Final Exam

Ajay Kanubhai Patel

2022-12-13

#Loading the package

```r
#Load packages to convert file in PDF.

if(!require(tinytex)){install.packages("tinytex")}

## Loading required package: tinytex
```

This section is for the basic set up. It will clear all the plots, the console and the workspace. It also sets the overall format for numbers.

```r
if(!is.null(dev.list())) dev.off()

## null device
##           1

cat("\014")
```

```
rm(list=ls())
options(scipen=9)
```

#To read Excel file in R data frame.

```
if(!require(readxl)){install.packages("readxl")}

## Loading required package: readxl

library("readxl")

if(!require(pastecs)){install.packages("pastecs")}

## Loading required package: pastecs

library("pastecs")

if(!require(lattice)){install.packages("lattice")}

## Loading required package: lattice

library("lattice")

if(!require(vcd)){install.packages("vcd")}

## Loading required package: vcd

## Loading required package: grid

library("vcd")

if(!require(HSAUR)){install.packages("HSAUR")}

## Loading required package: HSAUR

## Loading required package: tools

library("HSAUR")

if(!require(rmarkdown)){install.packages("rmarkdown")}

## Loading required package: rmarkdown

library("rmarkdown")


if(!require(ggplot2)){install.packages("ggplot2")}

## Loading required package: ggplot2

library("ggplot2")

if(!require(klaR)){install.packages("klaR")}
```

```
## Loading required package: klaR

## Loading required package: MASS

library("klaR")

if(!require(partykit)){install.packages("partykit")}

## Loading required package: partykit

## Loading required package: libcoin

## Loading required package: mvtnorm

library("partykit")
```

To get working directory

To read PROG8430_Assign04_22F.txt file located at

“D:/Final Assignment/DATA/Assignment5”

```
getwd()

## [1] "D:/Final Assignment/DATA/FINAL EXAM"

Train_AP <- read_excel("PROG8430_Final_22F_train.xlsx")

head(Train_AP)

## # A tibble: 6 × 8
##   Cancer    Age    HW Hst     Exe Smk   Drk   Hlth
##   <chr>   <dbl> <dbl> <chr> <dbl> <chr> <chr> <chr>
## 1 0          43 1.13  1        53 0     0     VG
## 2 0          39 1.21  0        38 0     0     A
## 3 1          49 0.898 0        18 1     1     P
## 4 0          59 1.03  0         9 1     1     P
## 5 0          45 1.08  0       127 0     0     VG
## 6 0          53 1.06  1        11 0     0     A

str(Train_AP)

## tibble [695 × 8] (S3: tbl_df/tbl/data.frame)
##  $ Cancer: chr [1:695] "0" "0" "1" "0" ...
##  $ Age   : num [1:695] 43 39 49 59 45 53 69 50 40 50 ...
##  $ HW    : num [1:695] 1.127 1.214 0.898 1.028 1.082 ...
##  $ Hst   : chr [1:695] "1" "0" "0" "0" ...
##  $ Exe   : num [1:695] 53 38 18 9 127 11 8 7 51 11 ...
##  $ Smk   : chr [1:695] "0" "0" "1" "1" ...
##  $ Drk   : chr [1:695] "0" "0" "1" "1" ...
##  $ Hlth  : chr [1:695] "VG" "A" "P" "P" ...
```

Rename all variables with your initials appended.

# to change all column name by appending my initials (Ajay Patel = AP) and separate it by "_".

```
colnames(Train_AP) <- paste(colnames(Train_AP), "AP", sep = "_")
head(Train_AP)

## # A tibble: 6 × 8
##   Cancer_AP Age_AP HW_AP Hst_AP Exe_AP Smk_AP Drk_AP Hlth_AP
##   <chr>      <dbl> <dbl> <chr>   <dbl> <chr>  <chr>  <chr>  
## 1 0             43 1.13  1          53 0      0      VG     
## 2 0             39 1.21  0          38 0      0      A      
## 3 1             49 0.898 0          18 1      1      P      
## 4 0             59 1.03  0           9 1      1      P      
## 5 0             45 1.08  0         127 0      0      VG     
## 6 0             53 1.06  1          11 0      0      A      

str(Train_AP)

## tibble [695 × 8] (S3: tbl_df/tbl/data.frame)
##  $ Cancer_AP: chr [1:695] "0" "0" "1" "0" ...
##  $ Age_AP   : num [1:695] 43 39 49 59 45 53 69 50 40 50 ...
##  $ HW_AP    : num [1:695] 1.127 1.214 0.898 1.028 1.082 ...
##  $ Hst_AP   : chr [1:695] "1" "0" "0" "0" ...
##  $ Exe_AP   : num [1:695] 53 38 18 9 127 11 8 7 51 11 ...
##  $ Smk_AP   : chr [1:695] "0" "0" "1" "1" ...
##  $ Drk_AP   : chr [1:695] "0" "0" "1" "1" ...
##  $ Hlth_AP  : chr [1:695] "VG" "A" "P" "P" ...
```

Convert Character variables into factor variables.

```
Train_AP$Cancer_AP <- as.factor(Train_AP$Cancer_AP)
Train_AP$Hst_AP <- as.factor(Train_AP$Hst_AP)
Train_AP$Smk_AP <- as.factor(Train_AP$Smk_AP)
Train_AP$Drk_AP <- as.factor(Train_AP$Drk_AP)
Train_AP$Hlth_AP <- as.factor(Train_AP$Hlth_AP)

str(Train_AP)

## tibble [695 × 8] (S3: tbl_df/tbl/data.frame)
##  $ Cancer_AP: Factor w/ 2 levels "0","1": 1 1 2 1 1 1 2 1 1 1 ...
##  $ Age_AP   : num [1:695] 43 39 49 59 45 53 69 50 40 50 ...
##  $ HW_AP    : num [1:695] 1.127 1.214 0.898 1.028 1.082 ...
##  $ Hst_AP   : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 2 1 1 1 ...
##  $ Exe_AP   : num [1:695] 53 38 18 9 127 11 8 7 51 11 ...
##  $ Smk_AP   : Factor w/ 2 levels "0","1": 1 1 2 2 1 1 2 1 1 1 ...
##  $ Drk_AP   : Factor w/ 2 levels "0","1": 1 1 2 2 1 1 2 1 1 1 ...
##  $ Hlth_AP  : Factor w/ 5 levels "A","G","P","VG",..: 4 1 3 3 4 1 3 1 1 3
...
```

Dimension reduction process.

