About Dataset (copy pasted)

Data Set Information: The dataset is Electronic Health Record Predicting collected from a private Hospital in Indonesia. It contains the patients laboratory test results used to determine next patient treatment whether in care or out care patient. The task embedded to the dataset is classification prediction. Attribute Information: Given is the attribute name, attribute type, the measurement unit and a brief description. The number of rings is the value to predict: either as a continuous value or as a classification problem.

Name / Data Type / Value Sample/ Description————————————

HAEMATOCRIT /Continuous /35.1 / Patient laboratory test result of haematocrit
HAEMOGLOBINS/Continuous/11.8 / Patient laboratory test result of haemoglobins
ERYTHROCYTE/Continuous/4.65 / Patient laboratory test result of erythrocyte
LEUCOCYTE/Continuous /6.3 / Patient laboratory test result of leucocyte
THROMBOCYTE/Continuous/310/ Patient laboratory test result of thrombocyte
MCH/Continuous /25.4/ Patient laboratory test result of MCH
MCHC/Continuous/33.6/ Patient laboratory test result of MCHC
MCV/Continuous /75.5/ Patient laboratory test result of MCV
AGE/Continuous/12/ Patient age SEX/Nominal – Binary/F/ Patient gender
SOURCE/Nominal/ {in,out}/The class target in.= in care patient, out = out care patient

First, all the libraries used were defined, and then we read the CSV file after setting the working directory

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.4
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```
library(ggplot2)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths

library(leaps)
library(caret)

## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift

library(MASS)

##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select

library("pROC")

## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

df <- read_csv(data2)

## Rows: 4412 Columns: 11
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr (2): SEX, SOURCE
## dbl (9): HAEMATOCRIT, HAEMOGLOBINS, ERYTHROCYTE, LEUCOCYTE, THROMBOCYTE,
MCH...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this
message.
```

No Na values are present in the file since the dim function gave the same output before and after calling the drop_na function. The aim of removing rows with missing values is to ensure our data is complete, avoid errors and improve the quality of our data.

```r
dim(df)
```

```
## [1] 4412   11
```

```r
df <- drop_na(df)
dim(df)
```

```
## [1] 4412   11
```

Here is an overview of the predictors and the response variable “SOURCE” we are working with

```r
colnames(df)
```

```
##  [1] "HAEMATOCRIT"  "HAEMOGLOBINS" "ERYTHROCYTE"  "LEUCOCYTE"
"THROMBOCYTE"
##  [6] "MCH"          "MCHC"         "MCV"          "AGE"          "SEX"
## [11] "SOURCE"
```

All predictors are of numerical type, except for ‘SEX’ and ‘SOURCE’, which are of character type. They will be converted to factors since they represent categorical variables.

```r
str(df)
```

```
## tibble [4,412 × 11] (S3: tbl_df/tbl/data.frame)
##  $ HAEMATOCRIT : num [1:4412] 35.1 43.5 33.5 39.1 30.9 34.3 31.1 40.3 33.6
35.4 ...
##  $ HAEMOGLOBINS: num [1:4412] 11.8 14.8 11.3 13.7 9.9 11.6 8.7 13.3 11.5
11.4 ...
##  $ ERYTHROCYTE : num [1:4412] 4.65 5.39 4.74 4.98 4.23 4.53 5.06 4.73 4.54
4.8 ...
##  $ LEUCOCYTE   : num [1:4412] 6.3 12.7 13.2 10.5 22.1 6.6 11.1 8.1 11.4
2.6 ...
##  $ THROMBOCYTE : num [1:4412] 310 334 305 366 333 185 416 257 262 183 ...
##  $ MCH         : num [1:4412] 25.4 27.5 23.8 27.5 23.4 25.6 17.2 28.1 25.3
23.8 ...
##  $ MCHC        : num [1:4412] 33.6 34 33.7 35 32 33.8 28 33 34.2 32.2 ...
##  $ MCV         : num [1:4412] 75.5 80.7 70.7 78.5 73 75.7 61.5 85.2 74
73.8 ...
##  $ AGE         : num [1:4412] 1 1 1 1 1 1 1 1 1 1 ...
##  $ SEX         : chr [1:4412] "F" "F" "F" "F" ...
##  $ SOURCE      : chr [1:4412] "out" "out" "out" "out" ...
```

Since the data description does not provide the range of values, we rely on the summary function to extract the range (min and max) Consequently, we are not be able to determine if any predictor represents out-of-range values.

```r
summary(df)
```

```
##   HAEMATOCRIT     HAEMOGLOBINS    ERYTHROCYTE      LEUCOCYTE
##  Min.   :13.70   Min.   : 3.80   Min.   :1.480   Min.   : 1.100
##  1st Qu.:34.38   1st Qu.:11.40   1st Qu.:4.040   1st Qu.: 5.675
```

```
##  Median :38.60   Median :12.90   Median :4.570   Median : 7.600
##  Mean   :38.20   Mean   :12.74   Mean   :4.541   Mean   : 8.719
##  3rd Qu.:42.50   3rd Qu.:14.20   3rd Qu.:5.050   3rd Qu.:10.300
##  Max.   :69.00   Max.   :18.90   Max.   :7.860   Max.   :76.600
##   THROMBOCYTE          MCH             MCHC            MCV
##  Min.   :   8.0   Min.   :14.90   Min.   :26.00   Min.   : 54.00
##  1st Qu.: 188.0   1st Qu.:27.20   1st Qu.:32.70   1st Qu.: 81.50
##  Median : 256.0   Median :28.70   Median :33.40   Median : 85.40
##  Mean   : 257.5   Mean   :28.23   Mean   :33.34   Mean   : 84.61
##  3rd Qu.: 321.0   3rd Qu.:29.80   3rd Qu.:34.10   3rd Qu.: 88.70
##  Max.   :1183.0   Max.   :40.80   Max.   :39.00   Max.   :115.60
##       AGE            SEX              SOURCE
##  Min.   : 1.00   Length:4412        Length:4412
##  1st Qu.:29.00   Class :character   Class :character
##  Median :47.00   Mode  :character   Mode  :character
##  Mean   :46.63
##  3rd Qu.:64.00
##  Max.   :99.00
```

We did not detect any duplicates in our dataset. However, if duplicates were present, it is advisable to remove them, as they can introduce correlated error terms, leading to an undeserved sense of confidence in our model in subsequent analyses. Then we view our dataset with the View() function

```
df[duplicated(df), ]
```

```
## # A tibble: 0 × 11
## # ℹ 11 variables: HAEMATOCRIT <dbl>, HAEMOGLOBINS <dbl>, ERYTHROCYTE
<dbl>,
## #   LEUCOCYTE <dbl>, THROMBOCYTE <dbl>, MCH <dbl>, MCHC <dbl>, MCV <dbl>,
## #   AGE <dbl>, SEX <chr>, SOURCE <chr>
```

```
View(df)
```

There are no sentinel or unexpected random values observed in these two predictors. Concerning the other numerical predictors, the summary function indicates the absence of unusual negative values in the minimum, and the maximum values appear to be within logical ranges.

```
unique(df$SEX)
```

```
## [1] "F" "M"
```

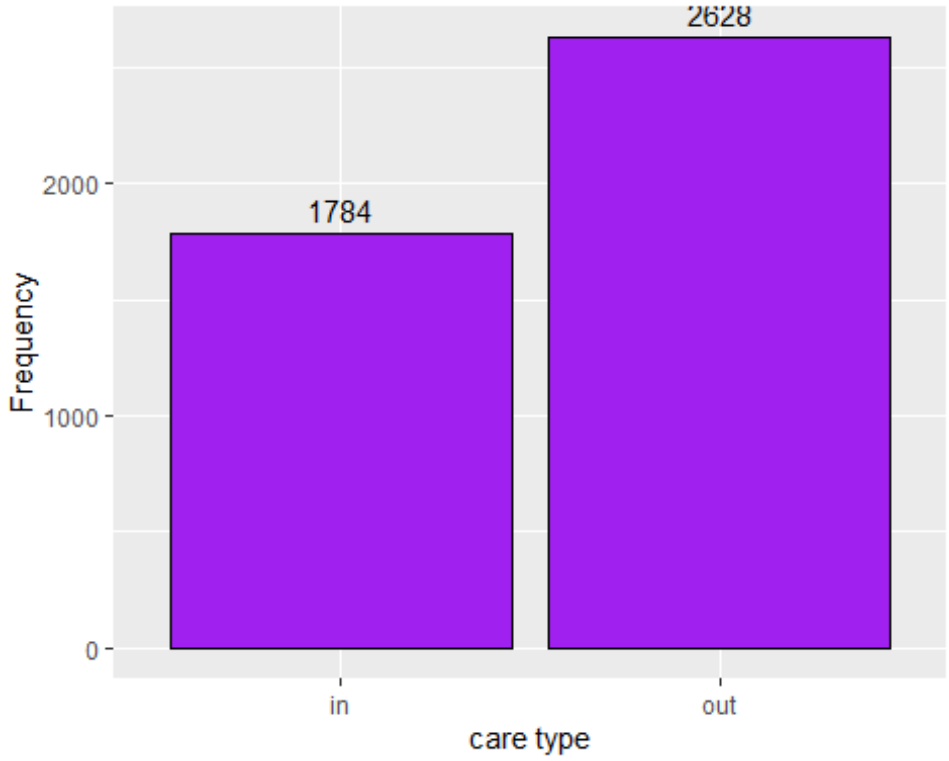
```
unique(df$SOURCE)
```

```
## [1] "out" "in"
```

The purpose of this plot is to check if the classes are balanced, if not, a specific classifier and a null classifier can give similar values. Approximately 40.4% of patients fall under ‘in care’ category, while around 59.6% are ‘out of care’ patients. Therefore, the data exhibits a relatively balanced distribution between the classes.

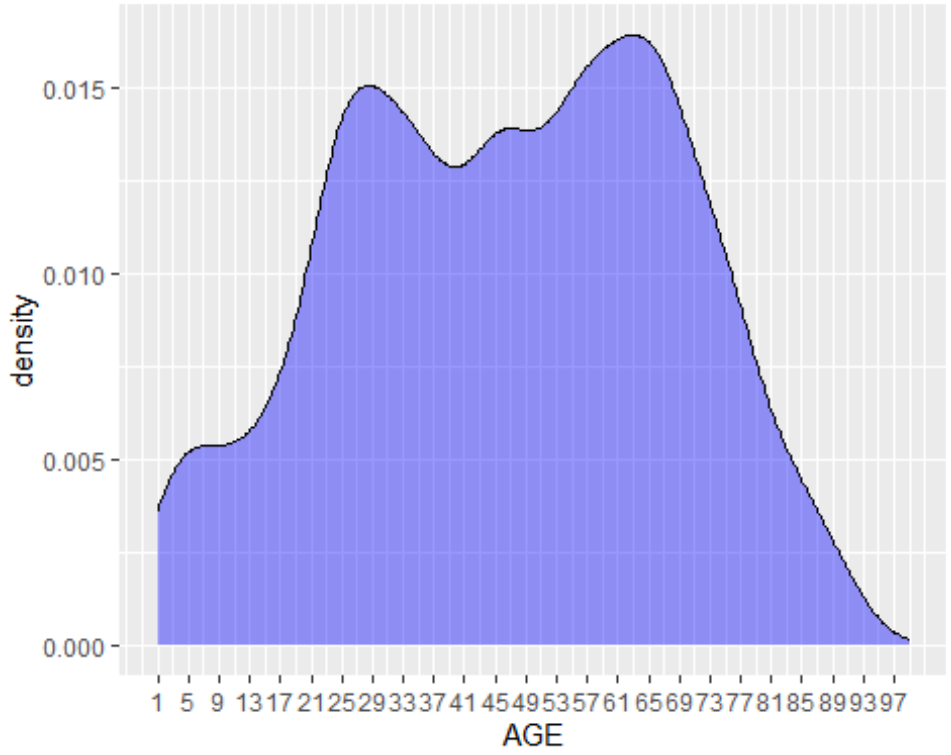
```r
ggplot(data = df, aes(x = SOURCE)) +
  geom_bar(fill = "purple", color = "black") +
  geom_text(stat = 'count', aes(label = ..count..), vjust = -0.5) +
  labs(x = "care type", y = "Frequency")
```

