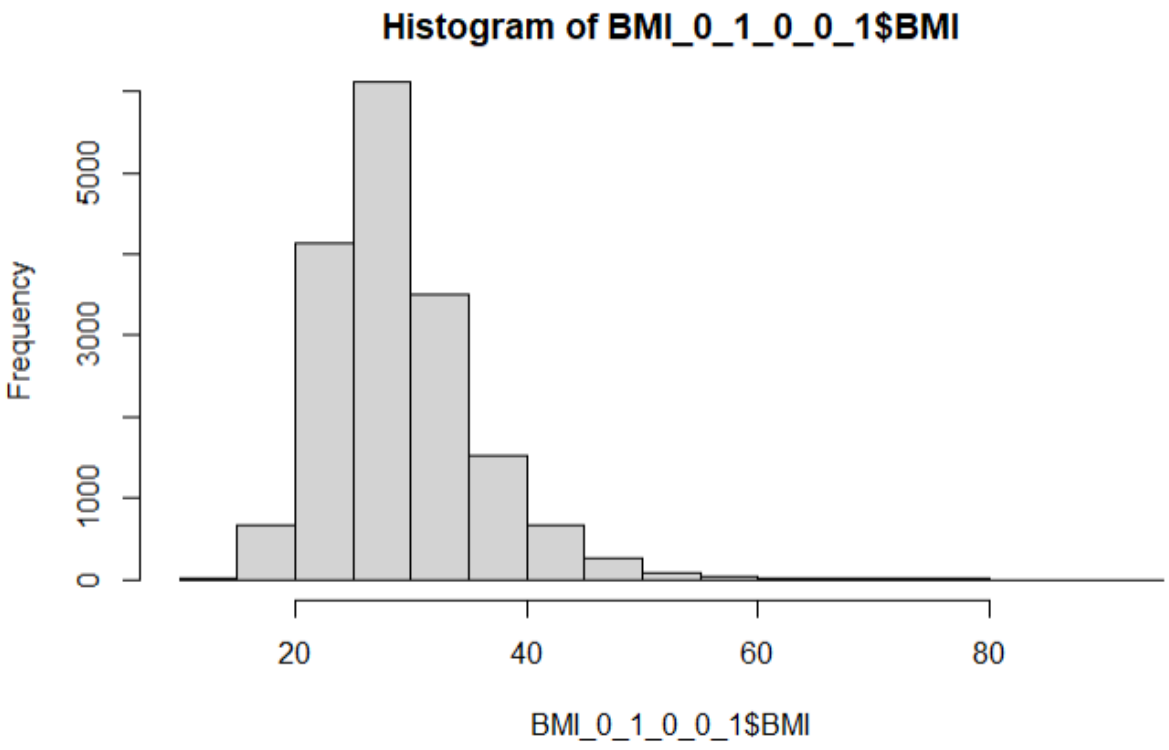




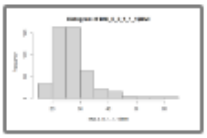
```{r, echo = FALSE}

BMI_0_1_0_0_1 = data.frame(HD[HD$Combined == '0_1_0_0_1',])

BMI_0_1_0_0_1


BMI_mean19 = mean(BMI_0_1_0_0_1$BMI)
BMI_variances19 = var(BMI_0_1_0_0_1$BMI)
BMI_mean19
BMI_variances19


hist(BMI_0_1_0_0_1$BMI)
```

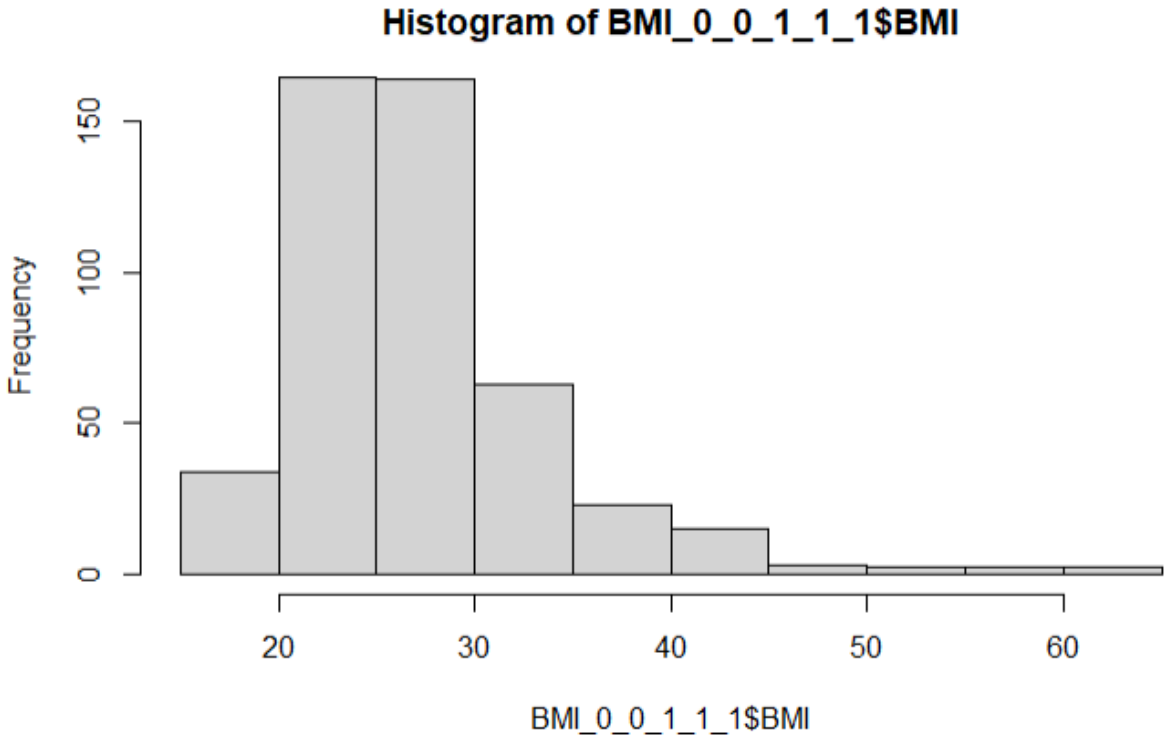




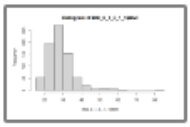


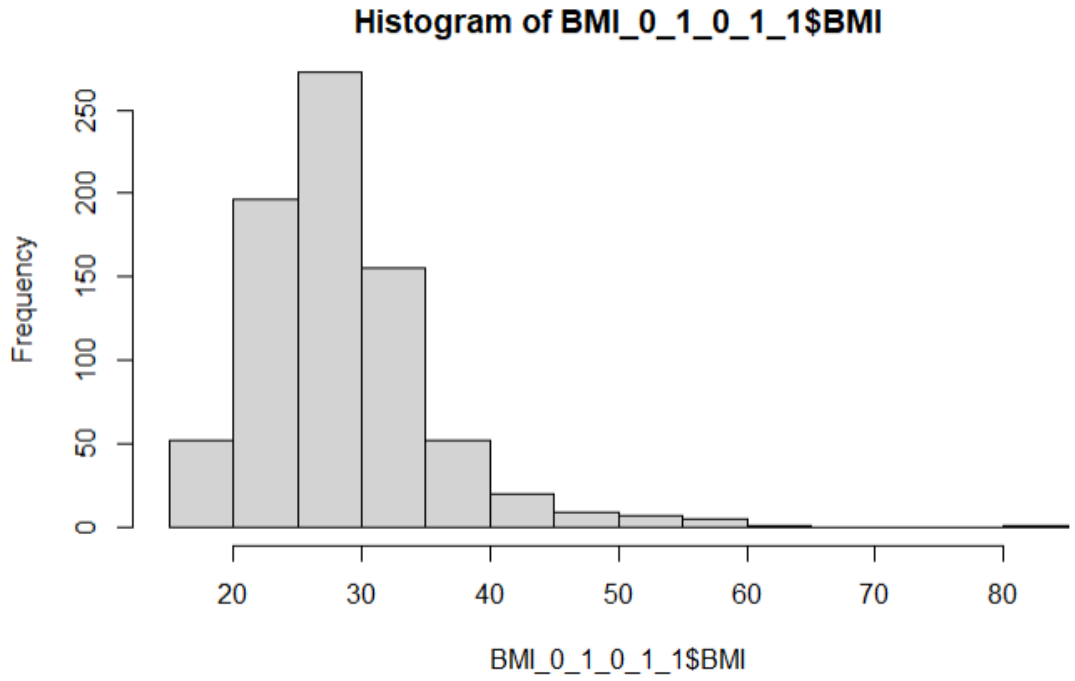
```{r, echo = FALSE}

BMI_0_0_1_1_1 = data.frame(HD[HD$Combined == '0_0_1_1_1',])

BMI_0_0_1_1_1


BMI_mean20 = mean(BMI_0_0_1_1_1$BMI)
BMI_variances20 = var(BMI_0_0_1_1_1$BMI)
BMI_mean20
BMI_variances20


hist(BMI_0_0_1_1_1$BMI)
```