let's check missing factor variables.

```
summary(Train_AP)

##  Cancer_AP     Age_AP          HW_AP        Hst_AP      Exe_AP        Smk_AP
##  0:344      Min.   :22.00   Min.   :0.678   0:447   Min.   :  1.00   0:342
##  1:351      1st Qu.:42.00   1st Qu.:0.948   1:248   1st Qu.: 15.00   1:353
##             Median :48.00   Median :1.031           Median : 26.00
##             Mean   :47.09   Mean   :1.040           Mean   : 31.45
##             3rd Qu.:52.00   3rd Qu.:1.115           3rd Qu.: 42.00
##             Max.   :70.00   Max.   :8.500           Max.   :140.00
##  Drk_AP   Hlth_AP
##  0:319   A :209
##  1:376   G :163
##          P :151
##          VG:114
##          VP: 58
##

#there is no missing values in all the variables.
stat.desc(Train_AP)

##           Cancer_AP        Age_AP         HW_AP Hst_AP        Exe_AP Smk_AP
## nbr.val          NA   695.0000000 695.00000000     NA   695.0000000     NA
## nbr.null         NA     0.0000000   0.00000000     NA     0.0000000     NA
## nbr.na           NA     0.0000000   0.00000000     NA     0.0000000     NA
## min              NA    22.0000000   0.67800000     NA     1.0000000     NA
## max              NA    70.0000000   8.50000000     NA   140.0000000     NA
## range            NA    48.0000000   7.82200000     NA   139.0000000     NA
## sum              NA 32731.0000000 722.46700000     NA 21858.0000000     NA
## median           NA    48.0000000   1.03100000     NA    26.0000000     NA
## mean             NA    47.0949640   1.03952086     NA    31.4503597     NA
## SE.mean          NA     0.3030735   0.01169660     NA     0.8477562     NA
## CI.mean          NA     0.5950510   0.02296496     NA     1.6644745     NA
## var              NA    63.8382311   0.09508318     NA   499.4899674     NA
## std.dev          NA     7.9898830   0.30835561     NA    22.3492722     NA
## coef.var         NA     0.1696547   0.29663244     NA     0.7106206     NA
##           Drk_AP Hlth_AP
## nbr.val       NA      NA
## nbr.null      NA      NA
## nbr.na        NA      NA
## min           NA      NA
## max           NA      NA
## range         NA      NA
## sum           NA      NA
## median        NA      NA
## mean         NA       NA
## SE.mean      NA       NA
## CI.mean      NA       NA
## var          NA       NA
## std.dev      NA       NA
## coef.var     NA       NA

#There is no low variance after inspection of Coef.var of all variables
#(for numeric).
```

Apply the High Correlation Filter to remove appropriate columns of data.

High correlation between two variables means they have similar trends and are likely to carry similar information.

No correlation available between numerical and nominal columns.

pearson, spearman,kendall methos can be used to measure the degree of association between two variables.

can only check for numerical and we have 3 column with numeric data so n(n-1)/2 (3*2/2 = 3) combination should be checked.

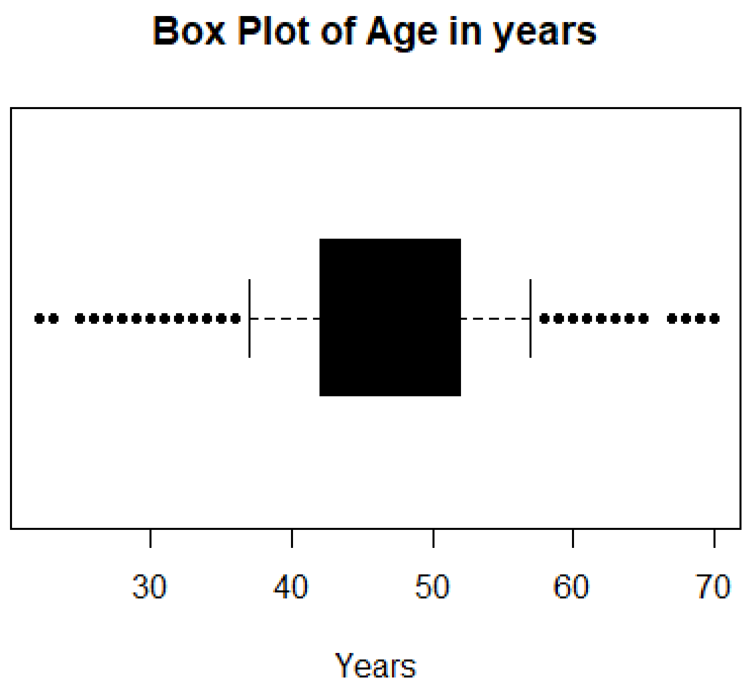
```
#by spearman method
#Speaman is non-parametric and therefore makes no normalacy assumption

cor(Train_AP$Age_AP,Train_AP$HW_AP,method = "spearman")

## [1] -0.1572284

cor(Train_AP$Age_AP,Train_AP$Exe_AP,method = "spearman")

## [1] 0.03219058

cor(Train_AP$HW_AP,Train_AP$Exe_AP,method = "spearman")

## [1] -0.04108865
```

Conclusion: There is no correlation between numeric variables. Moreover, 4 variables are binary so we cannot plot

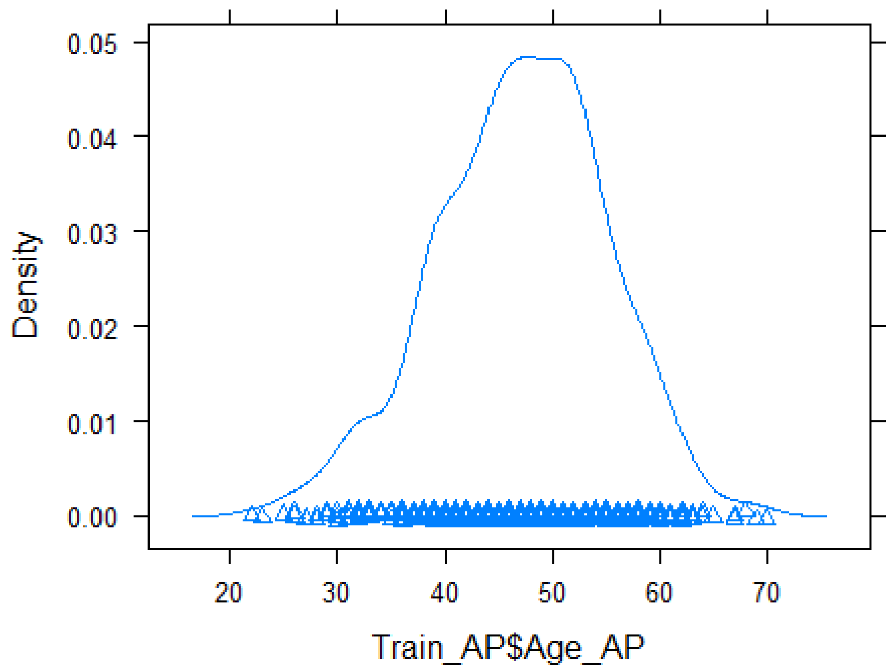
There no single variable which is not useful in analytic purpose.