```
## Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2
3.4.0.
## ℹ Please use `after_stat(count)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



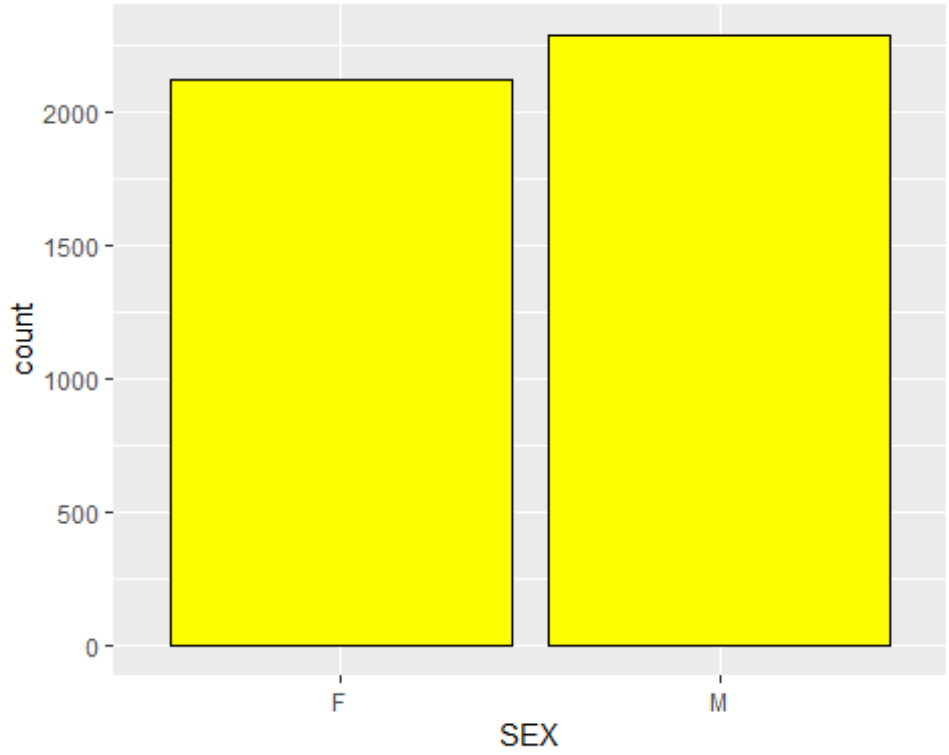
The majority of individuals in our dataset fall within the age range of 25 to 65.

```r
ggplot(data = df, aes(x = AGE)) +
  geom_density(fill = "blue", color = "black", alpha = 0.4) +
  scale_x_continuous(breaks = seq(1, 99, by = 4))
```



We have a nearly equal number of male and female patients in our dataset.

```
ggplot(data = df, aes(x = SEX)) +
  geom_bar(fill = "yellow", color = "black")
```



We found correlation between these predictors only

```
cor(df$HAEMATOCRIT,df$HAEMOGLOBINS)

## [1] 0.9732674

cor(df$HAEMATOCRIT,df$ERYTHROCYTE)

## [1] 0.8649885

cor(df$HAEMOGLOBINS,df$ERYTHROCYTE)

## [1] 0.8180128

cor(df$MCH,df$MCHC)

## [1] 0.5898305

cor(df$MCH,df$MCV)

## [1] 0.9318043

df$SOURCE <- as.factor(df$SOURCE)
df$SEX <- as.factor(df$SEX)
```

Subset selection We conduct best, forward and backward subset selection methods to determine the predictors chosen by each based on adjusted R^2, Cp, and BIC criteria

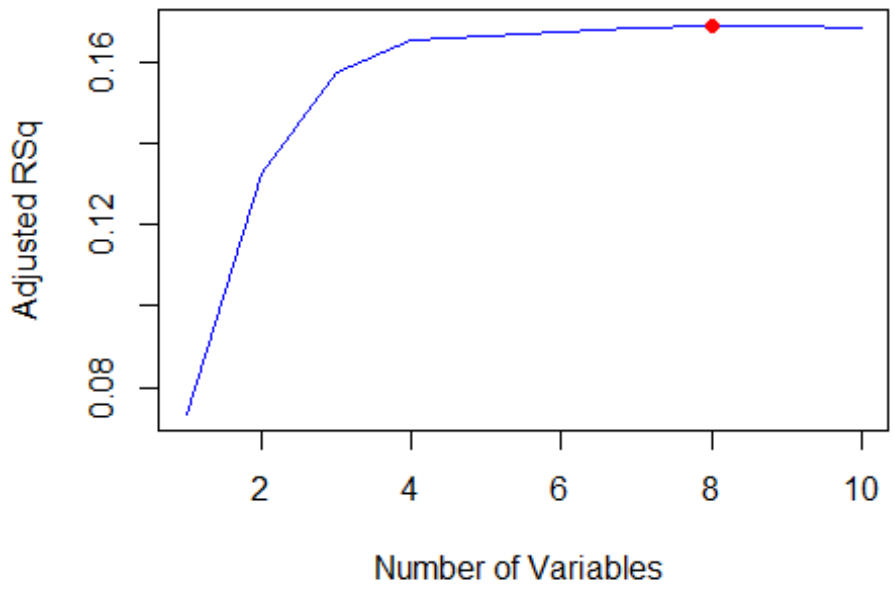
1- Best Subset Selection

```
regfit.full <- regsubsets(SOURCE ~ . ,df,nvmax=10)
reg.summary <- summary (regfit.full)
```

Based on these two graphs, adjusted R^2 selects the eight-variable model containing the predictors : “ERYTHROCYTE”, “LEUCOCYTE”, “THROMBOCYTE”, “MCH”, “MCHC”, “MCV”,“AGE” and “SEX”

It’s worth noting that in this graph, the curve remains quite flat after the rapid increase in adjusted R^2 at the beginning of the graph, indicating minimal differences in accuracy between models with eight variables and those having fewer predictors, e.g a model with four-variable.

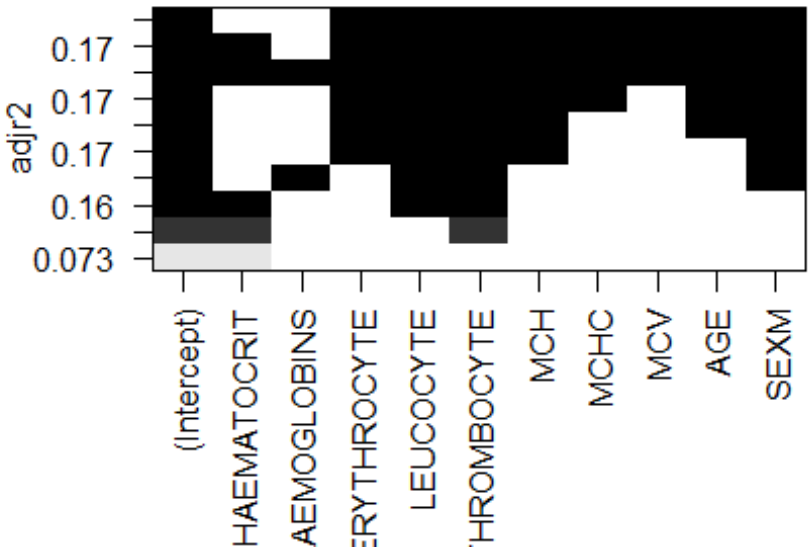
```
plot(reg.summary$adjr2 ,xlab="Number of Variables ",ylab="Adjusted
RSq",type="l",col='blue')
max_index <- which.max(reg.summary$adjr2)
max_adjr2 <- reg.summary$adjr2[max_index]
points(max_index, max_adjr2, col = "red", pch = 16)
```




```
coef(regfit.full ,max_index)
```

```
##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118
```

```r
plot(regfit.full ,scale="adjr2")
```

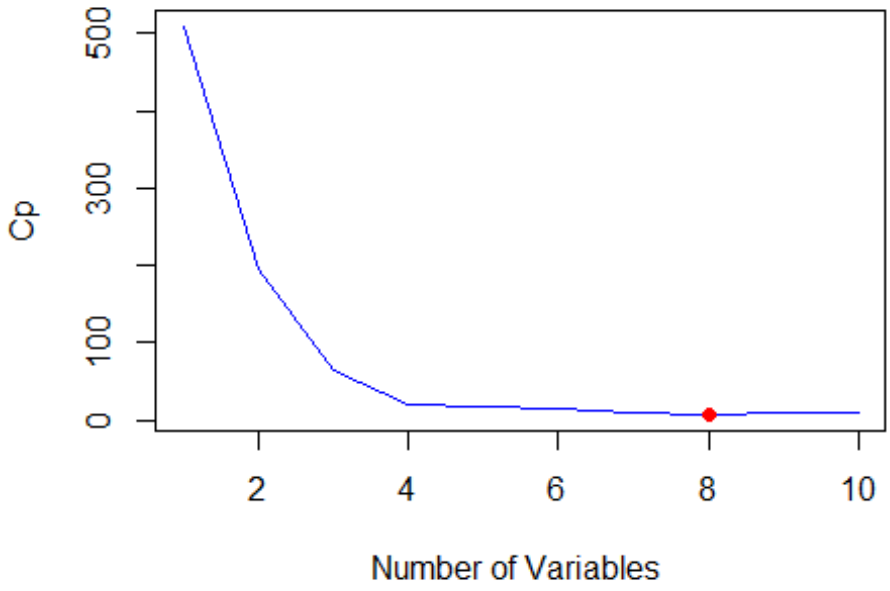



```r
coef(regfit.full ,8)
```

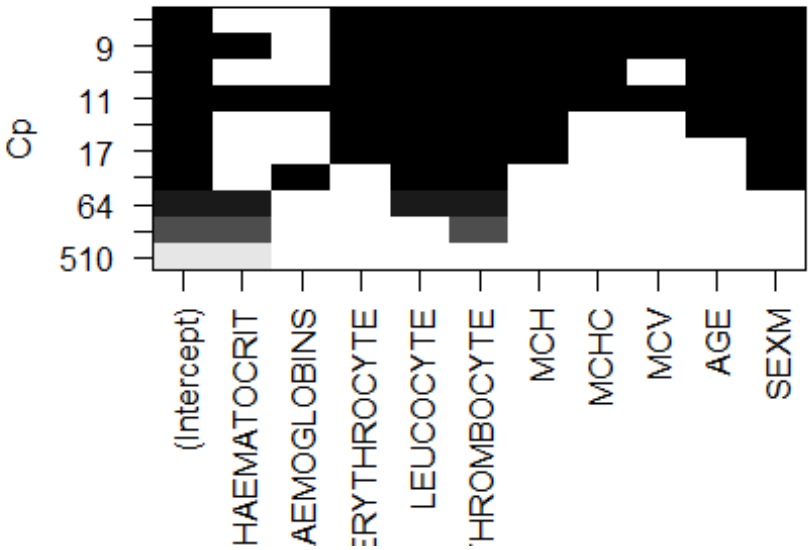
```
##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118
```

Cp selects the same predictors as Adjusted R^2 Similarly after the rapid decrease in Cp at the beginning of the graph, the curve remains quite flat, indicating minimal differences in accuracy between models with eight variables and those having fewer predictors, e.g a four-variable model.

```r
plot(reg.summary$cp ,xlab="Number of Variables
",ylab="Cp",type="l",col='blue')
min_index <- which.min(reg.summary$cp)
min_cp <- reg.summary$cp[min_index]
points(min_index, min_cp, col = "red", pch = 16)
```



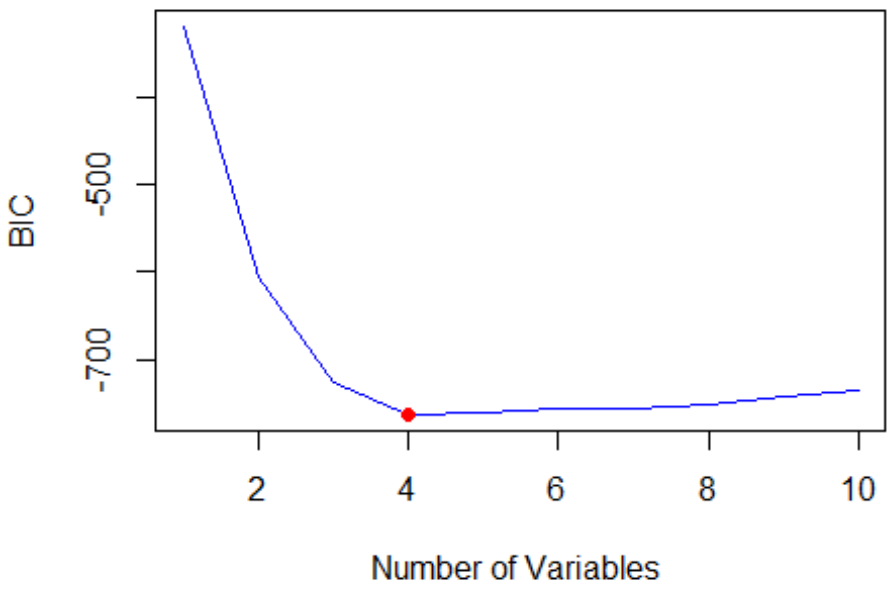

```
coef(regfit.full ,min_index)

##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118

plot(regfit.full ,scale="Cp")
```




```
coef(regfit.full ,8)

##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118
```

Using BIC results in the selection of a model that contains four variables: “HAEMOGLOBINS”, “LEUCOCYTE”, “THROMBOCYTE”, “SEX”
This specific model (four_variable model) represents the lowest point on the plot, with higher points both before and after it.