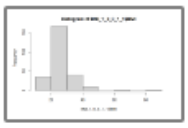


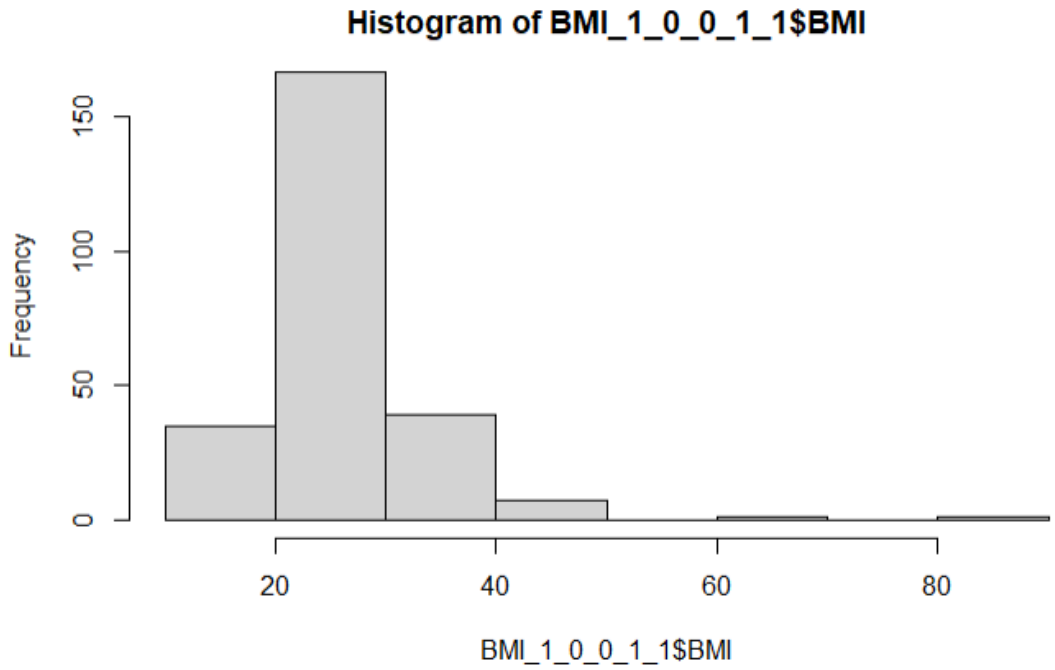
```{r, echo = FALSE}

BMI_0_1_0_1_1 = data.frame(HD[HD$Combined == '0_1_0_1_1',])

BMI_0_1_0_1_1


BMI_mean21 = mean(BMI_0_1_0_1_1$BMI)
BMI_variances21 = var(BMI_0_1_0_1_1$BMI)
BMI_mean21
BMI_variances21


hist(BMI_0_1_0_1_1$BMI)
```



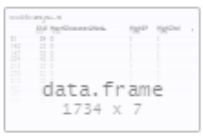


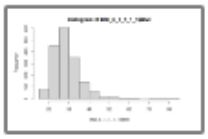
```{r, echo = FALSE}

BMI_1_0_0_1_1 = data.frame(HD[HD$Combined == '1_0_0_1_1',])

BMI_1_0_0_1_1


BMI_mean22 = mean(BMI_1_0_0_1_1$BMI)
BMI_variances22 = var(BMI_1_0_0_1_1$BMI)
BMI_mean22
BMI_variances22


hist(BMI_1_0_0_1_1$BMI)
```

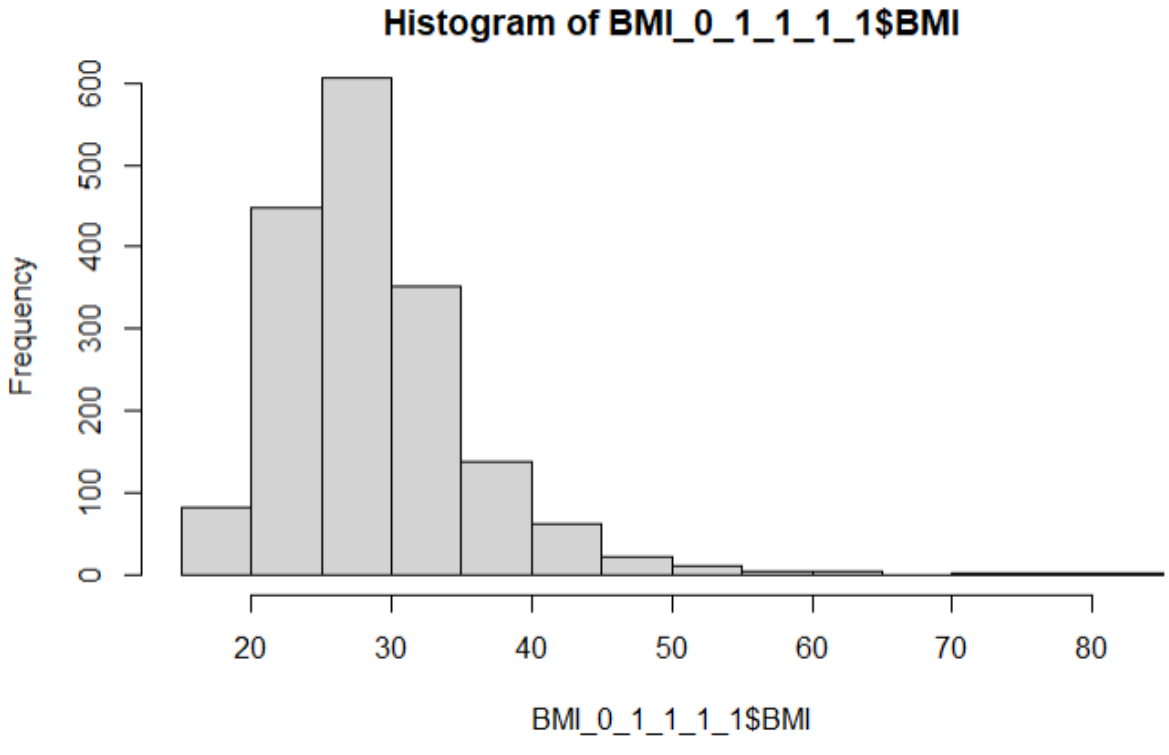




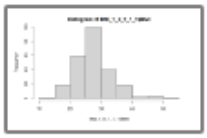
```{r, echo = FALSE}

BMI_0_1_1_1_1 = data.frame(HD[HD$Combined == '0_1_1_1_1',])

BMI_0_1_1_1_1


BMI_mean23 = mean(BMI_0_1_1_1_1$BMI)
BMI_variances23 = var(BMI_0_1_1_1_1$BMI)
BMI_mean23
BMI_variances23


hist(BMI_0_1_1_1_1$BMI)
```