Let’s check for the outliers.

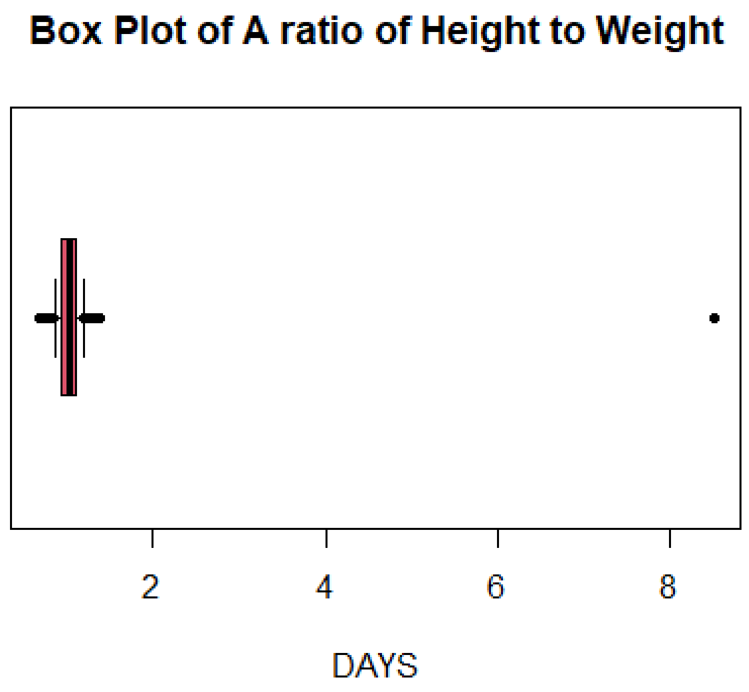
```
boxplot(Train_AP$Age_AP,
       main="Box Plot of Age in years",
        xlab="Years",
        col=1, horizontal=TRUE, pch=20, range = 0.5)
```



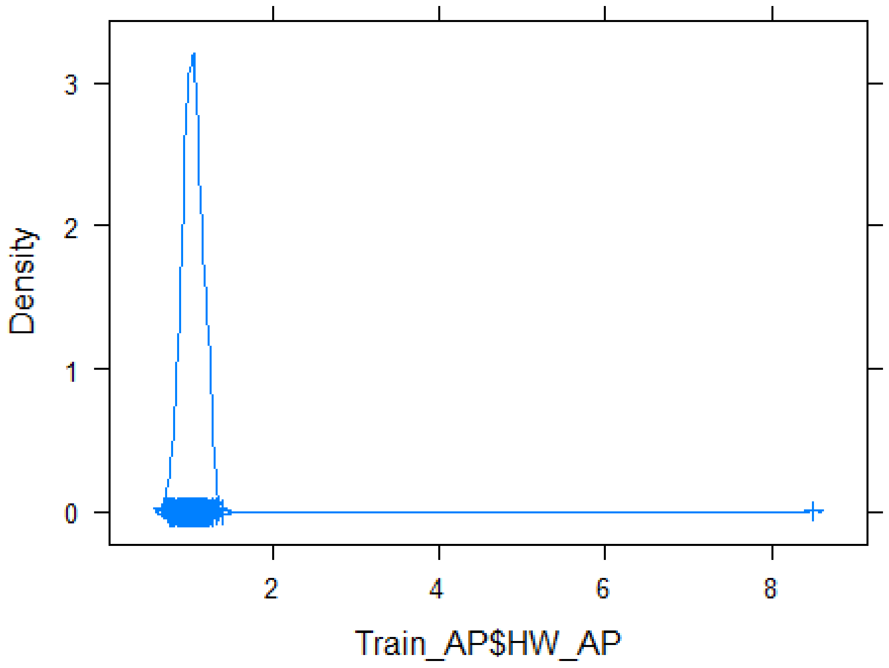
```
densityplot( ~ Train_AP$Age_AP, pch=2)
```



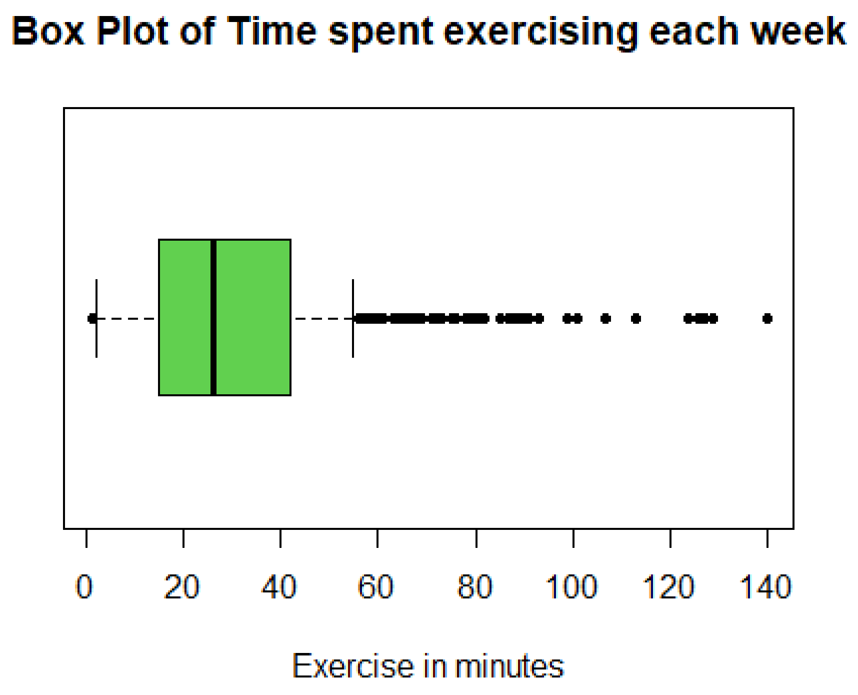
```r
boxplot(Train_AP$HW_AP,
       main="Box Plot of A ratio of Height to Weight",
        xlab="DAYS",
        col=2, horizontal=TRUE, pch=20, range = 0.5)
```