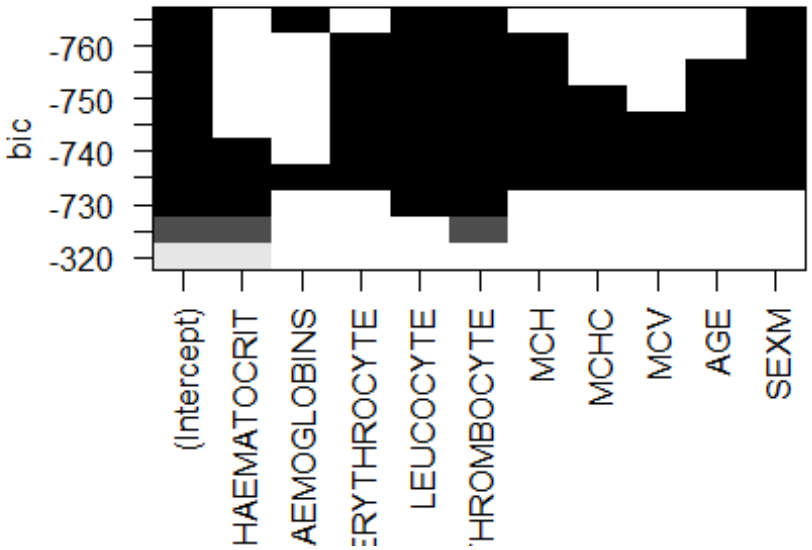
```
plot(reg.summary$bic ,xlab="Number of Variables
",ylab="BIC",type="l",col='blue')
min_index <- which.min(reg.summary$bic)
min_bic <- reg.summary$bic[min_index]
points(min_index, min_bic, col = "red", pch = 16)
```




```
coef(regfit.full ,min_index)
```

```
##  (Intercept) HAEMOGLOBINS    LEUCOCYTE  THROMBOCYTE         SEXM
##  0.654691935  0.063104925 -0.015470257  0.001258035 -0.100580490
```

```
plot(regfit.full ,scale="bic")
```




```
coef(regfit.full ,4)
```

```
##  (Intercept) HAEMOGLOBINS    LEUCOCYTE  THROMBOCYTE         SEXM
##  0.654691935  0.063104925 -0.015470257  0.001258035 -0.100580490
```

In the subsequent code, we showcase the predictors selected through best subset selection in each step of the algorithm, alongside the corresponding values of Adjusted R^2, Cp, and BIC at every stage. (we will do similarly for forward and backward subset selection)

Our graphical representations, previously depicted, are now translated into numerical insights. Notably, the adjusted R^2 exhibits marginal increments in models comprising more than four variables, ranging from 0.16585465 to a maximum of 0.16904373, with a difference of 0.0031.

Similarly, the Cp metric experiences a slight decrease following a four-variable model, dropping from 20.911412 to a minimum of 7.385411, indicating a difference of 13.53.

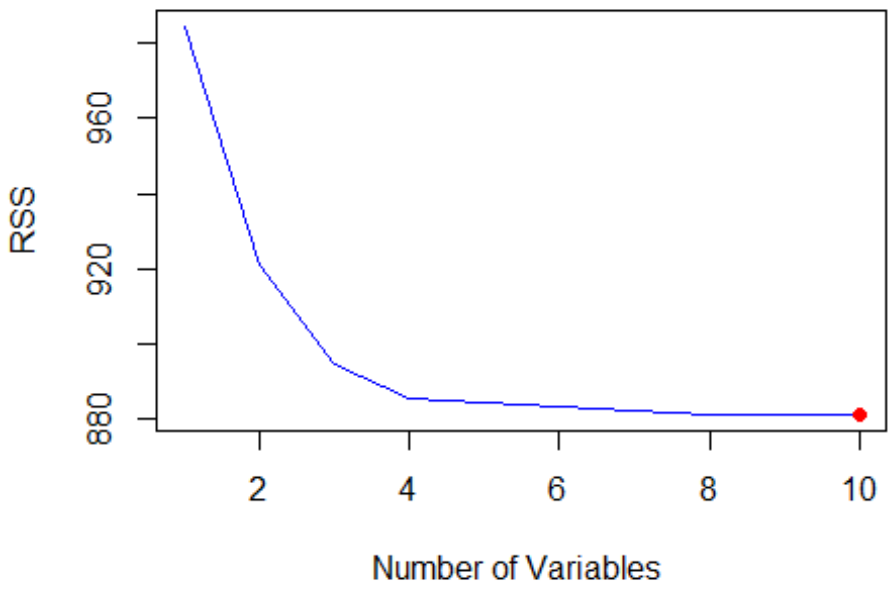
Furthermore, the BIC metric reaches its minimum value within a four-variable model, recorded at -762.1479. An interesting observation is the inclusion of 'HAEMATOCRIT' and 'HAEMOGLOBINS' in some models but not the chosen ones by our used metrics

```
best<- regsubsets(SOURCE ~ .,  data = df, nbest = 1, method
="exhaustive",nvmax=10)
with(summary(best), data.frame(adjr2,cp,bic,outmat))
```

```
##                adjr2         cp       bic HAEMATOCRIT HAEMOGLOBINS
ERYTHROCYTE
## 1  ( 1 )  0.07333386 508.831676 -320.2426           *
## 2  ( 1 )  0.13276967 194.425610 -605.3170           *
## 3  ( 1 )  0.15748602  64.298290 -725.4958           *
## 4  ( 1 )  0.16585465  20.911412 -762.1479                       *
## 5  ( 1 )  0.16673444  17.243897 -759.4130
*
## 6  ( 1 )  0.16749922  14.188083 -756.0736
*
## 7  ( 1 )  0.16862779   9.206503 -754.6684
*
## 8  ( 1 )  0.16916026   7.385411 -750.1049
*
## 9  ( 1 )  0.16904373   9.002993 -742.0962           *
*
## 10  ( 1 ) 0.16885548  11.000000 -733.7071           *            *
*
##           LEUCOCYTE THROMBOCYTE MCH MCHC MCV AGE SEXM
## 1  ( 1 )
## 2  ( 1 )                    *
## 3  ( 1 )          *         *
## 4  ( 1 )          *         *                      *
## 5  ( 1 )          *         *   *                  *
## 6  ( 1 )          *         *   *              *   *
## 7  ( 1 )          *         *   *    *         *   *
## 8  ( 1 )          *         *   *    *    *    *   *
## 9  ( 1 )          *         *   *    *    *    *   *
## 10  ( 1 )         *         *   *    *    *    *   *
```

The model containing all of the predictors will always have the smallest RSS,since this quantity is related to the training error. Instead, we wish to choose a model with a low test error.Therefore, RSS is not suitable for selecting the best model among a collection of models with different numbers of predictors. (similar for R^2)

```r
plot(reg.summary$rss ,xlab="Number of Variables
",ylab="RSS",type="l",col='blue')
min_index <- which.min(reg.summary$rss)
min_rss <- reg.summary$rss[min_index]
points(min_index, min_rss, col = "red", pch = 16)
```

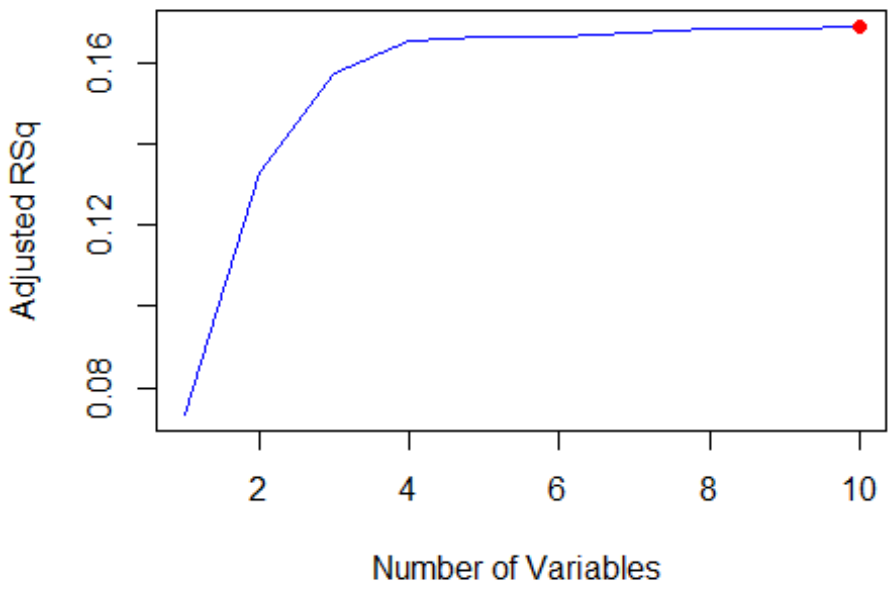


2- Forward Subset Selection

```
regfit.fwd <- regsubsets (SOURCE ~. ,data=df , nvmax=10,method ="forward")
fwd.summary <-  summary (regfit.fwd)
```

Both Adjusted R^2 and Cp results in the selection of a full model with minimal difference in these values between a full model and models containing less variables starting from a certain point like above. BIC also reached its minimum at a four- variable model containing : “HAEMOGLOBINS”, “LEUCOCYTE”, “THROMBOCYTE”, “SEX”

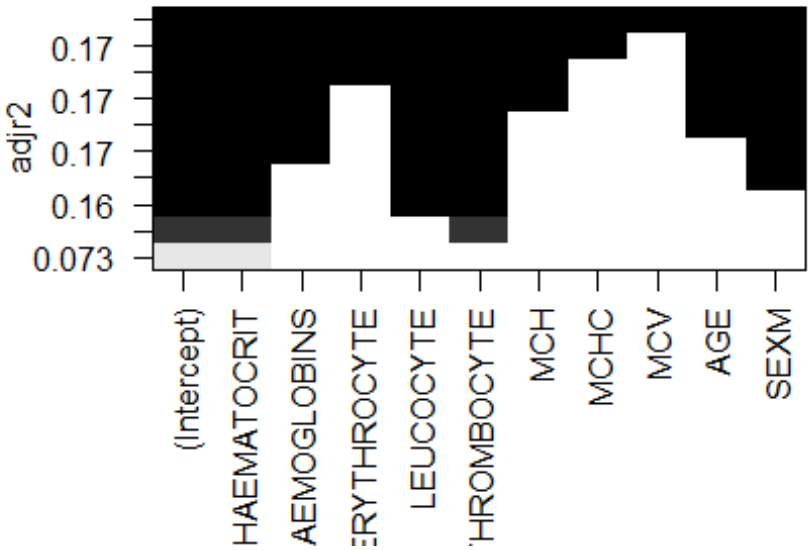
```
plot(fwd.summary$adjr2 ,xlab="Number of Variables ",ylab="Adjusted
RSq",type="l",col='blue')
max_index <- which.max(fwd.summary$adjr2)
max_adjr2 <- fwd.summary$adjr2[max_index]
points(max_index, max_adjr2, col = "red", pch = 16)
```




```
coef(regfit.fwd ,max_index)
```

```
##  (Intercept)  HAEMATOCRIT HAEMOGLOBINS  ERYTHROCYTE    LEUCOCYTE
THROMBOCYTE
##  3.543441580  0.004433542  0.002154334  0.132801509 -0.014427167
0.001250595
##          MCH         MCHC          MCV          AGE         SEXM
##  0.154166345 -0.114389166 -0.040022956 -0.001012640 -0.094475807
```