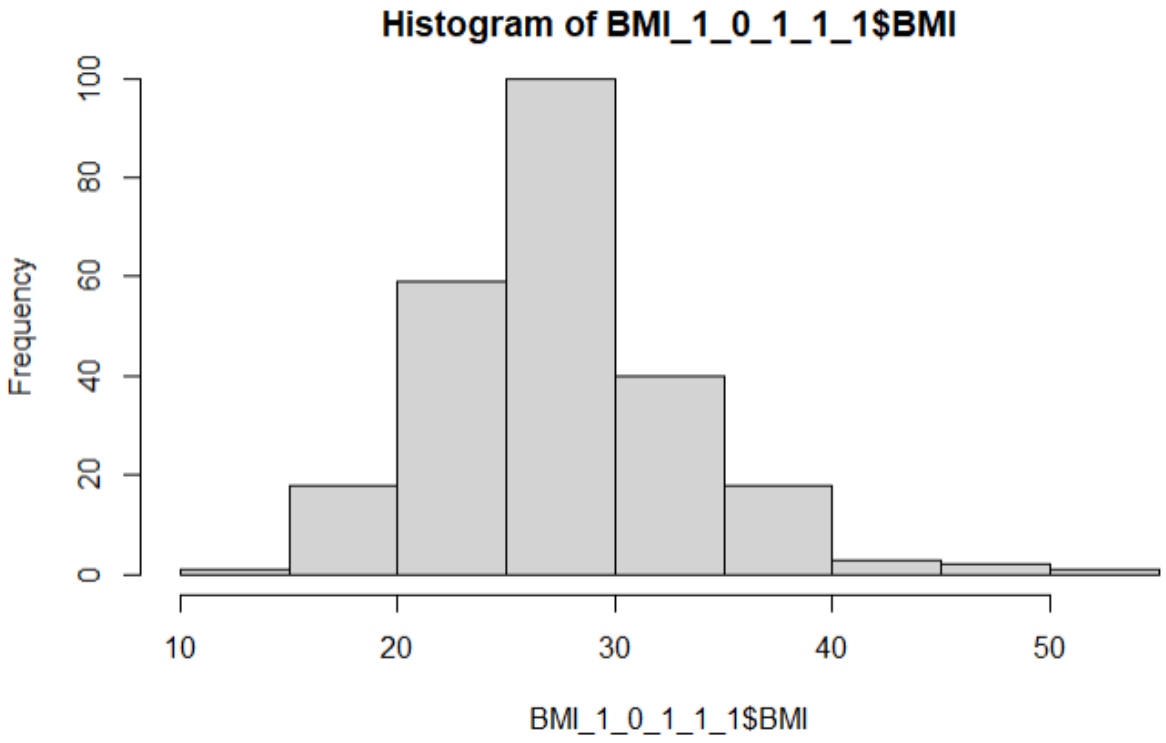




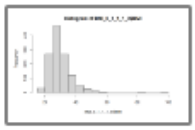


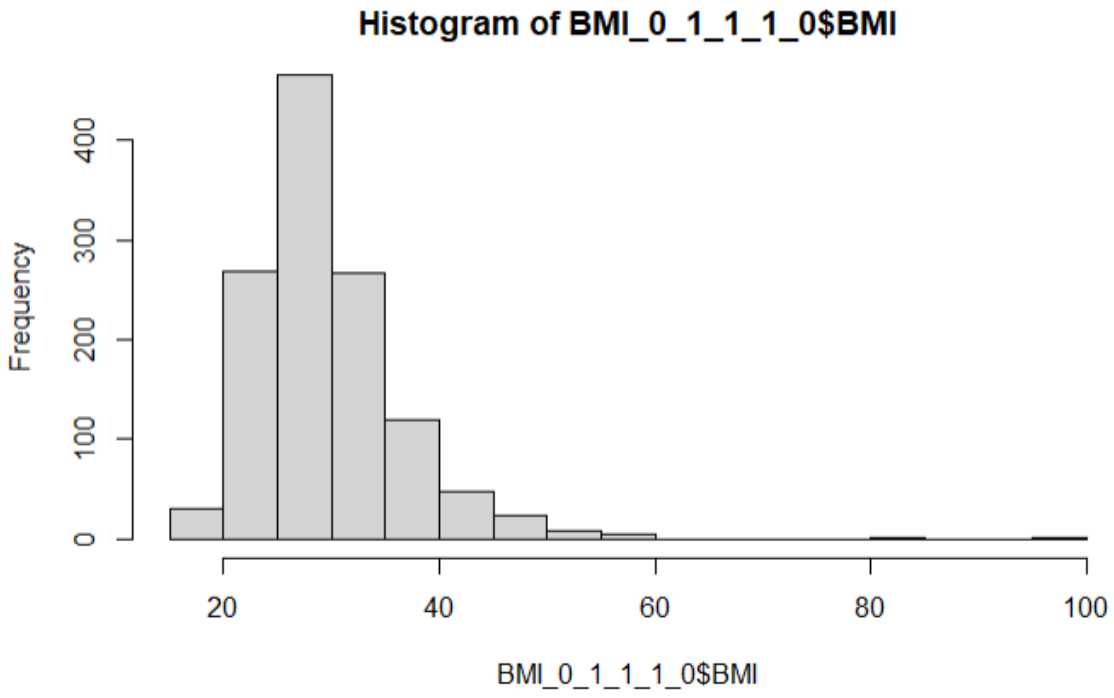
```{r, echo = FALSE}

BMI_1_0_1_1_1 = data.frame(HD[HD$Combined == '1_0_1_1_1',])

BMI_1_0_1_1_1


BMI_mean24 = mean(BMI_1_0_1_1_1$BMI)
BMI_variances24 = var(BMI_1_0_1_1_1$BMI)
BMI_mean24
BMI_variances24


hist(BMI_1_0_1_1_1$BMI)
```



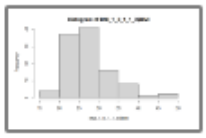


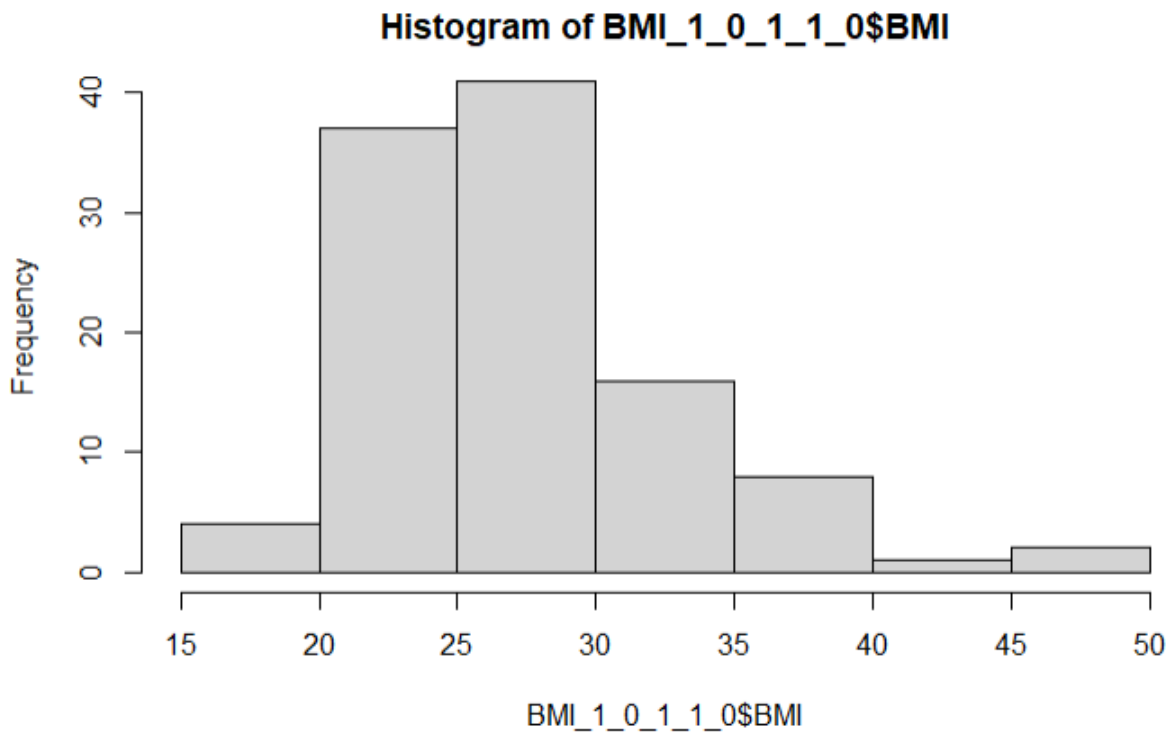
```{r, echo = FALSE}

BMI_0_1_1_1_0 = data.frame(HD[HD$Combined == '0_1_1_1_0',])

BMI_0_1_1_1_0


BMI_mean25 = mean(BMI_0_1_1_1_0$BMI)
BMI_variances25 = var(BMI_0_1_1_1_0$BMI)
BMI_mean25
BMI_variances25


hist(BMI_0_1_1_1_0$BMI)
```



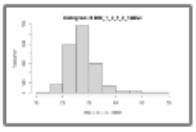


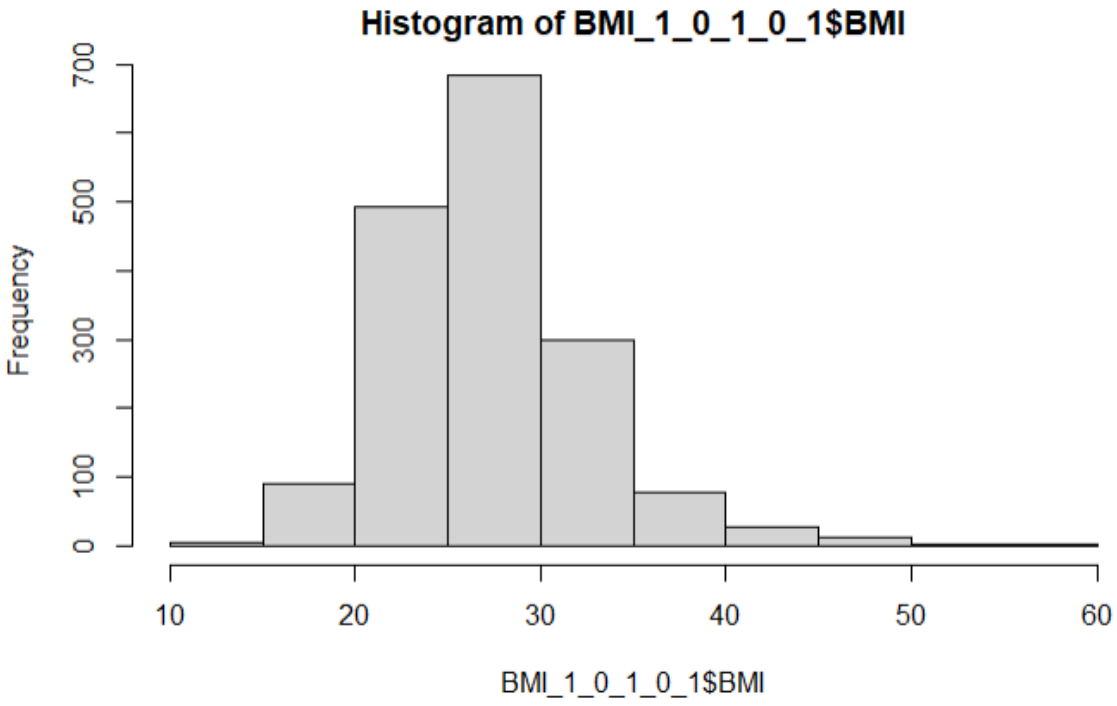
```{r, echo = FALSE}

BMI_1_0_1_1_0 = data.frame(HD[HD$Combined == '1_0_1_1_0',])

BMI_1_0_1_1_0


BMI_mean26 = mean(BMI_1_0_1_1_0$BMI)
BMI_variances26 = var(BMI_1_0_1_1_0$BMI)
BMI_mean26
BMI_variances26


hist(BMI_1_0_1_1_0$BMI)
```