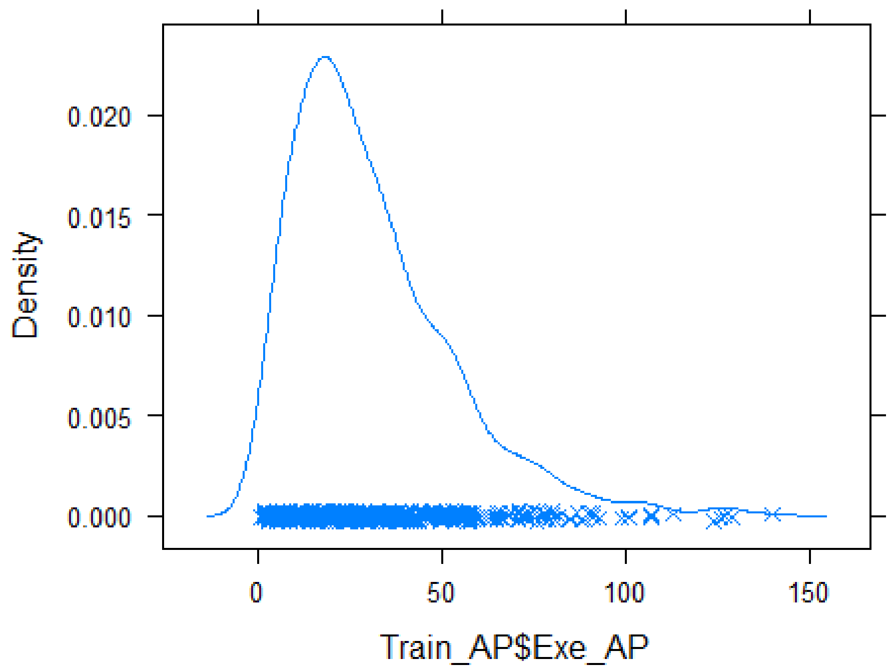
```r
densityplot( ~ Train_AP$HW_AP, pch=3)
```



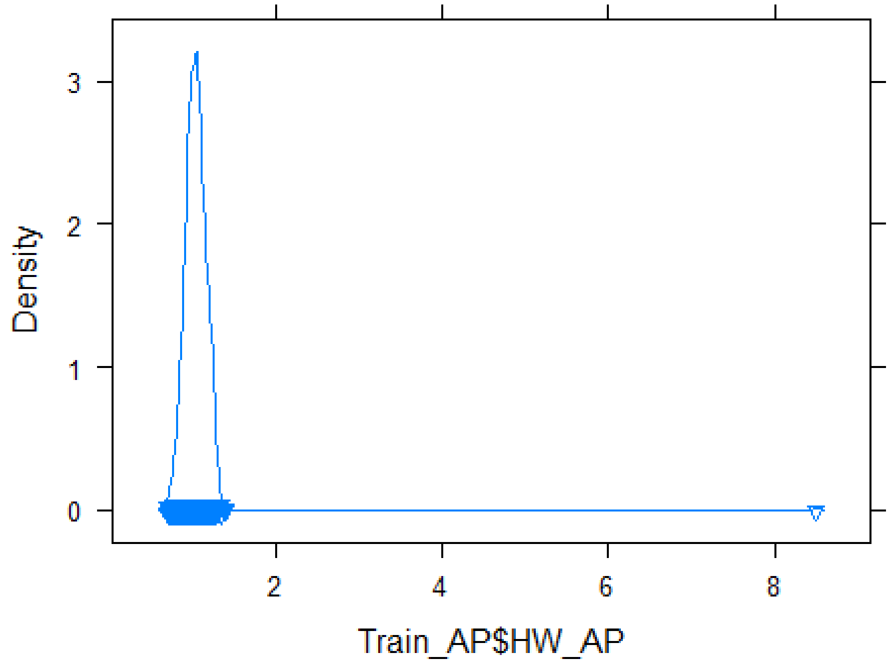
```
boxplot(Train_AP$Exe_AP,
       main="Box Plot of Time spent exercising each week",
        xlab="Exercise in minutes",
        col=3, horizontal=TRUE, pch=20, range = 0.5)
```



```
densityplot( ~ Train_AP$Exe_AP, pch=4)
```



```
densityplot( ~ Train_AP$HW_AP, pch=6)
```



```
nr <- which(Train_AP$HW_AP == max(Train_AP$HW_AP))#to see which row has max
value
nr

## [1] 257
```

From Density plot and boxplot we can say all the variables are seem fine. However, HW_AP has outlier which has value 8.5 but I am keep this variables as it is at this moment because it is possible that that person has height or weight more compared to each other so ratio is high.