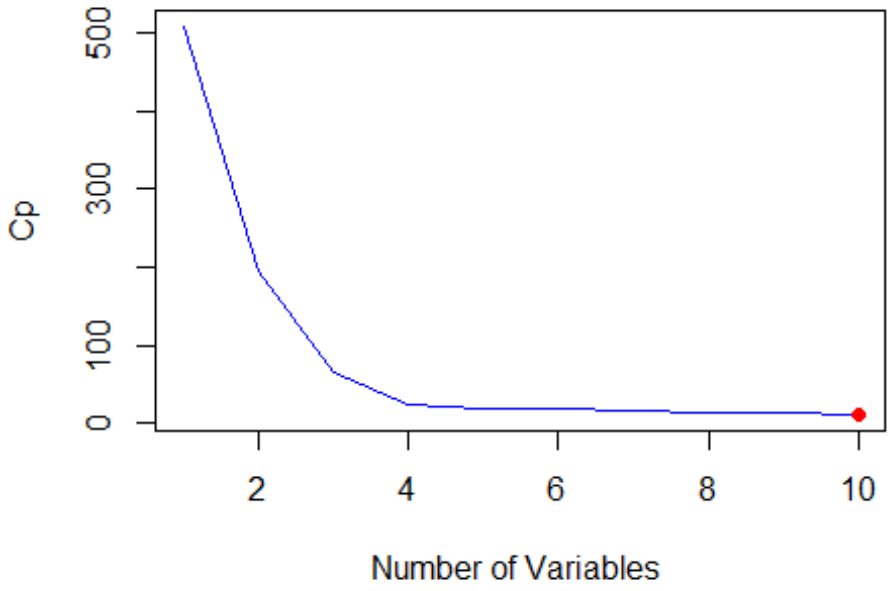
```
plot(regfit.fwd ,scale="adjr2")
```



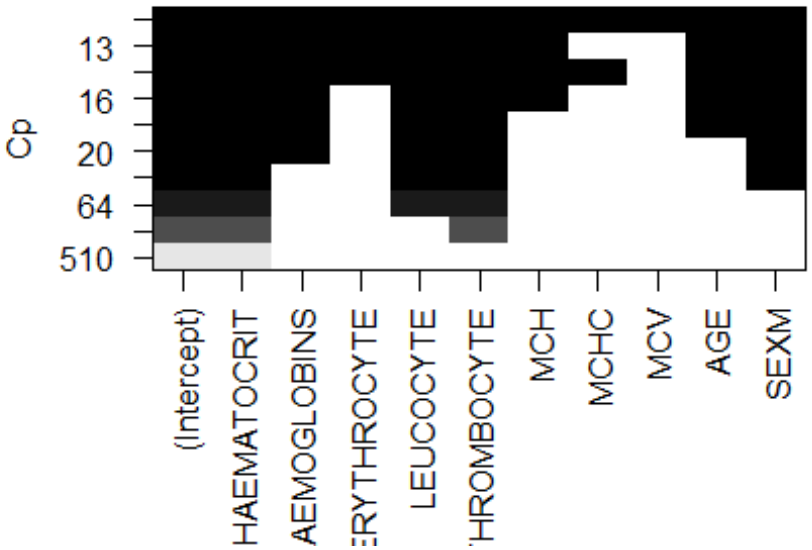

```
coef(regfit.fwd  ,10)
```

```
##  (Intercept)  HAEMATOCRIT HAEMOGLOBINS  ERYTHROCYTE    LEUCOCYTE
THROMBOCYTE
##  3.543441580  0.004433542  0.002154334  0.132801509 -0.014427167
0.001250595
##          MCH         MCHC          MCV          AGE         SEXM
##  0.154166345 -0.114389166 -0.040022956 -0.001012640 -0.094475807
```

```
plot(fwd.summary$cp ,xlab="Number of Variables
",ylab="Cp",type="l",col='blue')
min_index <- which.min(fwd.summary$cp)
min_cp <- fwd.summary$cp[min_index]
points(min_index, min_cp, col = "red", pch = 16)
```



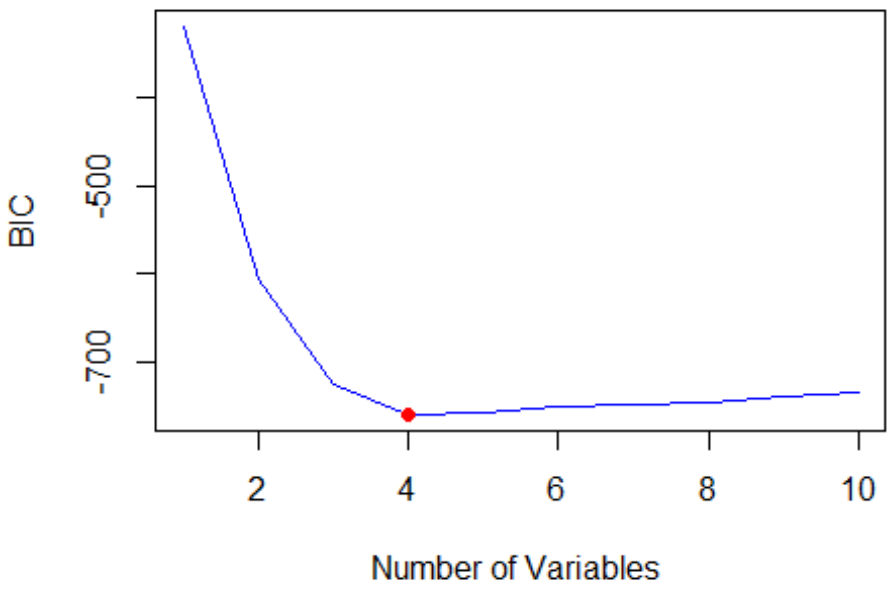

```
coef(regfit.fwd ,min_index)

##  (Intercept)  HAEMATOCRIT HAEMOGLOBINS  ERYTHROCYTE    LEUCOCYTE
THROMBOCYTE
##  3.543441580  0.004433542  0.002154334  0.132801509 -0.014427167
0.001250595
##          MCH         MCHC          MCV          AGE         SEXM
##  0.154166345 -0.114389166 -0.040022956 -0.001012640 -0.094475807

plot(regfit.fwd ,scale="Cp")
```




```
coef(regfit.fwd  ,10)
```

```
##  (Intercept)  HAEMATOCRIT HAEMOGLOBINS  ERYTHROCYTE    LEUCOCYTE
THROMBOCYTE
##  3.543441580  0.004433542  0.002154334  0.132801509 -0.014427167
0.001250595
##          MCH         MCHC          MCV          AGE         SEXM
##  0.154166345 -0.114389166 -0.040022956 -0.001012640 -0.094475807
```

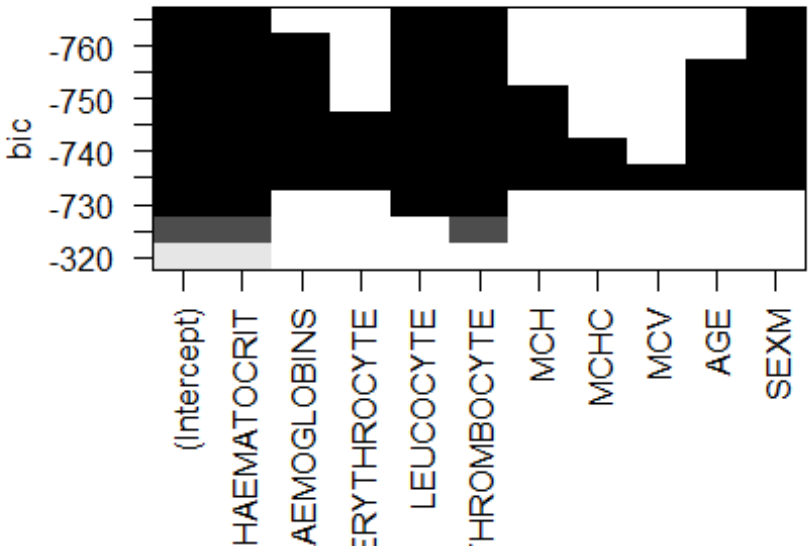
```
plot(fwd.summary$bic ,xlab="Number of Variables
",ylab="BIC",type="l",col='blue')
min_index <- which.min(fwd.summary$bic)
min_bic <- fwd.summary$bic[min_index]
points(min_index, min_bic, col = "red", pch = 16)
```




```
coef(regfit.fwd ,min_index)
```

```
##  (Intercept)  HAEMATOCRIT    LEUCOCYTE  THROMBOCYTE         SEXM
##  0.634149098  0.021714043 -0.015028872  0.001207245 -0.092076836
```

```
plot(regfit.fwd ,scale="bic")
```




```
coef(regfit.fwd  ,4)

##  (Intercept)  HAEMATOCRIT    LEUCOCYTE  THROMBOCYTE         SEXM
##  0.634149098  0.021714043 -0.015028872  0.001207245 -0.092076836
```

With slight variations in the values, our previous insights derived from best subset selection can be extrapolated to forward subset selection. A noticeable difference between best and forward subset selection is that, in forward subset selection, once 'HAEMOGLOBINS' is selected in the initial step, it cannot be removed anymore. Conversely, in best subset selection, although 'HAEMOGLOBINS' may initially appear, it is not seen among the chosen models.

```
forward <- regsubsets(SOURCE ~ .,data = df, nbest = 1, method
="forward",nvmax=10)
with(summary(forward), data.frame(adjr2,cp,bic,outmat))

##                adjr2        cp       bic HAEMATOCRIT HAEMOGLOBINS
ERYTHROCYTE
## 1  ( 1 )  0.07333386 508.83168 -320.2426           *
## 2  ( 1 )  0.13276967 194.42561 -605.3170           *
## 3  ( 1 )  0.15748602  64.29829 -725.4958           *
## 4  ( 1 )  0.16519371  24.41592 -758.6535           *
## 5  ( 1 )  0.16626977  19.70716 -756.9534           *            *
## 6  ( 1 )  0.16649691  19.50026 -750.7649           *            *
## 7  ( 1 )  0.16734676  15.99428 -747.8754           *            *
## 8  ( 1 )  0.16811411  12.92745 -744.5530           *            *
*
```