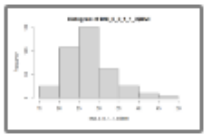


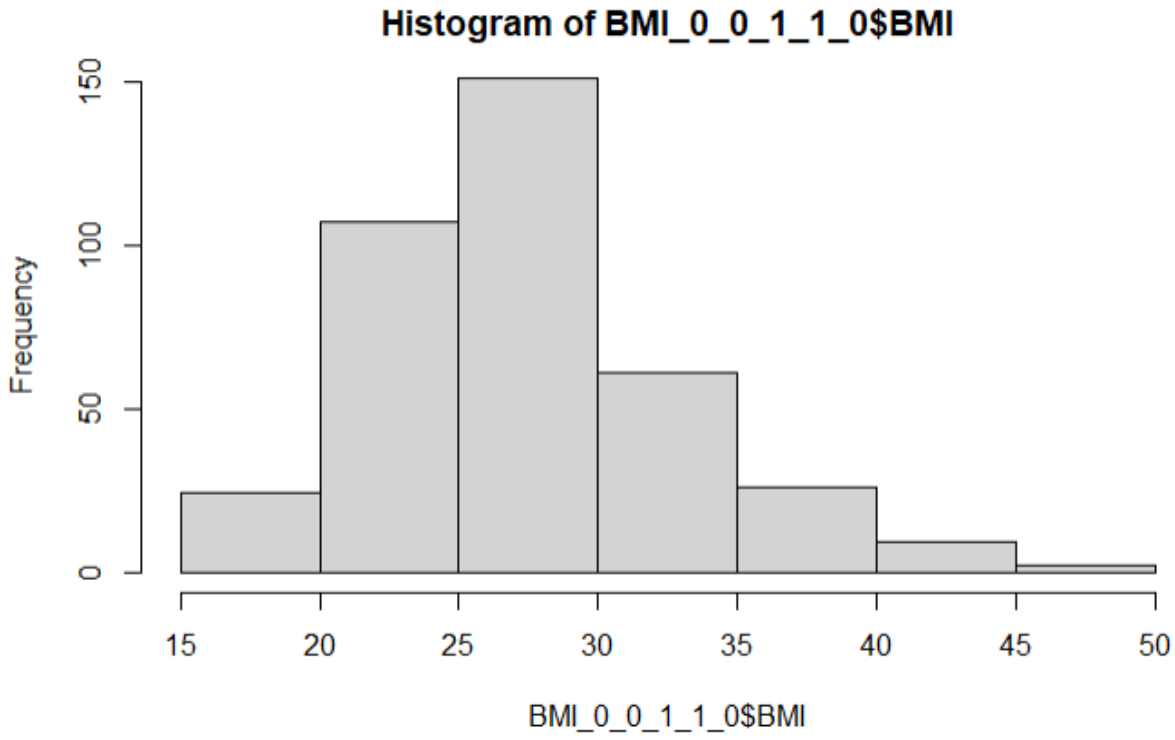
```{r, echo = FALSE}

BMI_1_0_1_0_1 = data.frame(HD[HD$Combined == '1_0_1_0_1',])

BMI_1_0_1_0_1


BMI_mean27 = mean(BMI_1_0_1_0_1$BMI)
BMI_variances27 = var(BMI_1_0_1_0_1$BMI)
BMI_mean27
BMI_variances27


hist(BMI_1_0_1_0_1$BMI)
```



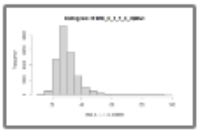


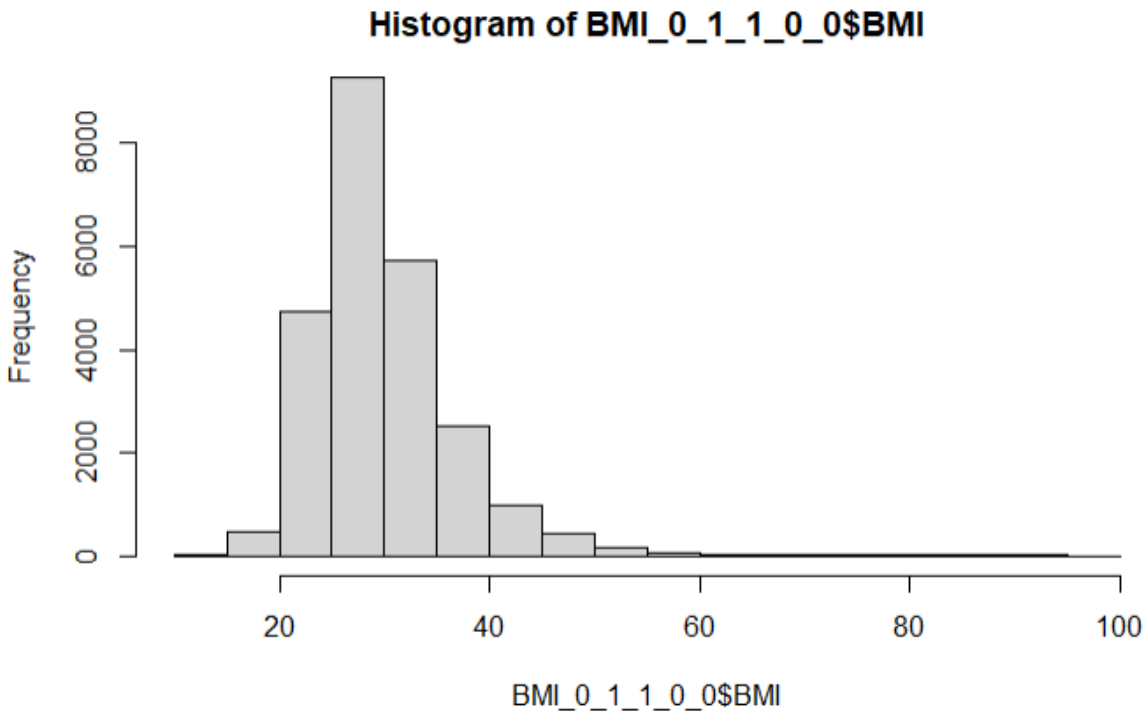
```{r, echo = FALSE}

BMI_0_0_1_1_0 = data.frame(HD[HD$Combined == '0_0_1_1_0',])

BMI_0_0_1_1_0


BMI_mean28 = mean(BMI_0_0_1_1_0$BMI)
BMI_variances28 = var(BMI_0_0_1_1_0$BMI)
BMI_mean28
BMI_variances28


hist(BMI_0_0_1_1_0$BMI)
```