#let's create glm model as our dependent variable is in binary so here we will apply logistic regression.

```
glmstart_time_AP <- Sys.time()
glm_AP <- glm(Train_AP$Cancer_AP ~ ., data=Train_AP, family = "binomial")
summary(glm_AP)

##
## Call:
## glm(formula = Train_AP$Cancer_AP ~ ., family = "binomial", data =
Train_AP)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.94946  -0.46807   0.05306   0.43762   2.83234
##
```

```
## Coefficients:
##                Estimate Std. Error z value        Pr(>|z|)    
## (Intercept)  10.132190   1.638007   6.186 0.0000000006183 ***
## Age_AP        0.056034   0.016007   3.501        0.000464 ***
## HW_AP       -14.018355   1.428407  -9.814         < 2e-16 ***
## Hst_AP1       1.777051   0.268118   6.628 0.0000000000341 ***
## Exe_AP       -0.006979   0.005379  -1.298        0.194417    
## Smk_AP1       2.976943   0.657854   4.525 0.0000060328439 ***
## Drk_AP1      -0.575130   0.661089  -0.870        0.384315    
## Hlth_APG      0.006291   0.321522   0.020        0.984389    
## Hlth_APP      0.194847   0.340547   0.572        0.567213    
## Hlth_APVG     0.022065   0.363307   0.061        0.951570    
## Hlth_APVP     0.416644   0.496110   0.840        0.401008    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 963.40  on 694  degrees of freedom
## Residual deviance: 459.55  on 684  degrees of freedom
## AIC: 481.55
## 
## Number of Fisher Scoring iterations: 6
```

```
glmend_time_AP <- Sys.time()
glm_Time_AP <- glmend_time_AP - glmstart_time_AP

str(Train_AP)
```

```
## tibble [695 × 8] (S3: tbl_df/tbl/data.frame)
##  $ Cancer_AP: Factor w/ 2 levels "0","1": 1 1 2 1 1 1 2 1 1 1 ...
##  $ Age_AP   : num [1:695] 43 39 49 59 45 53 69 50 40 50 ...
##  $ HW_AP    : num [1:695] 1.127 1.214 0.898 1.028 1.082 ...
##  $ Hst_AP   : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 2 1 1 1 ...
##  $ Exe_AP   : num [1:695] 53 38 18 9 127 11 8 7 51 11 ...
##  $ Smk_AP   : Factor w/ 2 levels "0","1": 1 1 2 2 1 1 2 1 1 1 ...
##  $ Drk_AP   : Factor w/ 2 levels "0","1": 1 1 2 2 1 1 2 1 1 1 ...
##  $ Hlth_AP  : Factor w/ 5 levels "A","G","P","VG",..: 4 1 3 3 4 1 3 1 1 3
...
```

```
#just experimenting
#glm1_AP <- glm(Train_AP$Cancer_AP ~ Age_AP + HW_AP + Hst_AP +
#   Smk_AP+Drk_AP, data=Train_AP, family = "binomial")
#summary(glm1_AP)
#---------
resp_glm_AP <- predict(glm_AP, type="response")
  Class_glm_AP <- ifelse(resp_glm_AP > 0.5,"1","0")
  CF_GLM_AP <- table(Train_AP$Cancer_AP, Class_glm_AP,
                dnn=list("Act ","Predicted") )
  CF_GLM_AP
```

```
##     Predicted
## Act    0   1
##    0 292  52
##    1  47 304
```

```
glmTP_AP <- CF_GLM_AP[2,2]
glmTN_AP <- CF_GLM_AP[1,1]
glmFP_AP <- CF_GLM_AP[1,2]
glmFN_AP <- CF_GLM_AP[2,1]

GLMAccuracy_AP <- (glmTP_AP + glmTN_AP) / 695
GLMAccuracy_AP
```

```
## [1] 0.857554
```

```
#let's see by applying backward

backstart_time_AP <- Sys.time()
glm_AP <- glm(Train_AP$Cancer_AP ~ ., data=Train_AP, family = "binomial")
Backstep.fit_AP <- step(glm_AP, direction = "backward",trace = 0)
backend_time_AP <- Sys.time()
back_Time_AP <- backend_time_AP - backstart_time_AP
summary(Backstep.fit_AP)
```

```
##
## Call:
## glm(formula = Train_AP$Cancer_AP ~ Age_AP + HW_AP + Hst_AP +
##     Smk_AP, family = "binomial", data = Train_AP)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -3.00105  -0.46973   0.05708   0.45025   2.86729
##
## Coefficients:
##              Estimate Std. Error z value         Pr(>|z|)
## (Intercept)  10.02415    1.61313   6.214 0.00000000051625 ***
## Age_AP        0.05456    0.01573   3.469         0.000523 ***
## HW_AP       -14.02958    1.42152  -9.869          < 2e-16 ***
## Hst_AP1       1.80988    0.26535   6.821 0.00000000000906 ***
## Smk_AP1       2.43806    0.24161  10.091          < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 963.4  on 694  degrees of freedom
## Residual deviance: 463.2  on 690  degrees of freedom
## AIC: 473.2
##
## Number of Fisher Scoring iterations: 6
```

```
#confusion matrix for backward

resp_glm_AP <- predict(Backstep.fit_AP, type="response")
  Class_glm_AP <- ifelse(resp_glm_AP > 0.5,"1","0")
  CF_bGLM_AP <- table(Train_AP$Cancer_AP, Class_glm_AP,
                dnn=list("Act ","Predicted") )
  CF_bGLM_AP

##      Predicted
## Act    0   1
##    0 292  52
##    1  46 305

BackTP_AP <- CF_bGLM_AP[2,2]
BackTN_AP <- CF_bGLM_AP[1,1]
BackFP_AP <- CF_bGLM_AP[1,2]
BackFN_AP <- CF_bGLM_AP[2,1]

BackAccuracy_AP <- (BackTP_AP + BackTN_AP) / 695
BackAccuracy_AP

## [1] 0.8589928

back_Time_AP

## Time difference of 0.157881 secs
```

Conclusion: I am selecting backward model as a the best model because it has low AIC all the variables are significant, residuals are symmetrical. However, there is slightly high differnece between NUll and residuals deviance compared to normal glm model but from other factors backward model is the best.

Naïve-Bayes Classification

```
NBstart_time_AP <- Sys.time()

NB.fit_AP <- NaiveBayes(Train_AP$Cancer_AP ~ . ,data = Train_AP,
na.action=na.omit)

NBend_time_AP <- Sys.time()

NB_Time_AP <- NBend_time_AP - NBstart_time_AP

NB_Time_AP

## Time difference of 0.005084038 secs

pred_bay_AP <- predict(NB.fit_AP,Train_AP)

#Creates Confusion Matrix
```

```
CF_NB_AP <- table(Actual=Train_AP$Cancer_AP, Predicted=pred_bay_AP$class)

#Confusion matrix of Naïve-Bayesian classification.

CF_NB_AP

##       Predicted
## Actual   0   1
##      0 263  81
##      1  55 296

NB_TP_AP <- CF_NB_AP[2,2]
NB_TN_AP <- CF_NB_AP[1,1]
NB_FP_AP <- CF_NB_AP[1,2]
NB_FN_AP <- CF_NB_AP[2,1]

NBAccuracy_AP <- (NB_TP_AP + NB_TN_AP) / 695
NBAccuracy_AP

## [1] 0.8043165
```

Linear Discriminant Analysis

```
LDAstart_time_AP <- Sys.time()

LDA.fit_AP <- lda(Train_AP$Cancer_AP ~ ., data = Train_AP, na.action=na.omit)

LDAend_time_AP <- Sys.time()

LDA_Time_AP <- LDAend_time_AP - LDAstart_time_AP

LDA_Time_AP

## Time difference of 0.008917093 secs

#Predicting LDA Model

LDApred_AP <- predict(LDA.fit_AP, data=Train_AP)

#Confusion matrix for LDA Model.

CF_LDA_AP <- table(Actual=Train_AP$Cancer_AP, Predicted=LDApred_AP$class)

CF_LDA_AP

##       Predicted
## Actual   0   1
##      0 272  72
##      1  63 288
```

```
LDA_TP_AP <- CF_LDA_AP[2,2]
LDA_TN_AP <- CF_LDA_AP[1,1]
LDA_FP_AP <- CF_LDA_AP[1,2]
LDA_FN_AP <- CF_LDA_AP[2,1]

LDA_Accuracy_AP <- (LDA_TP_AP + LDA_TN_AP) / 695
LDA_Accuracy_AP

## [1] 0.8057554
```

Conclusion: here we can see the backward model has the lowest false positives, false negatives, and high accuracy. However, model processing time is high compared to others but I am choosing backward model as the best model.