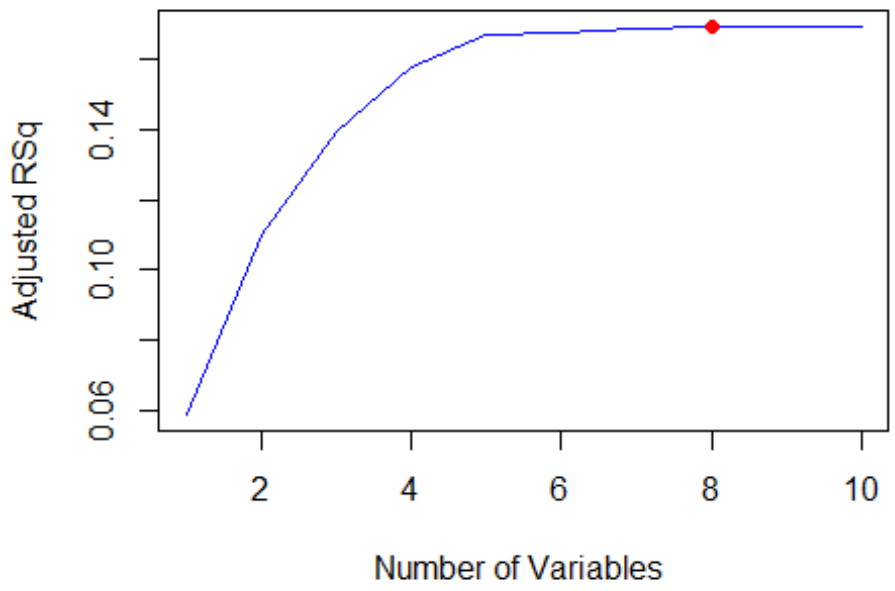
```
## 9  ( 1 )  0.16825006  13.20648 -737.8842           *             *
*
## 10  ( 1 ) 0.16885548  11.00000 -733.7071           *             *
*
##           LEUCOCYTE THROMBOCYTE MCH MCHC MCV AGE SEXM
## 1  ( 1 )
## 2  ( 1 )                     *
## 3  ( 1 )          *          *
## 4  ( 1 )          *          *                     *
## 5  ( 1 )          *          *                     *
## 6  ( 1 )          *          *                *    *
## 7  ( 1 )          *          *   *            *    *
## 8  ( 1 )          *          *   *            *    *
## 9  ( 1 )          *          *   *    *       *    *
## 10  ( 1 )         *          *   *    *   *   *    *
```

3- Backward Subset Selection

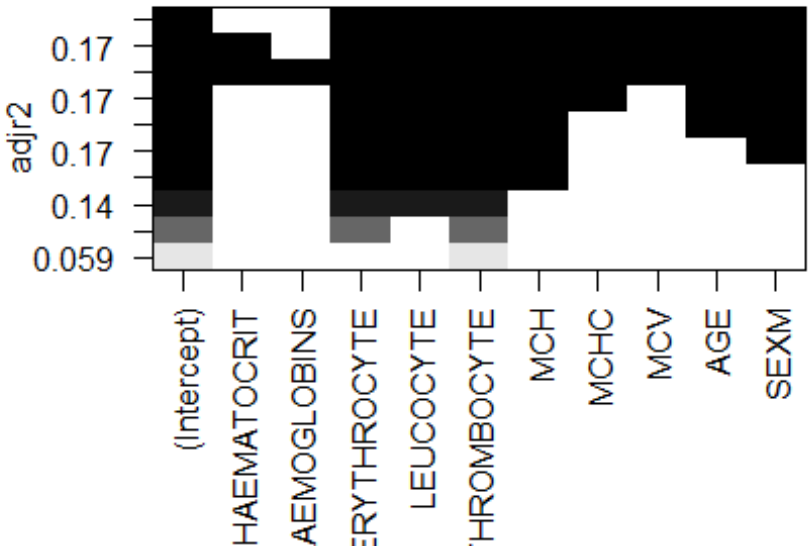
```
regfit.bwd <- regsubsets (SOURCE ~ .,data=df , nvmax=10,method ="backward")
bwd.summary = summary (regfit.bwd)
```

We can see that Adjusted R^2 and Cp chose the same models chosen in best subset selection with a little difference in the curves And BIC chose a five- variable model containing : “ERYTHROCYTE”, “LEUCOCYTE”, “THROMBOCYTE”, “MCH” and “SEX”

```
plot(bwd.summary$adjr2 ,xlab="Number of Variables ",ylab="Adjusted
RSq",type="l",col='blue')
max_index <- which.max(bwd.summary$adjr2)
max_adjr2 <- bwd.summary$adjr2[max_index]
points(max_index, max_adjr2, col = "red", pch = 16)
```



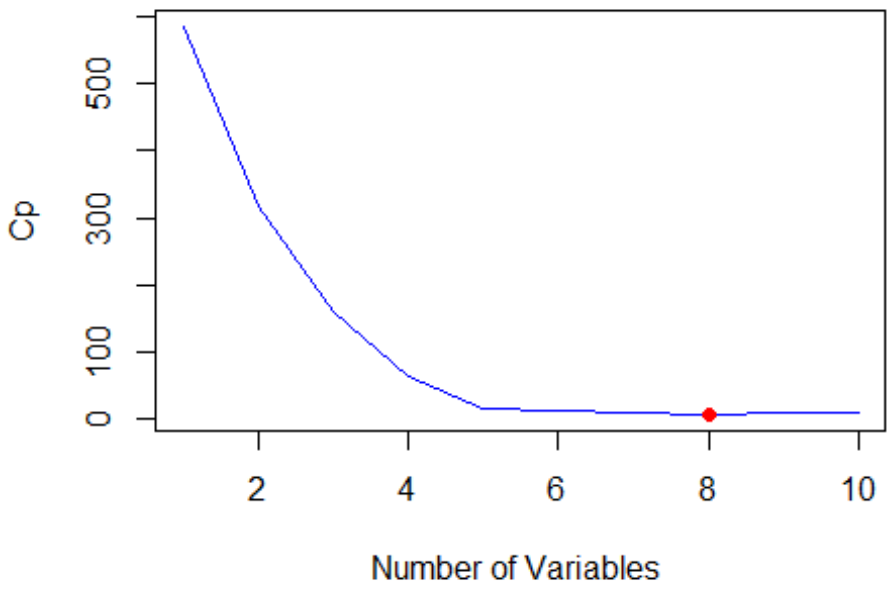

```
coef(regfit.bwd ,max_index)

##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118

plot(regfit.bwd ,scale="adjr2")
```



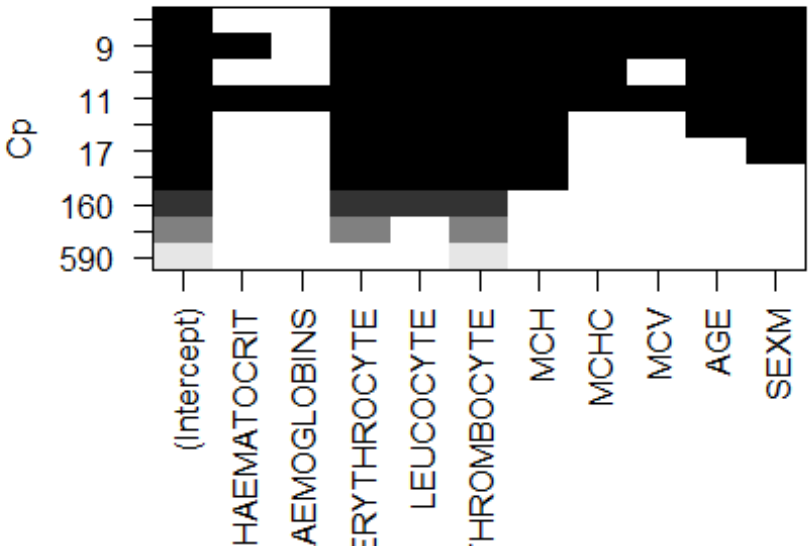

```
coef(regfit.bwd  ,8)
```

```
##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118
```

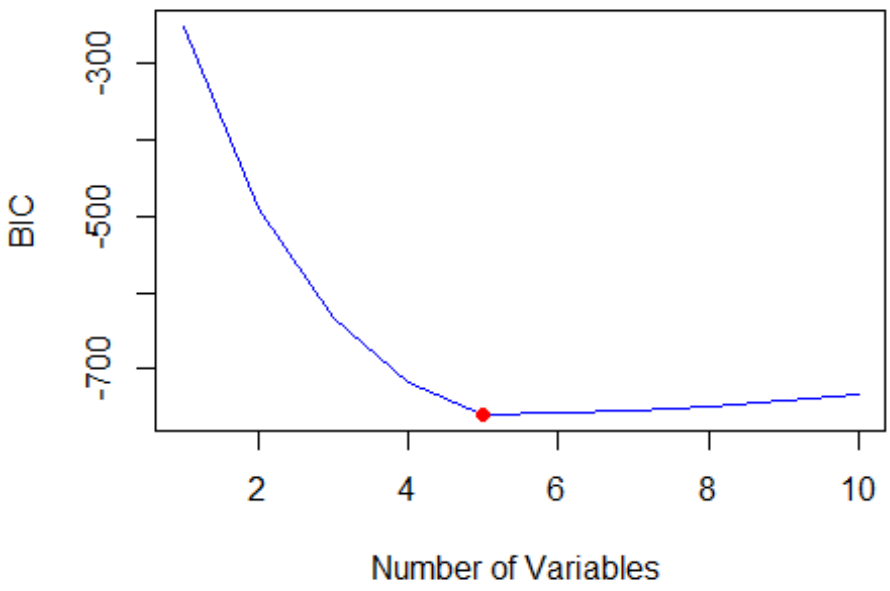
```
plot(bwd.summary$cp ,xlab="Number of Variables
",ylab="Cp",type="l",col='blue')
min_index <- which.min(bwd.summary$cp)
min_cp <- bwd.summary$cp[min_index]
points(min_index, min_cp, col = "red", pch = 16)
```




```
coef(regfit.bwd ,min_index)

##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118

plot(regfit.bwd ,scale="Cp")
```



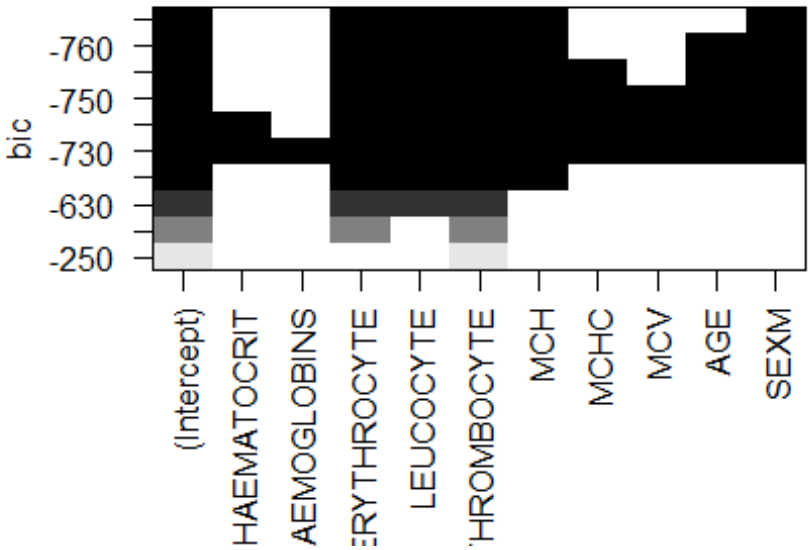

```
coef(regfit.bwd  ,8)

##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
MCHC
##  3.195278368  0.175478003 -0.014484470  0.001255187  0.150263058 -
0.109789821
##          MCV          AGE         SEXM
## -0.036381517 -0.001032425 -0.094210118

plot(bwd.summary$bic ,xlab="Number of Variables
",ylab="BIC",type="l",col='blue')
min_index <- which.min(bwd.summary$bic)
min_bic <- bwd.summary$bic[min_index]
points(min_index, min_bic, col = "red", pch = 16)
```




```
coef(regfit.bwd ,min_index)
```

```
##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
SEXM
## -0.229666633  0.178108391 -0.015596530  0.001275908  0.031036432 -
0.101001368
```

```
plot(regfit.bwd ,scale="bic")
```




```
coef(regfit.bwd  ,5)

##  (Intercept)  ERYTHROCYTE    LEUCOCYTE  THROMBOCYTE          MCH
SEXM
## -0.229666633  0.178108391 -0.015596530  0.001275908  0.031036432 -
0.101001368
```