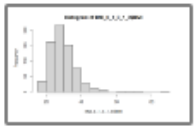


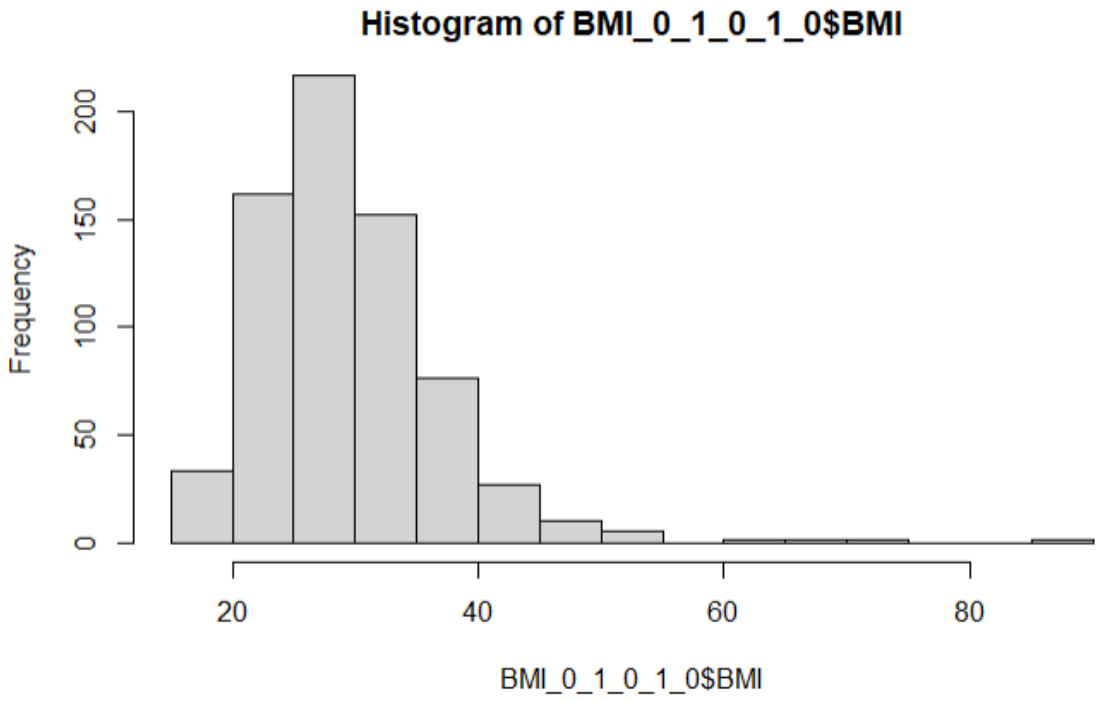
```{r, echo = FALSE}

BMI_0_1_1_0_0 = data.frame(HD[HD$Combined == '0_1_1_0_0',])

BMI_0_1_1_0_0


BMI_mean29 = mean(BMI_0_1_1_0_0$BMI)
BMI_variances29 = var(BMI_0_1_1_0_0$BMI)
BMI_mean29
BMI_variances29


hist(BMI_0_1_1_0_0$BMI)
```



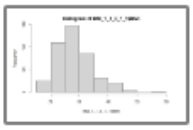


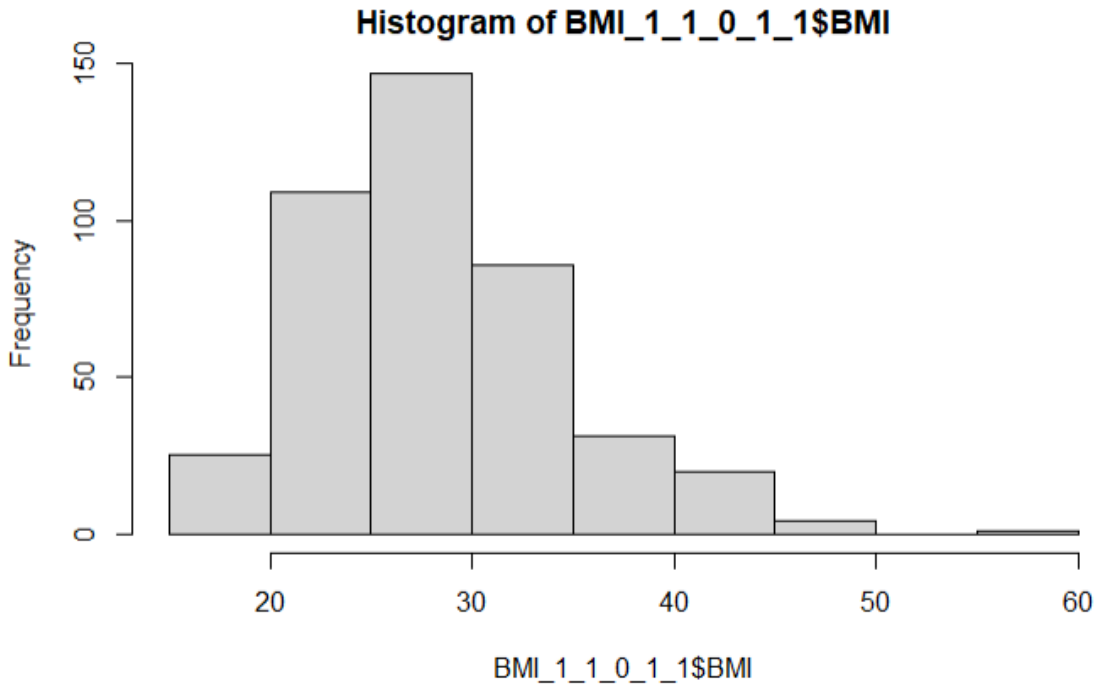
```{r, echo = FALSE}

BMI_0_1_0_1_0 = data.frame(HD[HD$Combined == '0_1_0_1_0',])

BMI_0_1_0_1_0


BMI_mean30 = mean(BMI_0_1_0_1_0$BMI)
BMI_variances30 = var(BMI_0_1_0_1_0$BMI)
BMI_mean30
BMI_variances30


hist(BMI_0_1_0_1_0$BMI)
```



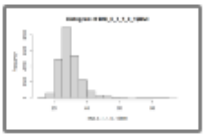


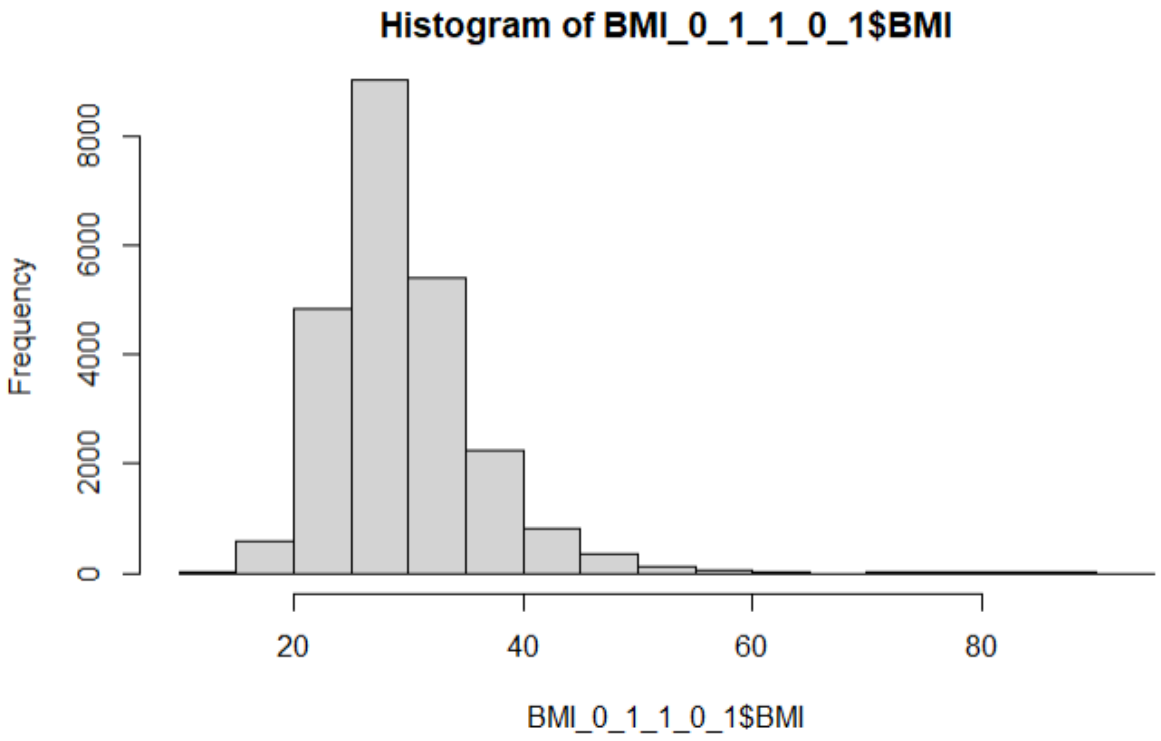
```{r, echo = FALSE}

BMI_1_1_0_1_1 = data.frame(HD[HD$Combined == '1_1_0_1_1',])

BMI_1_1_0_1_1


BMI_mean31 = mean(BMI_1_1_0_1_1$BMI)
BMI_variances31 = var(BMI_1_1_0_1_1$BMI)
BMI_mean31
BMI_variances31


hist(BMI_1_1_0_1_1$BMI)
```