For Test data

```
Test_AP <- read_excel("PROG8430_Final_22F_test.xlsx")

head(Test_AP)

## # A tibble: 6 × 7
##     Age    HW Hst     Exe Smk   Drk   Hlth 
##   <dbl> <dbl> <chr> <dbl> <chr> <chr> <chr>
## 1    43 0.99  0        19 0     0     VG   
## 2    50 0.847 0        29 1     1     A    
## 3    62 0.792 0        20 1     1     G    
## 4    44 0.857 0         5 0     0     VG   
## 5    52 1.06  1       149 0     0     P    
## 6    55 0.963 0        10 0     0     P    

str(Test_AP)

## tibble [705 × 7] (S3: tbl_df/tbl/data.frame)
##  $ Age : num [1:705] 43 50 62 44 52 55 43 51 50 41 ...
##  $ HW  : num [1:705] 0.99 0.847 0.792 0.857 1.065 ...
##  $ Hst : chr [1:705] "0" "0" "0" "0" ...
##  $ Exe : num [1:705] 19 29 20 5 149 10 21 31 19 15 ...
##  $ Smk : chr [1:705] "0" "1" "1" "0" ...
##  $ Drk : chr [1:705] "0" "1" "1" "0" ...
##  $ Hlth: chr [1:705] "VG" "A" "G" "VG" ...

#initials

colnames(Test_AP) <- paste(colnames(Test_AP), "AP", sep = "_")
head(Test_AP)

## # A tibble: 6 × 7
##   Age_AP HW_AP Hst_AP Exe_AP Smk_AP Drk_AP Hlth_AP
##    <dbl> <dbl> <chr>   <dbl> <chr>  <chr>  <chr>  
## 1     43 0.99  0          19 0      0      VG     
## 2     50 0.847 0          29 1      1      A      
## 3     62 0.792 0          20 1      1      G
## 4     44 0.857 0           5 0      0      VG
## 5     52 1.06  1         149 0      0      P
## 6     55 0.963 0          10 0      0      P
```

```
str(Test_AP)
```

```
## tibble [705 × 7] (S3: tbl_df/tbl/data.frame)
##  $ Age_AP : num [1:705] 43 50 62 44 52 55 43 51 50 41 ...
##  $ HW_AP  : num [1:705] 0.99 0.847 0.792 0.857 1.065 ...
##  $ Hst_AP : chr [1:705] "0" "0" "0" "0" ...
##  $ Exe_AP : num [1:705] 19 29 20 5 149 10 21 31 19 15 ...
##  $ Smk_AP : chr [1:705] "0" "1" "1" "0" ...
##  $ Drk_AP : chr [1:705] "0" "1" "1" "0" ...
##  $ Hlth_AP: chr [1:705] "VG" "A" "G" "VG" ...
```

```
# converting characters into the factors

Test_AP$Hst_AP <- as.factor(Test_AP$Hst_AP)
Test_AP$Smk_AP <- as.factor(Test_AP$Smk_AP)
Test_AP$Drk_AP <- as.factor(Test_AP$Drk_AP)
Test_AP$Hlth_AP <- as.factor(Test_AP$Hlth_AP)

str(Test_AP)
```

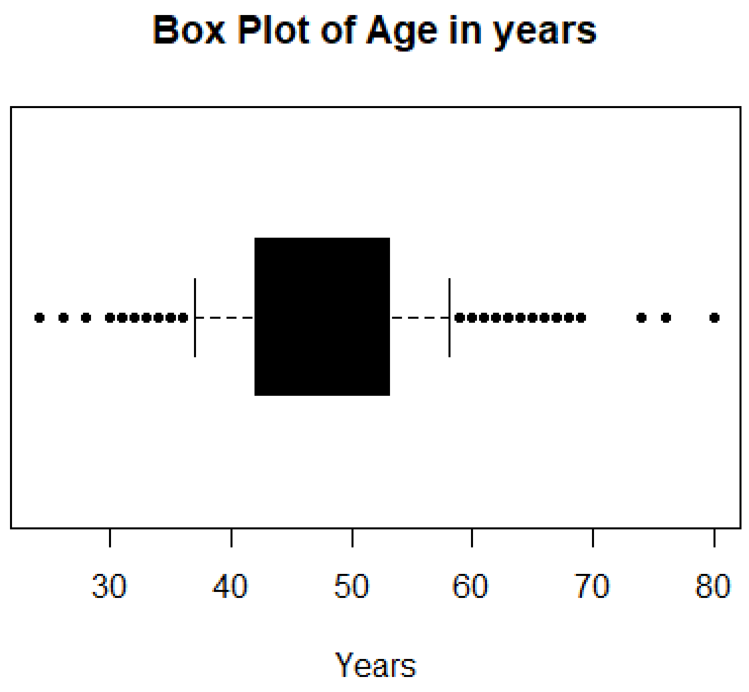
```
## tibble [705 × 7] (S3: tbl_df/tbl/data.frame)
##  $ Age_AP : num [1:705] 43 50 62 44 52 55 43 51 50 41 ...
##  $ HW_AP  : num [1:705] 0.99 0.847 0.792 0.857 1.065 ...
##  $ Hst_AP : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 1 1 1 ...
##  $ Exe_AP : num [1:705] 19 29 20 5 149 10 21 31 19 15 ...
##  $ Smk_AP : Factor w/ 2 levels "0","1": 1 2 2 1 1 1 1 1 1 1 ...
##  $ Drk_AP : Factor w/ 2 levels "0","1": 1 2 2 1 1 1 1 1 2 1 ...
##  $ Hlth_AP: Factor w/ 5 levels "A","G","P","VG",..: 4 1 2 4 3 3 4 3 1 4
...
```

```
summary(Test_AP)
```