Since the curves in the graphs represent a slower increase in Adjusted R^2 and a slower decrease in Cp and BIC compared to best and forward subset selection a model with four-variable predictors and a model with five variable predictors show a remarkable difference in adjusted R^2 and Cp in backward subset selection compared to the other methods. Following this point(number of predictors =5),a minimal increase and decrease in adjusted R^2 and Cp respectively are shown in the values and curves.

```
backward <- regsubsets(SOURCE ~., data = df, nbest = 1, method
="backward",nvmax=10)
with(summary(backward), data.frame(adjr2,cp,bic,outmat))

##                adjr2         cp       bic HAEMATOCRIT HAEMOGLOBINS
ERYTHROCYTE
## 1  ( 1 )  0.05893411 585.235815 -252.2104
## 2  ( 1 )  0.10972175 316.688690 -489.5927
*
## 3  ( 1 )  0.13944780 159.964523 -632.0320
*
## 4  ( 1 )  0.15759897  64.685715 -718.6963
*
```

```
## 5  ( 1 )  0.16673444  17.243897 -759.4130
*
## 6  ( 1 )  0.16749922  14.188083 -756.0736
*
## 7  ( 1 )  0.16862779   9.206503 -754.6684
*
## 8  ( 1 )  0.16916026   7.385411 -750.1049
*
## 9  ( 1 )  0.16904373   9.002993 -742.0962           *
*
## 10  ( 1 ) 0.16885548  11.000000 -733.7071           *            *
*
##           LEUCOCYTE THROMBOCYTE MCH MCHC MCV AGE SEXM
## 1  ( 1 )                      *
## 2  ( 1 )                      *
## 3  ( 1 )          *           *
## 4  ( 1 )          *           *   *
## 5  ( 1 )          *           *   *                *
## 6  ( 1 )          *           *   *            *   *
## 7  ( 1 )          *           *   *    *       *   *
## 8  ( 1 )          *           *   *    *   *   *   *
## 9  ( 1 )          *           *   *    *   *   *   *
## 10  ( 1 )         *           *   *    *   *   *   *
```

We present an overall summary :

The same models were selected by Cp and Adjusted R^2 in best and backward subset selection but a full one was selected in the forward approach. BIC chose a four-variable model in best and forward subset selection with a predictor difference HAEMOGLOBINS in best and HAEMATOCRIT in forward Finally BIC selected a five variable model in the backward method Since BIC place a heavier penalty on models with many variables when n>7 it is not unusual to see that BIC chose models with lowest predictors

```
values <- c(8, 8,
4,'ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX','ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX','HAEM
O,LEU,THR,SEX',
            10, 10, 4, 'ALL', 'ALL','HAEMA,LEU,THR,SEX', 8, 8,
5,'ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX','ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX','ERY,
LEU,THR,MCH,SEX')
overview <- matrix(values,nrow=6, ncol=3, byrow=TRUE)
colnames(overview) <- c('Adjusted R^2', 'Cp', 'BIC')
rownames(overview) <- c('Best:numb of X selected', 'Best:Selected
features','Forward:numb of X selected', 'Forward:Selected
features','Backward:numb of X selected', 'Backward:Selected features')
print(overview)
```

```
##                            Adjusted R^2
## Best:numb of X selected     "8"
## Best:Selected features      "ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX"
## Forward:numb of X selected  "10"
```

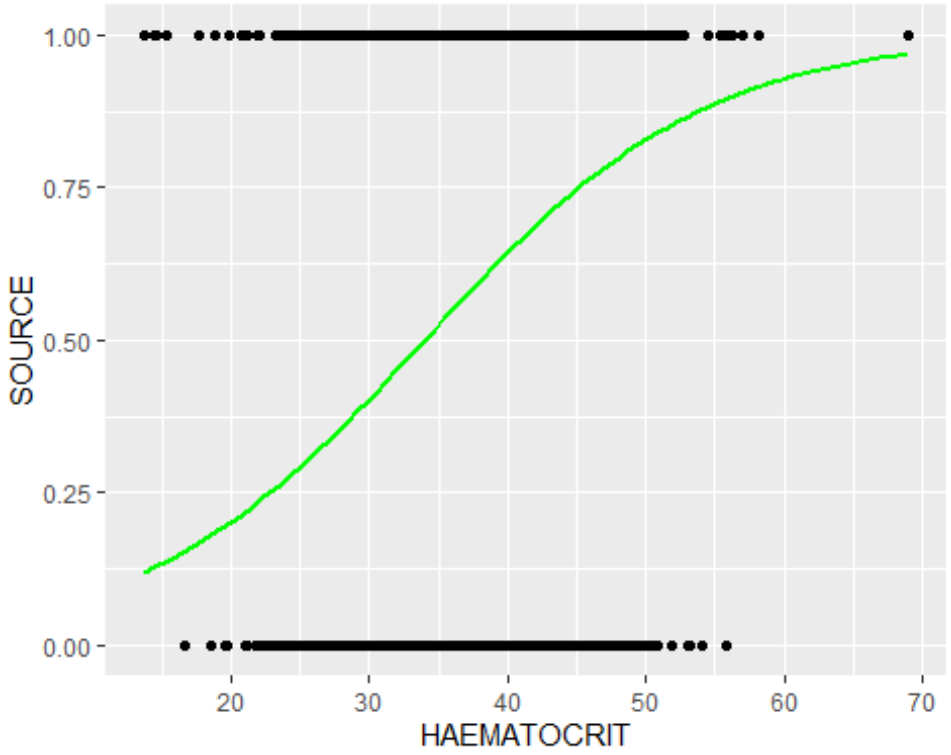
```
## Forward:Selected features   "ALL"
## Backward:numb of X selected "8"
## Backward:Selected features  "ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX"
##                             Cp
## Best:numb of X selected     "8"
## Best:Selected features      "ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX"
## Forward:numb of X selected  "10"
## Forward:Selected features   "ALL"
## Backward:numb of X selected "8"
## Backward:Selected features  "ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX"
##                             BIC
## Best:numb of X selected     "4"
## Best:Selected features      "HAEMO,LEU,THR,SEX"
## Forward:numb of X selected  "4"
## Forward:Selected features   "HAEMA,LEU,THR,SEX"
## Backward:numb of X selected "5"
## Backward:Selected features  "ERY,LEU,THR,MCH,SEX"
```

```
df$SOURCE<-ifelse(df$SOURCE=="in",0,1)
```

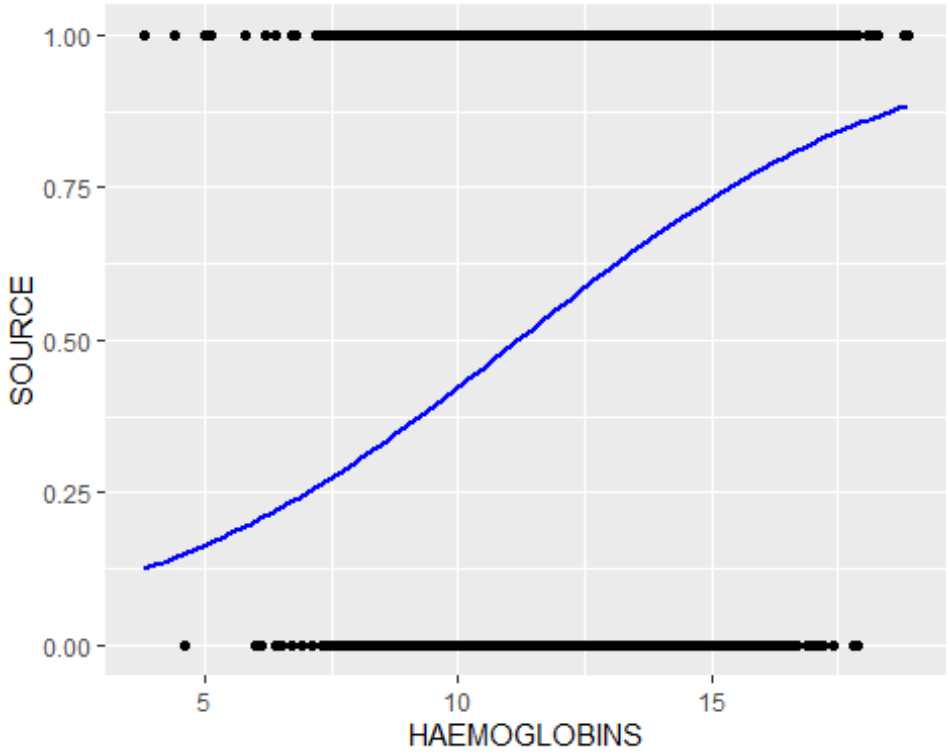
We plot each predictor with the response with the aim to see an S-shaped curve However we do not obtain any complete S-shape curve and the final four plots are linear, knowing that our models are correct. We checked the output using a different library than ggplot, library(lessR) and we got the same curves What we found after doing a lit bit of research was that An intercept estimates the expected value of the response on the logit scale when all of the features are zero. In this case, the intercept is such that setting all the features to zero yields a prediction that is not zero on the probability scale.

```
ggplot(df, aes(x = HAEMATOCRIT , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="green", se=FALSE,
              method.args = list(family=binomial))
```

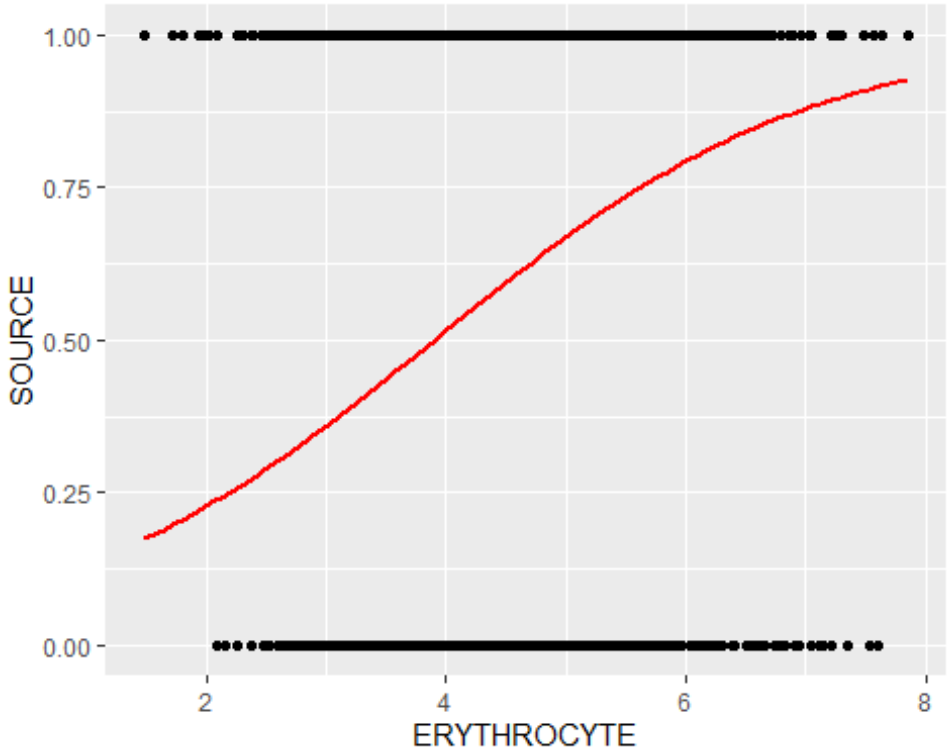
```
## `geom_smooth()` using formula = 'y ~ x'
```