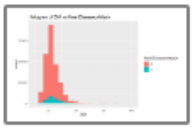


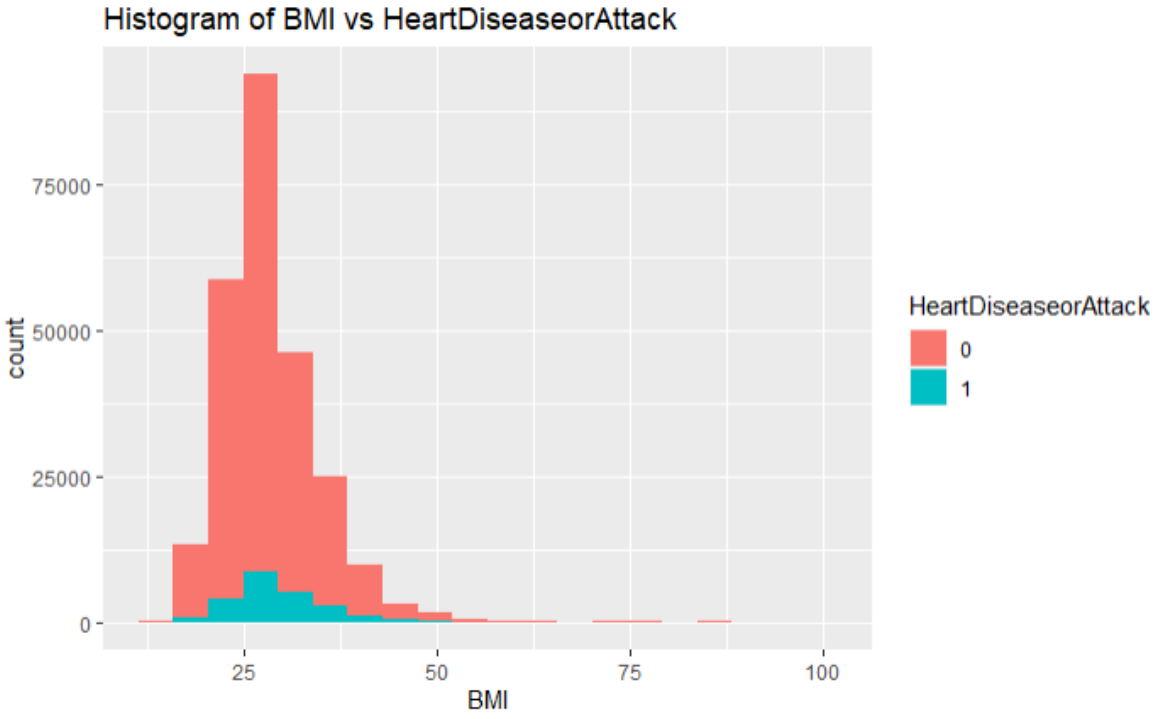
```{r, echo = FALSE}

BMI_0_1_1_0_1 = data.frame(HD[HD$Combined == '0_1_1_0_1',])

BMI_0_1_1_0_1


BMI_mean32 = mean(BMI_0_1_1_0_1$BMI)
BMI_variances32 = var(BMI_0_1_1_0_1$BMI)
BMI_mean32
BMI_variances32


hist(BMI_0_1_1_0_1$BMI)
```



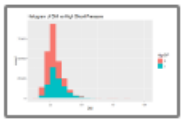


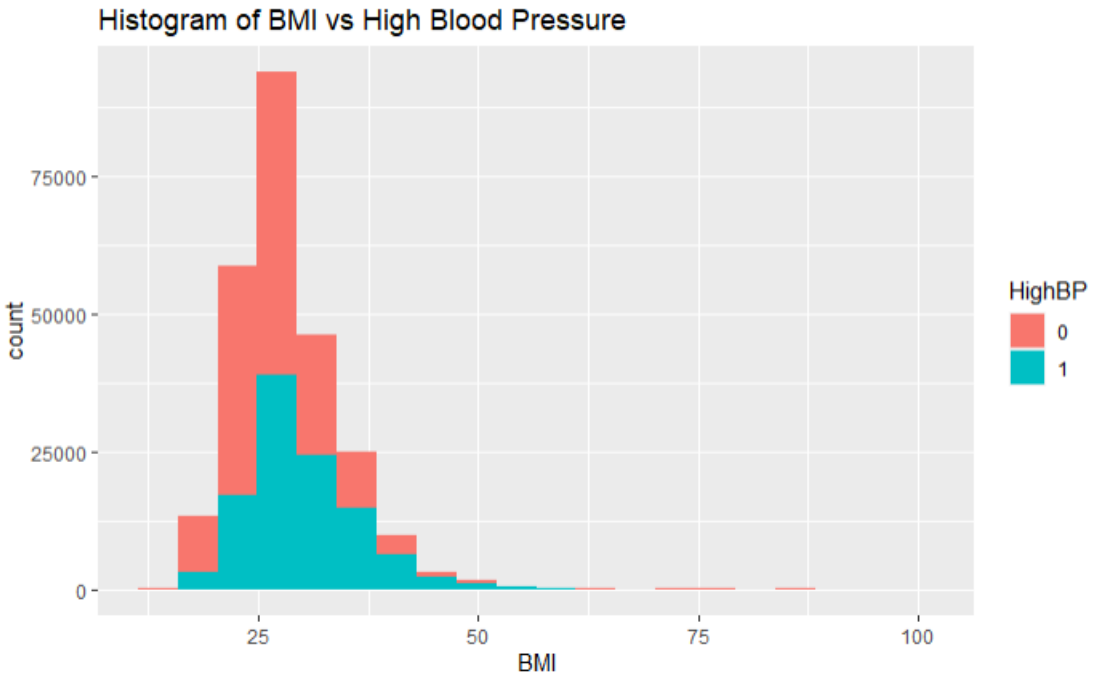
```{r, echo = FALSE}
HD$HeartDiseaseorAttack=as.factor(HD$HeartDiseaseorAttack)

xtabs(~HeartDiseaseorAttack, data = HD)
xtabs(~HeartDiseaseorAttack + BMI, data = HD)

library(ggplot2)
ggplot(data=HD, aes(x=BMI,fill = HeartDiseaseorAttack)) +
  geom_histogram(bins=20,position='stack')+
  labs(x ='BMI', title ='Histogram of BMI vs HeartDiseaseorAttack')
```



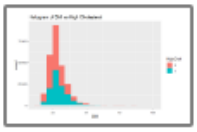


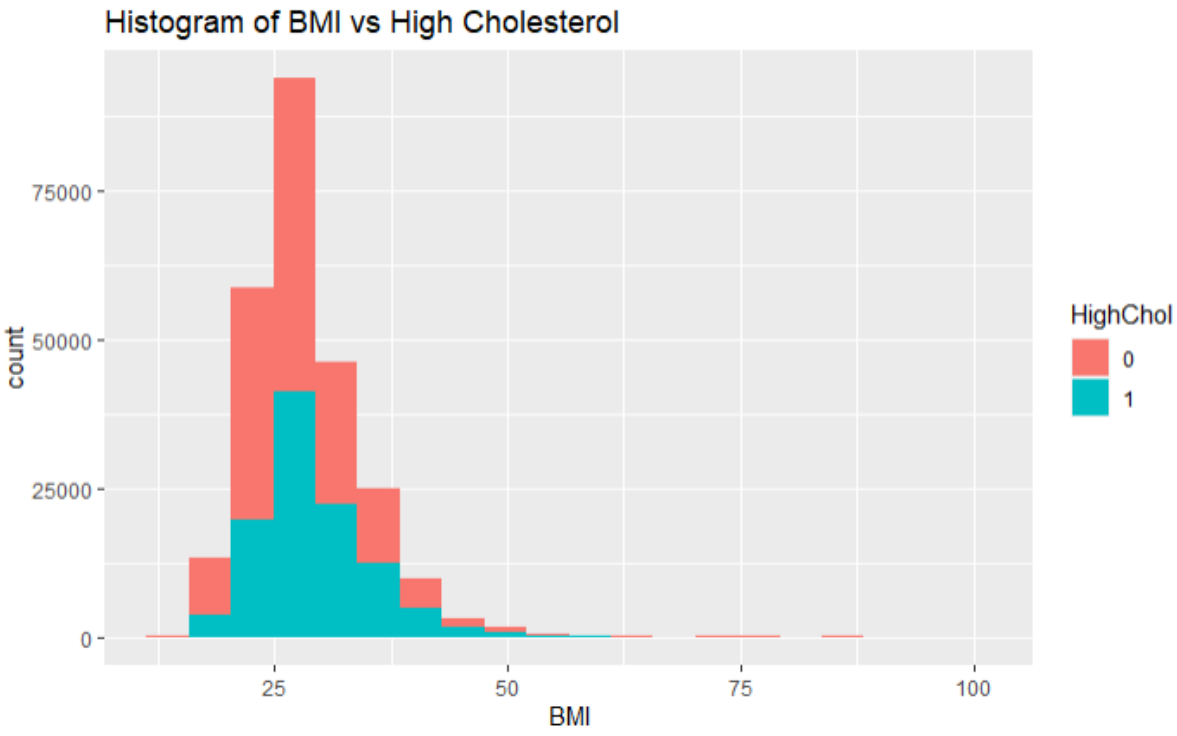
```r
HD$HighBP=as.factor(HD$HighBP)

xtabs(~HighBP, data = HD)
xtabs(~HighBP + BMI, data = HD)

library(ggplot2)
ggplot(data=HD, aes(x=BMI,fill = HighBP)) +
  geom_histogram(bins=20,position='stack')+
  labs(x ='BMI', title ='Histogram of BMI vs High Blood Pressure')
```