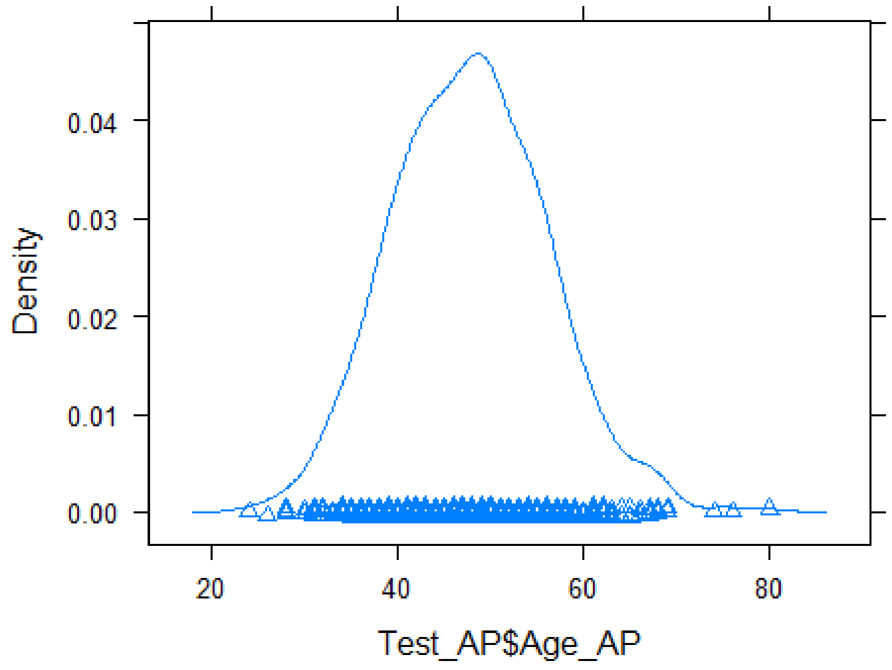
```
##      Age_AP          HW_AP         Hst_AP      Exe_AP       Smk_AP   Drk_AP 
##  Min.   :24.00   Min.   :0.654   0:465   Min.   :  0.0   0:345   0:324  
##  1st Qu.:42.00   1st Qu.:0.922   1:240   1st Qu.: 15.0   1:360   1:381  
##  Median :48.00   Median :1.024           Median : 25.0                  
##  Mean   :47.79   Mean   :1.021           Mean   : 30.2                  
##  3rd Qu.:53.00   3rd Qu.:1.108           3rd Qu.: 41.0                  
##  Max.   :80.00   Max.   :1.356           Max.   :149.0                  
##  Hlth_AP
##  A :198 
##  G :177 
##  P :138 
##  VG:119 
##  VP: 73 
## 
```

```
#there is no missing values in all the variables.
stat.desc(Test_AP)

##                     Age_AP         HW_AP Hst_AP        Exe_AP Smk_AP
Drk_AP
## nbr.val        705.0000000 705.000000000     NA   705.0000000     NA
NA
## nbr.null         0.0000000   0.000000000     NA     1.0000000     NA
NA
## nbr.na           0.0000000   0.000000000     NA     0.0000000     NA
NA
## min             24.0000000   0.654000000     NA     0.0000000     NA
NA
## max             80.0000000   1.356000000     NA   149.0000000     NA
NA
## range           56.0000000   0.702000000     NA   149.0000000     NA
NA
## sum          33689.0000000 719.929000000     NA 21294.0000000     NA
NA
## median          48.0000000   1.024000000     NA    25.0000000     NA
NA
## mean            47.7858156   1.021175887     NA    30.2042553     NA
NA
## SE.mean          0.3116863   0.004901292     NA     0.8260822     NA
NA
## CI.mean.0.95     0.6119459   0.009622900     NA     1.6218796     NA
NA
## var             68.4895712   0.016935980     NA   481.1002660     NA
NA
## std.dev          8.2758426   0.130138313     NA    21.9339979     NA
NA
## coef.var         0.1731862   0.127439665     NA     0.7261890     NA
NA
##              Hlth_AP
## nbr.val           NA
## nbr.null          NA
## nbr.na            NA
## min               NA
## max               NA
## range             NA
## sum               NA
## median            NA
## mean              NA
## SE.mean           NA
## CI.mean.0.95      NA
## var               NA
## std.dev           NA
## coef.var          NA
```

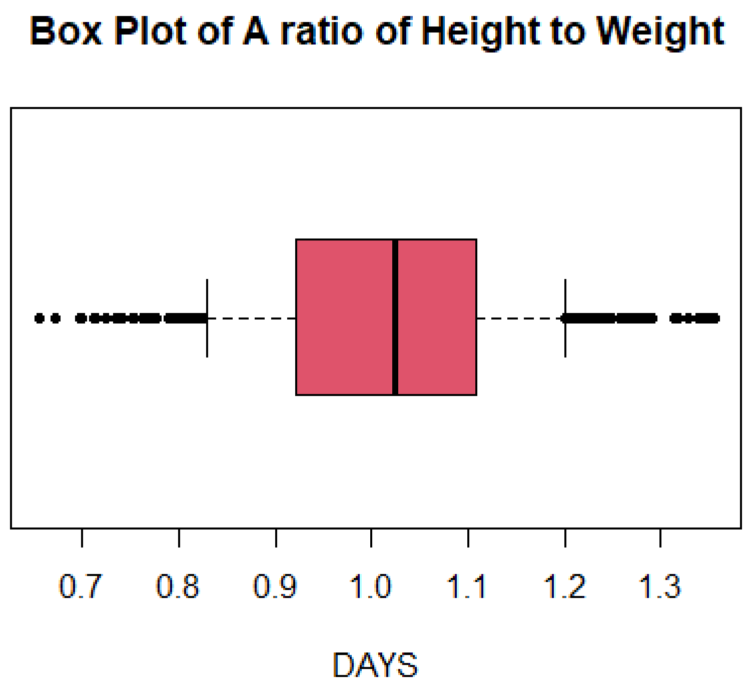
```
#There is no low variance after inspection of Coef.var of all variables
#(for numeric).
#by spearman method
#Speaman is non-parametric and therefore makes no normalacy assumption

cor(Train_AP$Age_AP,Train_AP$HW_AP,method = "spearman")

## [1] -0.1572284

cor(Train_AP$Age_AP,Train_AP$Exe_AP,method = "spearman")

## [1] 0.03219058

cor(Train_AP$HW_AP,Train_AP$Exe_AP,method = "spearman")

## [1] -0.04108865

#There is no correlation between numeric variables

boxplot(Test_AP$Age_AP,
       main="Box Plot of Age in years",
        xlab="Years",
        col=1, horizontal=TRUE, pch=20, range = 0.5)
```