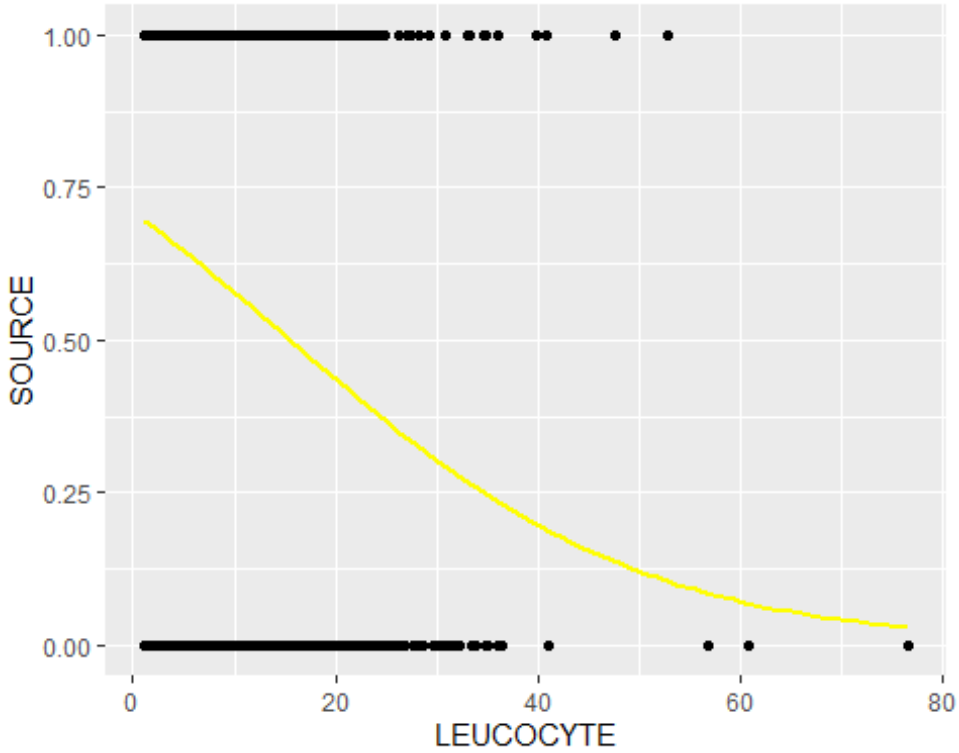

```
ggplot(df, aes(x = HAEMOGLOBINS , y =SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="blue", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```



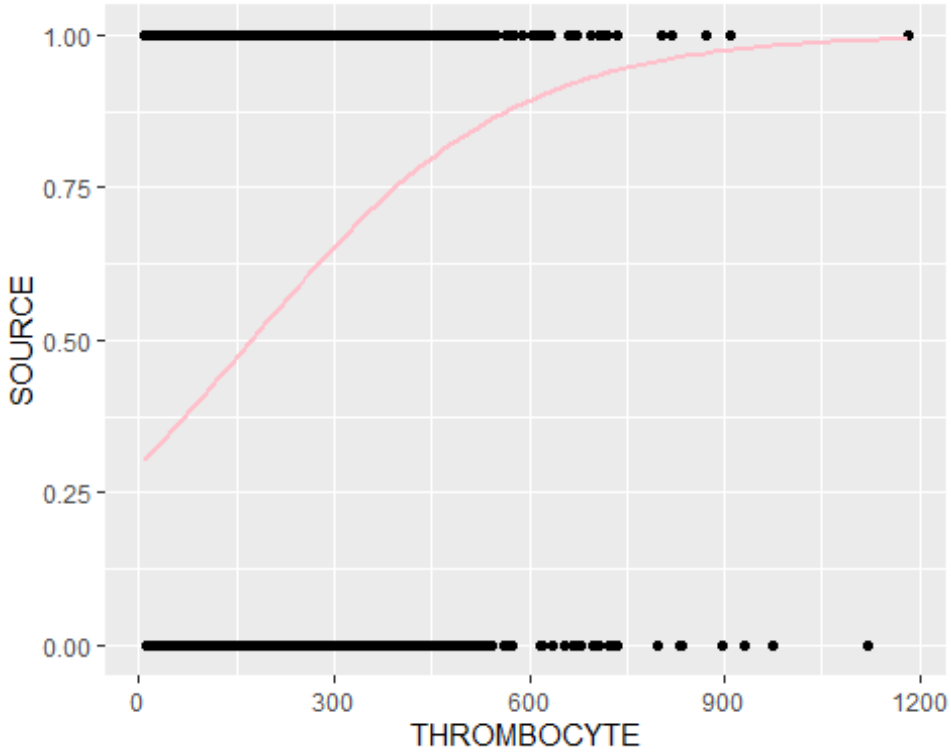

```
ggplot(df, aes(x = ERYTHROCYTE , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="red", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```



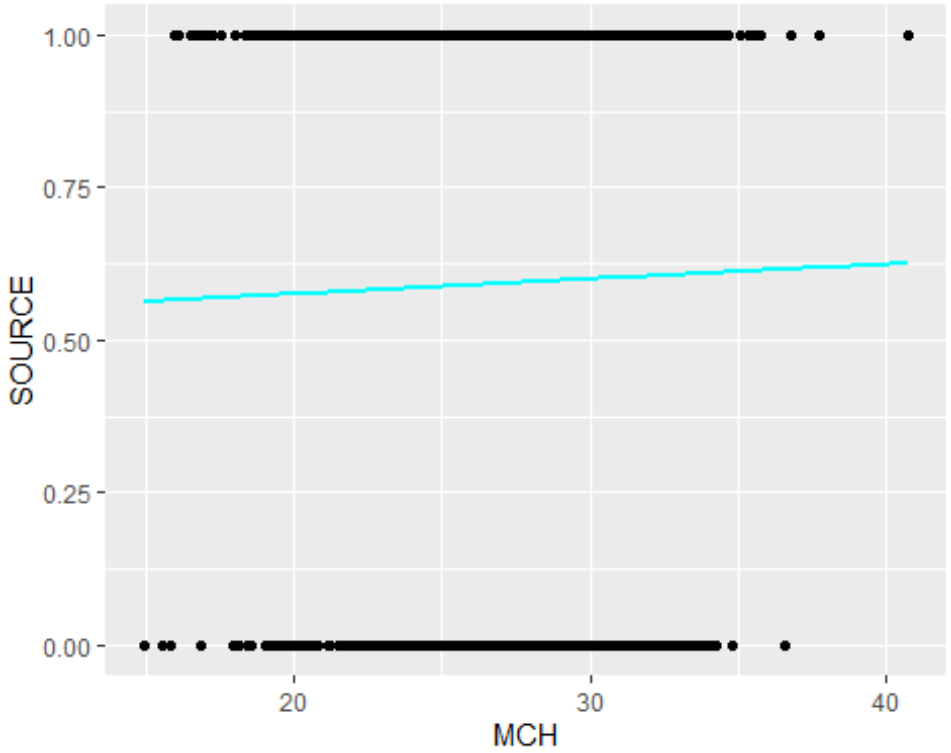

```
ggplot(df, aes(x = LEUCOCYTE , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="yellow", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```



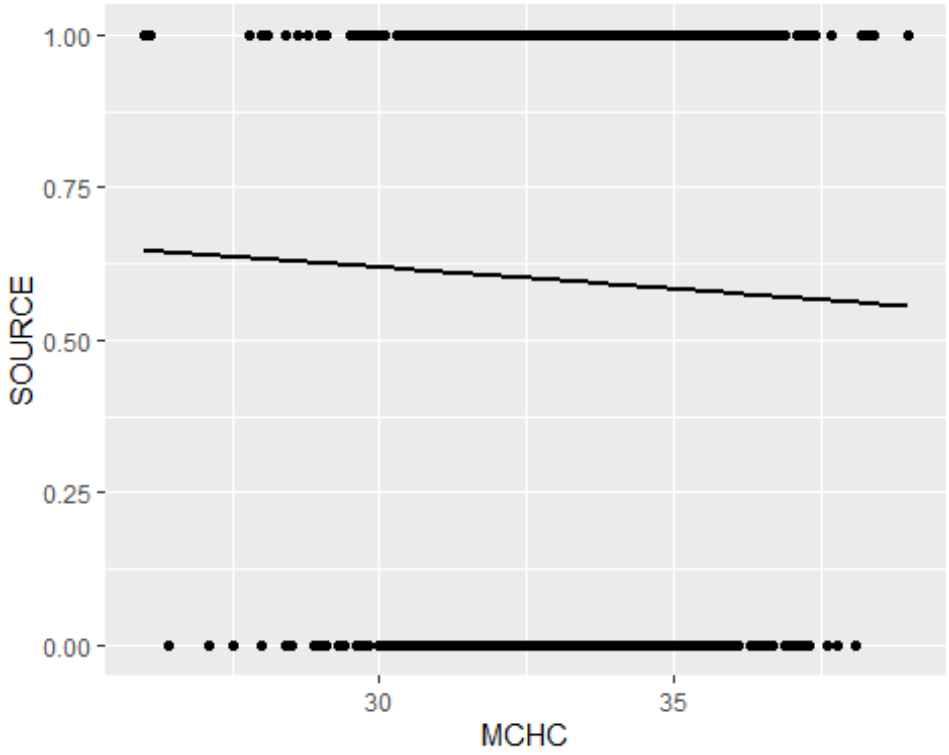

```
ggplot(df, aes(x = THROMBOCYTE , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="pink", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```



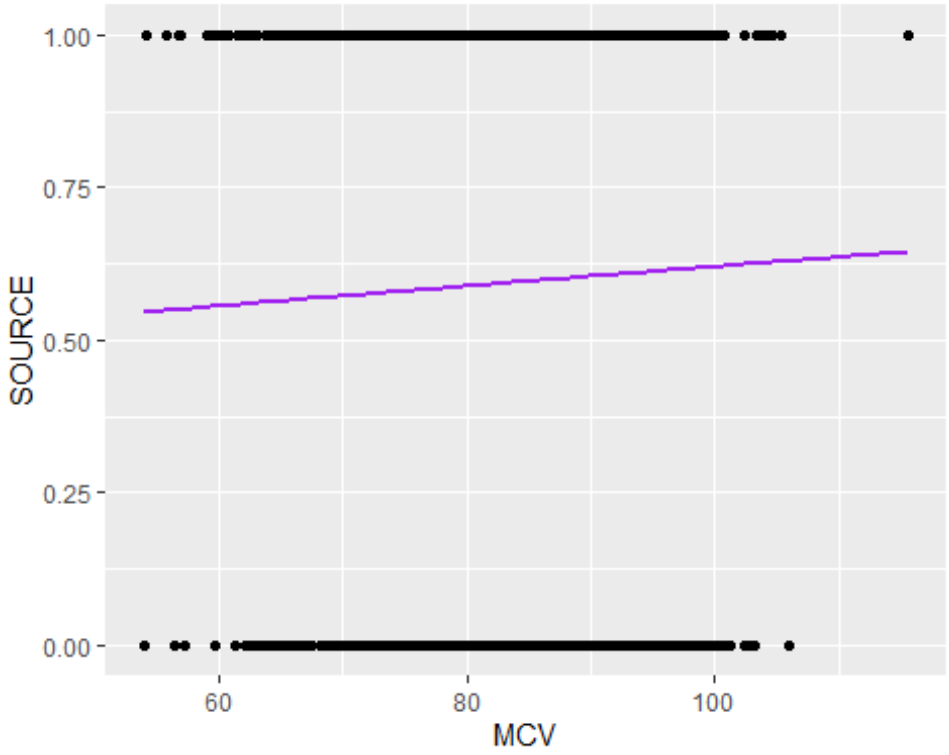

```
ggplot(df, aes(x = MCH , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="cyan", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```



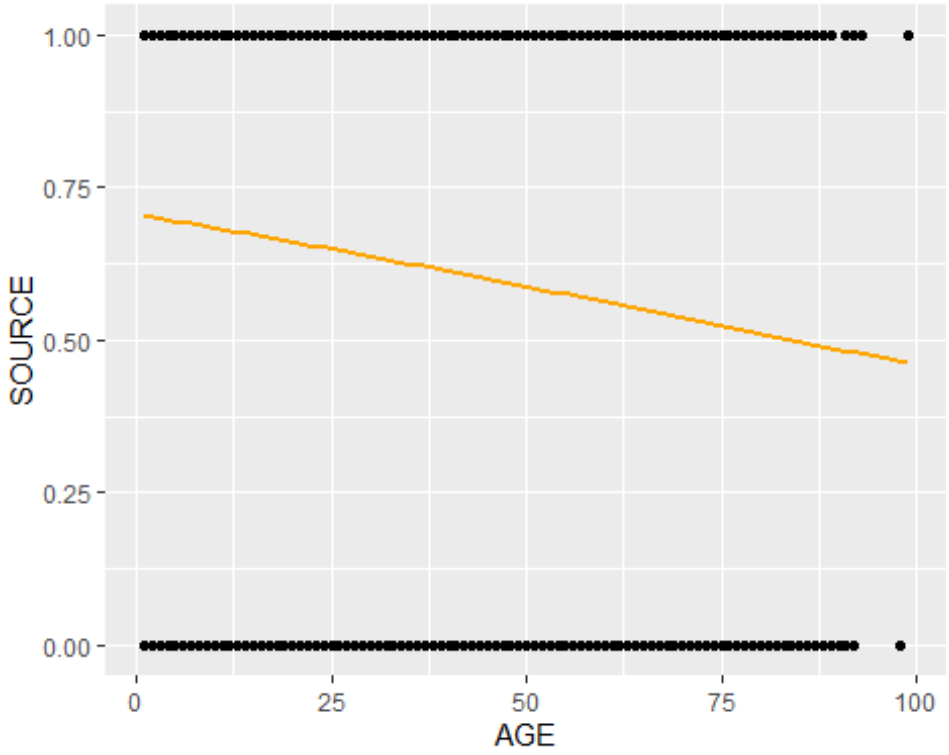

```
ggplot(df, aes(x = MCHC , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="black", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```




```
ggplot(df, aes(x = MCV , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="purple", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```




```
ggplot(df, aes(x = AGE , y = SOURCE)) + geom_point() +
  stat_smooth(method="glm", color="orange", se=FALSE,
              method.args = list(family=binomial))

## `geom_smooth()` using formula = 'y ~ x'
```



From the selected models identified through best, forward and backward subset selection employing adjusted R^2, Cp, and BIC statistics, our objective is to determine the most suitable one for our data. We plan to begin by fitting these models—starting with logistic regression, followed by LDA and QDA—and then assess the accuracy of these models using k-fold cross-validation.