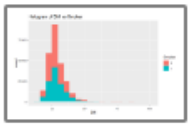


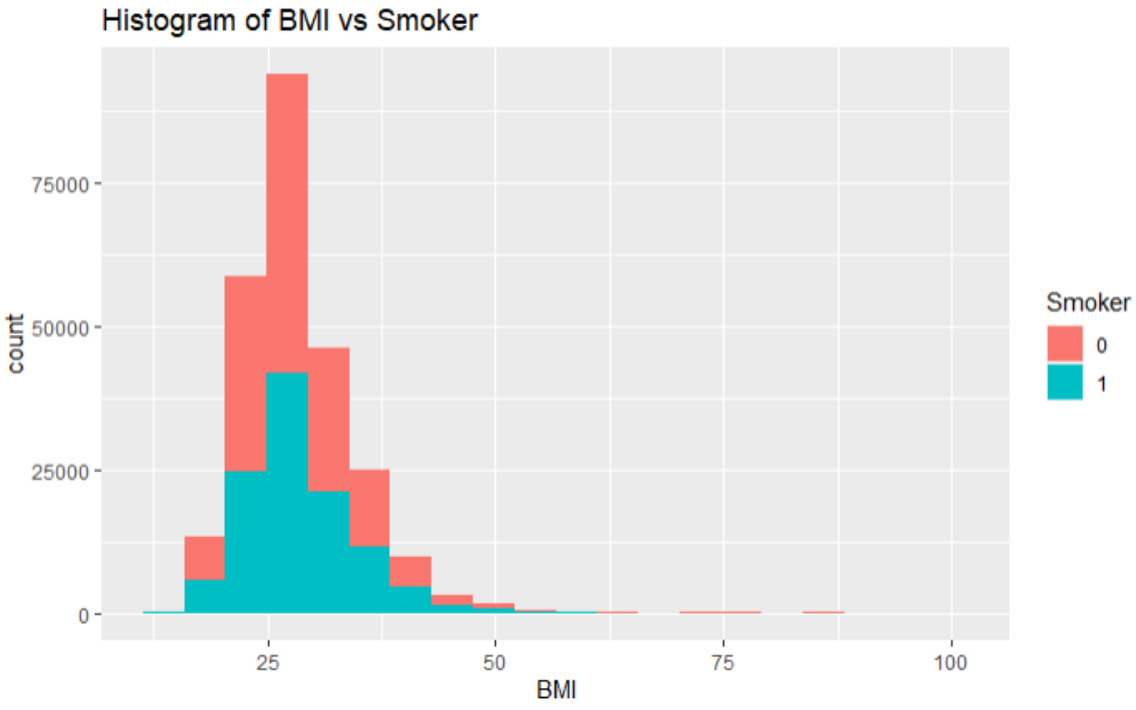
```{r, echo = FALSE}
HD$HighChol=as.factor(HD$HighChol)

xtabs(~HighChol, data = HD)
xtabs(~HighChol + BMI, data = HD)

library(ggplot2)
ggplot(data=HD, aes(x=BMI,fill = HighChol)) +
  geom_histogram(bins=20,position='stack')+
  labs(x ='BMI', title ='Histogram of BMI vs High Cholesterol')
```



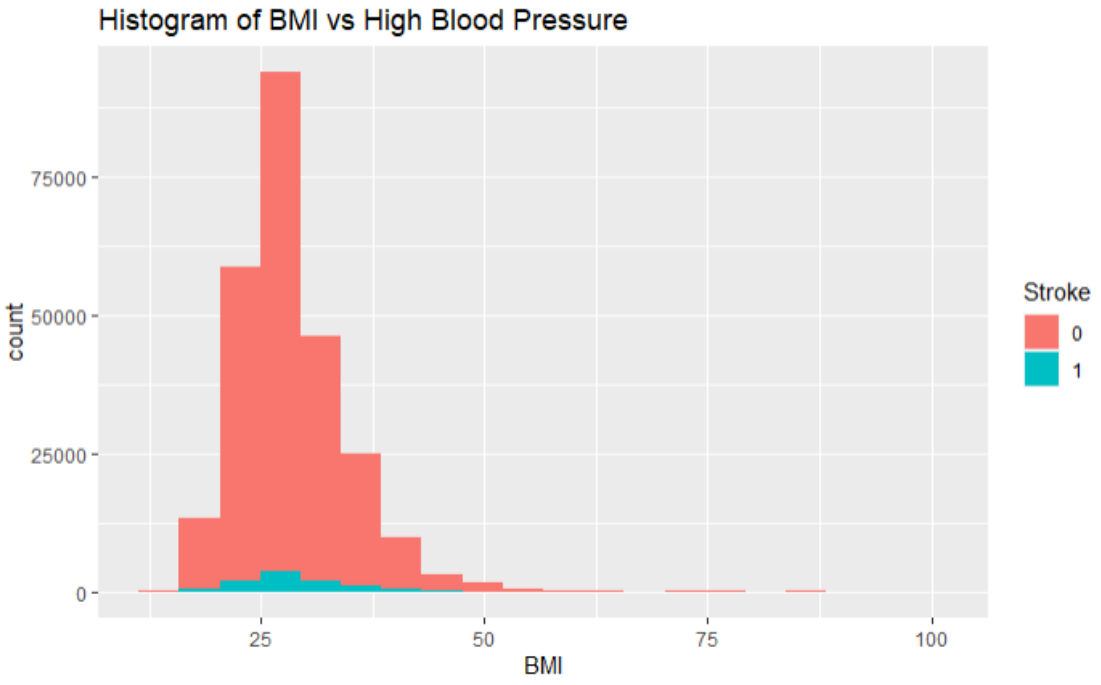


```r
HD$Smoker=as.factor(HD$Smoker)

xtabs(~Smoker, data = HD)
xtabs(~Smoker + BMI, data = HD)

library(ggplot2)
ggplot(data=HD, aes(x=BMI,fill = Smoker)) +
  geom_histogram(bins=20,position='stack')+
  labs(x ='BMI', title ='Histogram of BMI vs Smoker')
```

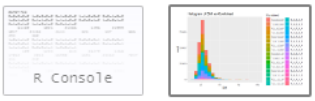


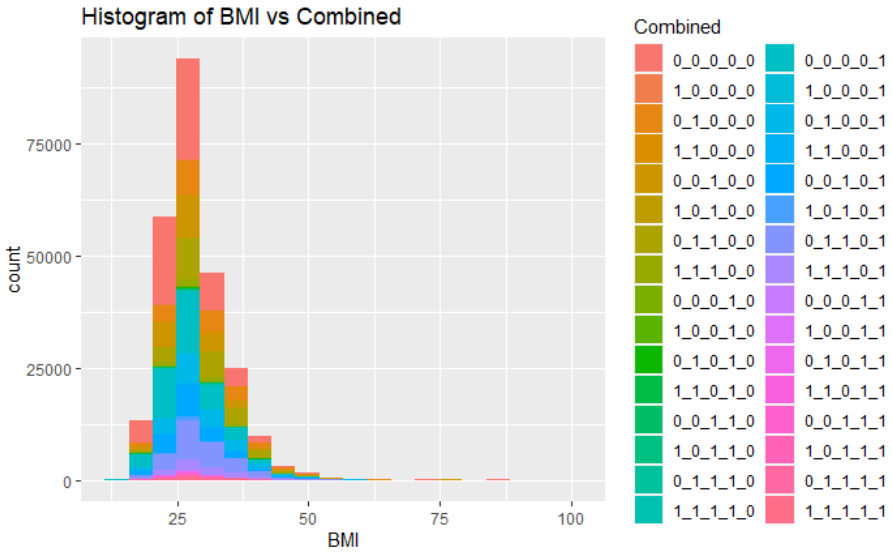


```r
HD$Stroke=as.factor(HD$Stroke)

xtabs(~Stroke, data = HD)
xtabs(~Stroke + BMI, data = HD)

library(ggplot2)
ggplot(data=HD, aes(x=BMI,fill = Stroke)) +
  geom_histogram(bins=20,position='stack')+
  labs(x ='BMI', title ='Histogram of BMI vs Stroke')
```



```{r, echo = FALSE}
HD$Combined=as.factor(HD$Combined)

xtabs(~Combined, data = HD)
xtabs(~Combined + BMI, data = HD)

library(ggplot2)
ggplot(data=HD, aes(x=BMI,fill = Combined)) +
  geom_histogram(bins=20,position='stack')+
  labs(x ='BMI', title ='Histogram of BMI vs Combined')
```





- Since all the histograms are not bell-shaped, we prove that it is not normally distributed. Therefore, we conclude that the Normality assumption is violated.

Perform the one-way ANOVA and Tukey-Kramer's simultaneous pairwise comparison as if all Normality and equal-variance assumptions hold.