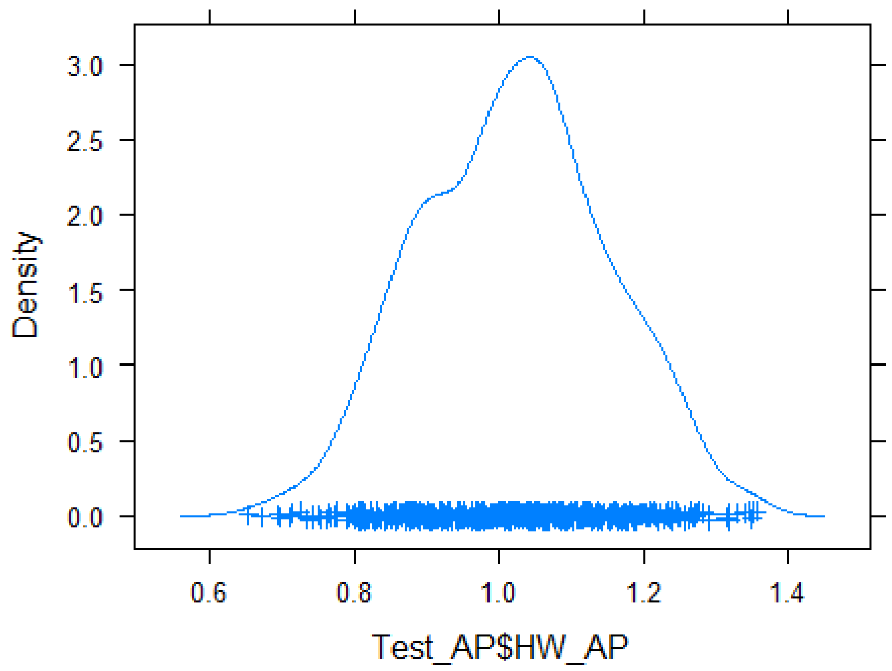
```
densityplot( ~ Test_AP$Age_AP, pch=2)
```



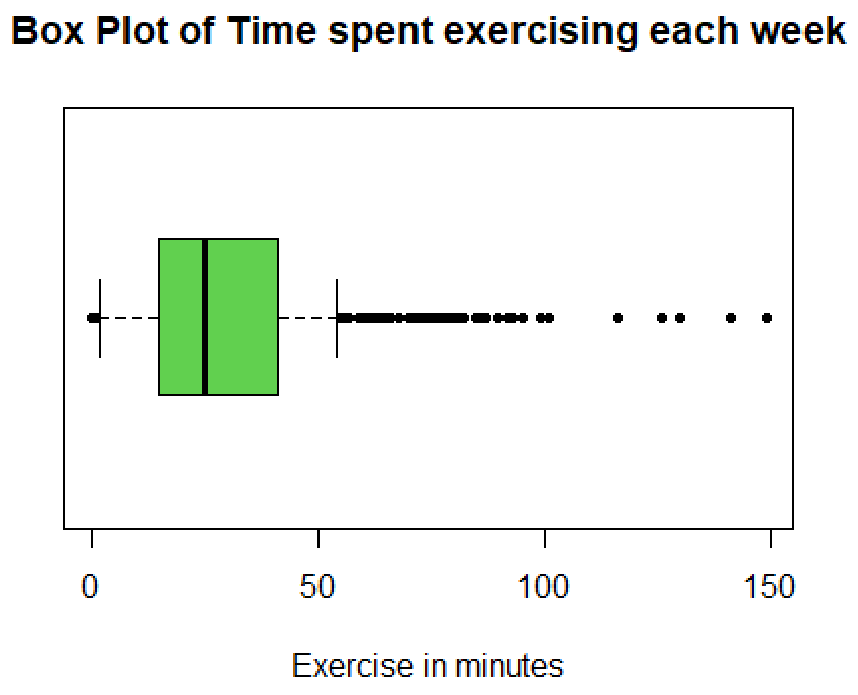
```
boxplot(Test_AP$HW_AP,
       main="Box Plot of A ratio of Height to Weight",
        xlab="DAYS",
        col=2, horizontal=TRUE, pch=20, range = 0.5)
```



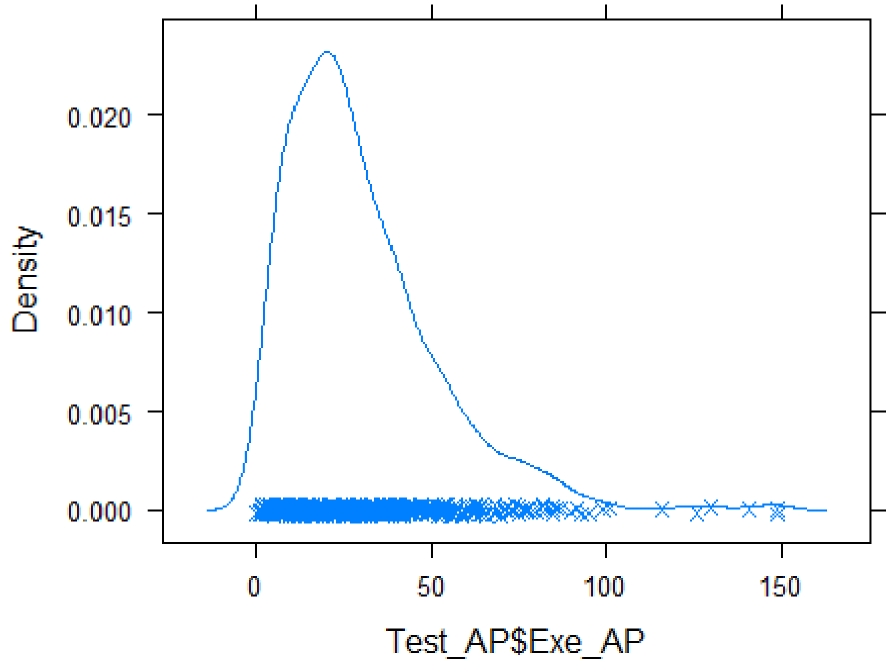
```
densityplot( ~ Test_AP$HW_AP, pch=3)
```



```r
boxplot(Test_AP$Exe_AP,
       main="Box Plot of Time spent exercising each week",
        xlab="Exercise in minutes",
        col=3, horizontal=TRUE, pch=20, range = 0.5)
```



```r
densityplot( ~ Test_AP$Exe_AP, pch=4)
```



From above graphs,

All variables seem reasonable.

```r
if(!require(writexl)){install.packages("writexl")}

## Loading required package: writexl

library("writexl")

pred_AP <- predict(Backstep.fit_AP, newdata=Test_AP)
head(pred_AP)

##          1          2          3          4          5          6
## -1.5189399  3.3072872  4.7336655  0.4015572 -0.2702163 -0.4853899

test_fin_AP <- cbind(Test_AP,pred_AP)

write_xlsx(test_fin_AP,"PROG8430_22F_Final_Pred_AP.xlsx")
```

References:

David Marsh.(2022).[PROG8430-L10-22F].eConestoga.

David Marsh.(2022).[PROG8430-L11-22F].eConestoga.

David Marsh.(2022).[PROG8430-L12-22F].eConestoga.

David Marsh.(2022).[R Documents].eConestoga.