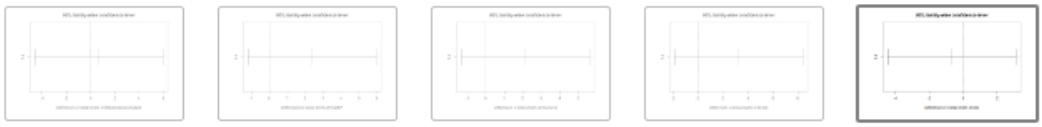
```{r, echo = FALSE}
# Get the ANOVA
fit_1=aov(BMI~HeartDiseaseorAttack,data=HD)
anova(fit_1)
fit_2=aov(BMI~HighBP,data=HD)
anova(fit_2)
fit_3=aov(BMI~HighChol,data=HD)
anova(fit_3)
fit_4=aov(BMI~Stroke,data=HD)
anova(fit_4)
fit_5=aov(BMI~Smoker,data=HD)
anova(fit_5)
fit_6=aov(BMI~Combined,data=HD)
anova(fit_6)
```

```
Analysis of Variance Table

Response: BMI
                       Df   Sum Sq Mean Sq F value    Pr(>F)
HeartDiseaseorAttack      1    31010 31009.7     712 < 2.2e-16 ***
Residuals            253678 11048380    43.6
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Analysis of Variance Table

Response: BMI
             Df   Sum Sq Mean Sq F value    Pr(>F)
HighBP        1   506198  506198   12145 < 2.2e-16 ***
Residuals 253678 10573191      42
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Analysis of Variance Table

Response: BMI
             Df   Sum Sq Mean Sq F value    Pr(>F)
HighChol      1   126190  126190  2922.6 < 2.2e-16 ***
Residuals 253678 10953200      43
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Analysis of Variance Table

Response: BMI
             Df   Sum Sq Mean Sq F value    Pr(>F)
Stroke        1     4500  4499.7  103.07 < 2.2e-16 ***
Residuals 253678 11074890    43.7
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Analysis of Variance Table

Response: BMI
```

```{r, echo = FALSE}
# Get the Tukey
tukey_1=TukeyHSD(fit_1, conf.level=.95)
tukey_1

tukey_2=TukeyHSD(fit_2, conf.level=.95)
tukey_2

tukey_3=TukeyHSD(fit_3, conf.level=.95)
tukey_3

tukey_4=TukeyHSD(fit_4, conf.level=.95)
tukey_4

tukey_5=TukeyHSD(fit_5, conf.level=.95)
tukey_5

tukey_6=TukeyHSD(fit_6, conf.level=.95)
tukey_6
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = BMI ~ HeartDiseaseorAttack, data = HD)

$HeartDiseaseorAttack
        diff      lwr      upr p adj
1-0 1.196998 1.109076 1.284921     0

  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = BMI ~ HighBP, data = HD)

$HighBP
        diff      lwr      upr p adj
1-0 2.854107 2.803347 2.904867     0

  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = BMI ~ HighChol, data = HD)

$HighChol
        diff      lwr      upr p adj
1-0 1.427114 1.375374 1.478853     0

  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = BMI ~ Stroke, data = HD)

$Stroke
         diff       lwr       upr p adj
```

```{r, echo = FALSE}
# plot the tukey
plot(tukey_1)
plot(tukey_2)
plot(tukey_3)
plot(tukey_4)
plot(tukey_5)
```



Check whether the equal variance assumption is violated? That is, members of {(σ_1)^2 ,(σ_2)^2 ,..,(σ_K)^2 } are not all equal.

- We find the maximum and minimum of the whole list of variances in order to see if the equal variance assumption is violated

```{r, echo = FALSE}
total_variances = list(BMI_variances1,BMI_variances2,BMI_variances3,BMI_variances4,BMI_variances5,BMI_variances6,BMI_variances7,BMI_variances8,BMI_variances9,BMI_variances10,BMI_variances11,BMI_variances12,BMI_variances13,BMI_variances14,BMI_variances15,BMI_variances16,BMI_variances17,BMI_variances18,BMI_variances19,BMI_variances20,BMI_variances21,BMI_variances22,BMI_variances23,BMI_variances24,BMI_variances25,BMI_variances26,BMI_variances27,BMI_variances28,BMI_variances29,BMI_variances30,BMI_variances31,BMI_variances32)
total_variances
```

- If the result of the maximum of the squared variance divided by the minimum of the squared variance is around 1, then the variances are all equal.

```{r, echo = FALSE}
x <- BMI_variances12
y <- BMI_variances27
z = (x^2)/(y^2)
z
```

```
[1] 7.331883
```

- Since the result is greater than 1, we conclude that the equal variance assumption is violated.

Construct HC-tree to discover the community structures among the K samples?

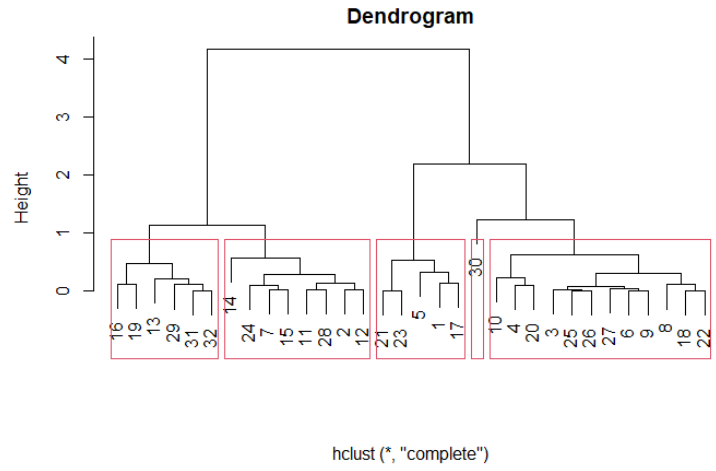
- Constructing HC-tree for 32 means:

```{r, echo = FALSE}
total_mean = list(BMI_mean1,BMI_mean2,BMI_mean3,BMI_mean4,BMI_mean5,BMI_mean6,BMI_mean7,BMI_mean8,BMI_mean9,BMI_mean10,BMI_mean11,BMI_mean12,
BMI_mean13,BMI_mean14,BMI_mean15,BMI_mean16,BMI_mean17,BMI_mean18,BMI_mean19,BMI_mean20,BMI_mean21,BMI_mean22,BMI_mean23,BMI_mean24,BMI_mean2
5,BMI_mean26,BMI_mean27,BMI_mean28,BMI_mean29,BMI_mean30,BMI_mean31,BMI_mean32)

clusters <- hclust(dist(total_mean))
plot(clusters,xlab='',main='Dendrogram')

# cut off the tree at the desired number of clusters using cutree.
clusterCut <- cutree(clusters, 5)
# plot the tree
rect.hclust(clusters, k=5)
```

Construct HC-tree to discover the community structures among the K samples?

- Constructing HC-tree for 32 means:

```{r, echo = FALSE}
total_mean = list(BMI_mean1,BMI_mean2,BMI_mean3,BMI_mean4,BMI_mean5,BMI_mean6,BMI_mean7,BMI_mean8,BMI_mean9,BMI_mean10,BMI_mean11,BMI_mean12,
BMI_mean13,BMI_mean14,BMI_mean15,BMI_mean16,BMI_mean17,BMI_mean18,BMI_mean19,BMI_mean20,BMI_mean21,BMI_mean22,BMI_mean23,BMI_mean24,BMI_mean2
5,BMI_mean26,BMI_mean27,BMI_mean28,BMI_mean29,BMI_mean30,BMI_mean31,BMI_mean32)

clusters <- hclust(dist(total_mean))
plot(clusters,xlab='',main='Dendrogram')

# cut off the tree at the desired number of clusters using cutree.
clusterCut <- cutree(clusters, 5)
# plot the tree
rect.hclust(clusters, k=5)
```



- Based on the result of each tree, we can see that in each cluster, there are chunks that contain leafs similar to one another. However, only one of the clusters has one chunk that contains one leaf which means that it is substantially different from the rest of the remaining leafs. This tells us that the distribution of this leaf is significantly different from the distribution of the remaining.

- Based on the result of each tree, we can see that in each cluster, there are chunks that contain leafs similar to one another. However, only one of the clusters has one chunk that contains one leaf which means that it is substantially different from the rest of the remaining leafs. This tells us that the distribution of this leaf is significantly different from the distribution of the remaining.

Compare results from ANOVA and and Tukey-Kramer's comparison with results found in HC-tree.

- It is stated that the null hypothesis in the ANOVA is that there is no difference among the group mean, and the alternative hypothesis is that the means are not all similar to one another. Based on the results from the ANOVA and Tukey-Kramer, we can see that the p-values of all the samples are less than the significance level of 0.05, which indicates that we reject the null hypothesis. Therefore, there are significance differences in the means.

- Based on the results from the HC tree, we can see that the clusters among the K samples (which we are doing 5) show that there are chunks (which represents the mean) in each cluster similar to one another than the other chunks. However, there is one chunk which separates from the rest as it only contains one leaf. This indicates that the distribution of this chunk is significantly different from the rest. However, the majority of the leafs are not the same, but close to one another.

- In conclusion, the results of the ANOVA and Tukey show that there is significant difference in the group mean. In the HC tree, we can see that results that there are no similarity in the leafs.

Code Appendix