```
df$SOURCE <- as.factor(df$SOURCE)
set.seed(1234)
df <- df %>% mutate(id=row_number())
dim(df)

## [1] 4412   12

tr= df %>% slice_sample(prop=0.7)
te=anti_join(df,tr,by='id')
tr=tr %>% mutate(id=NULL)
te=te %>% mutate(id=NULL)
```

Here the number of predictors is 10 so performing 10-fold cross validation is the same as leave_on_out cross validation So in our case we are performing both at the same time

1- logistic regression

```
set.seed(000)
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats =
10,savePredictions = "all")
```

In the first model containing eight predictors, the Intercept and AGE have insignificant p-value.

```
model1 <- train(SOURCE ~
ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+MCHC+MCV+AGE+SEX,
                tr,method="glm",family = binomial(),
                trControl=ctrl)

model1

## Generalized Linear Model
##
## 3088 samples
##    8 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 2780, 2778, 2779, 2780, 2779, 2779, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.721049  0.3969339

summary(model1)

##
## Call:
## NULL
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) 18.5699860  9.5015236   1.954  0.05065 .
## ERYTHROCYTE  0.8935236  0.0684533  13.053  < 2e-16 ***
## LEUCOCYTE   -0.0790774  0.0098989  -7.988 1.37e-15 ***
## THROMBOCYTE  0.0067932  0.0004473  15.187  < 2e-16 ***
## MCH          1.0973082  0.3599748   3.048  0.00230 **
## MCHC        -0.8596684  0.2932508  -2.932  0.00337 **
## MCV         -0.2973140  0.1169461  -2.542  0.01101 *
## AGE         -0.0037321  0.0021626  -1.726  0.08440 .
## SEXM        -0.4696637  0.0887430  -5.292 1.21e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4177.3  on 3087  degrees of freedom
## Residual deviance: 3568.0  on 3079  degrees of freedom
## AIC: 3586
##
## Number of Fisher Scoring iterations: 4
```

All four predictors have a significant p-value

```
model2 <- train(SOURCE ~ HAEMOGLOBINS+LEUCOCYTE+THROMBOCYTE+SEX,
                tr,method="glm",family = binomial(),
                trControl=ctrl)
model2
```

```
## Generalized Linear Model
##
## 3088 samples
##    4 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 2778, 2780, 2780, 2780, 2779, 2780, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7255493  0.4066896
```

```
summary(model2)
```

```
##
## Call:
## NULL
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.2387857  0.3173089 -13.359  < 2e-16 ***
## HAEMOGLOBINS  0.3053447  0.0224595  13.595  < 2e-16 ***
## LEUCOCYTE    -0.0842576  0.0097469  -8.645  < 2e-16 ***
## THROMBOCYTE   0.0068817  0.0004357  15.794  < 2e-16 ***
## SEXM         -0.4920781  0.0869681  -5.658 1.53e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4177.3  on 3087  degrees of freedom
## Residual deviance: 3590.7  on 3083  degrees of freedom
## AIC: 3600.7
##
## Number of Fisher Scoring iterations: 4
```

The full model has insignificant p-values for the Intercept, HAEMATOCRIT, HAEMOGLOBINS and AGE

```
model3 <- train(SOURCE ~ .,
                tr,method="glm",family = binomial(),
```

```
                trControl=ctrl)
model3
```

```
## Generalized Linear Model
##
## 3088 samples
##   10 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 2779, 2780, 2779, 2779, 2779, 2778, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7217233  0.3983438
```

```
summary(model3)
```

```
##
## Call:
## NULL
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  18.0575894 10.4446025   1.729  0.08383 .
## HAEMATOCRIT   0.0178641  0.0637646   0.280  0.77936
## HAEMOGLOBINS -0.0589073  0.2250913  -0.262  0.79355
## ERYTHROCYTE   0.9076705  0.4617120   1.966  0.04931 *
## LEUCOCYTE    -0.0790839  0.0099226  -7.970 1.59e-15 ***
## THROMBOCYTE   0.0067947  0.0004511  15.061  < 2e-16 ***
## MCH           1.1069003  0.3617949   3.059  0.00222 **
## MCHC         -0.8463667  0.3019136  -2.803  0.00506 **
## MCV          -0.2996352  0.1211388  -2.473  0.01338 *
## AGE          -0.0037782  0.0021856  -1.729  0.08387 .
## SEXM         -0.4691714  0.0887762  -5.285 1.26e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4177.3  on 3087  degrees of freedom
## Residual deviance: 3567.9  on 3077  degrees of freedom
## AIC: 3589.9
##
## Number of Fisher Scoring iterations: 4
```

All preditors in model 4 and 5 have significant p-values.

```
model4 <- train(SOURCE ~ HAEMATOCRIT+LEUCOCYTE+THROMBOCYTE+SEX,
                tr,method="glm",family = binomial(),
```

```
                trControl=ctrl)
model4
```

```
## Generalized Linear Model
##
## 3088 samples
##    4 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 2779, 2779, 2779, 2780, 2779, 2779, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7214133  0.3974131
```

```
summary(model4)
```

```
##
## Call:
## NULL
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.4242567  0.3277480 -13.499  < 2e-16 ***
## HAEMATOCRIT  0.1072899  0.0078250  13.711  < 2e-16 ***
## LEUCOCYTE   -0.0809339  0.0097659  -8.287  < 2e-16 ***
## THROMBOCYTE  0.0066031  0.0004344  15.199  < 2e-16 ***
## SEXM        -0.4564210  0.0860579  -5.304 1.14e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4177.3  on 3087  degrees of freedom
## Residual deviance: 3585.8  on 3083  degrees of freedom
## AIC: 3595.8
##
## Number of Fisher Scoring iterations: 4
```

```
model5 <- train(SOURCE ~ ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+SEX,
                tr,method="glm",family = binomial(),
                trControl=ctrl)
model5
```

```
## Generalized Linear Model
##
## 3088 samples
##    5 predictor
##    2 classes: '0', '1'
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 2779, 2779, 2779, 2780, 2779, 2779, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7245792  0.4042994
```

```
summary(model5)
```

```
##
## Call:
## NULL
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.3521228  0.6903437 -12.098  < 2e-16 ***
## ERYTHROCYTE  0.8800144  0.0647384  13.593  < 2e-16 ***
## LEUCOCYTE   -0.0844079  0.0097921  -8.620  < 2e-16 ***
## THROMBOCYTE  0.0069421  0.0004421  15.702  < 2e-16 ***
## MCH          0.1416350  0.0169674   8.347  < 2e-16 ***
## SEXM        -0.5044531  0.0872448  -5.782 7.38e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4177.3  on 3087  degrees of freedom
## Residual deviance: 3583.4  on 3082  degrees of freedom
## AIC: 3595.4
##
## Number of Fisher Scoring iterations: 4
```

What we really care about is how each of our models perform on unseen data

```
pred1 <- predict(model1,newdata = te)
confusionMatrix(data=pred1, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 248 104
##          1 274 698
##
##                Accuracy : 0.7145
##                  95% CI : (0.6893, 0.7387)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                  Kappa : 0.3662
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4751
##             Specificity : 0.8703
##          Pos Pred Value : 0.7045
##          Neg Pred Value : 0.7181
##              Prevalence : 0.3943
##          Detection Rate : 0.1873
##    Detection Prevalence : 0.2659
##       Balanced Accuracy : 0.6727
##
##        'Positive' Class : 0
##
```

```
pred2 <- predict(model2,newdata = te)
confusionMatrix(data=pred2, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 247 107
##          1 275 695
##
##                Accuracy : 0.7115
##                  95% CI : (0.6862, 0.7358)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 6.035e-16
##
##                   Kappa : 0.36
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4732
##             Specificity : 0.8666
##          Pos Pred Value : 0.6977
##          Neg Pred Value : 0.7165
##              Prevalence : 0.3943
##          Detection Rate : 0.1866
##    Detection Prevalence : 0.2674
##       Balanced Accuracy : 0.6699
##
##        'Positive' Class : 0
##
```

```
pred3 <- predict(model3,newdata = te)
confusionMatrix(data=pred3, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 248 104
##          1 274 698
##
##                Accuracy : 0.7145
##                  95% CI : (0.6893, 0.7387)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3662
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4751
##             Specificity : 0.8703
##          Pos Pred Value : 0.7045
##          Neg Pred Value : 0.7181
##              Prevalence : 0.3943
##          Detection Rate : 0.1873
##    Detection Prevalence : 0.2659
##       Balanced Accuracy : 0.6727
##
##        'Positive' Class : 0
##
```

```
pred4 <- predict(model4,newdata = te)
confusionMatrix(data=pred4, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 245 112
##          1 277 690
##
##                Accuracy : 0.7062
##                  95% CI : (0.6808, 0.7306)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.571e-14
##
##                   Kappa : 0.349
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4693
##             Specificity : 0.8603
##          Pos Pred Value : 0.6863
```

```
##          Neg Pred Value : 0.7135
##              Prevalence : 0.3943
##          Detection Rate : 0.1850
##    Detection Prevalence : 0.2696
##       Balanced Accuracy : 0.6648
##
##        'Positive' Class : 0
##
```

```
pred5 <- predict(model5,newdata = te)
confusionMatrix(data=pred5, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 247 105
##          1 275 697
##
##                Accuracy : 0.713
##                  95% CI : (0.6878, 0.7372)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 2.302e-16
##
##                   Kappa : 0.3629
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4732
##             Specificity : 0.8691
##          Pos Pred Value : 0.7017
##          Neg Pred Value : 0.7171
##              Prevalence : 0.3943
##          Detection Rate : 0.1866
##    Detection Prevalence : 0.2659
##       Balanced Accuracy : 0.6711
##
##        'Positive' Class : 0
##
```

```
valuesGLM <- c(3568,0.7145,0.3662,0.4751,0.8703,
               3590.7,0.7115,0.36,0.4732,0.8666,
               3567.9,0.7145,0.3662,0.4751,0.8703,
               3585.8,0.7062,0.349,0.4693,0.8603,
               3583.4,0.713,0.3629,0.4732,0.8691)
GLMoverview <- matrix(valuesGLM,nrow=5, ncol=5, byrow=TRUE)
colnames(GLMoverview) <- c('Deviance', 'Accuracy (test)', 'Kappa',
'Sensitivity','Specificity')
rownames(GLMoverview) <- c('Model1', 'Model2','Model3', 'Model4','Model5')
```

```
view(GLMoverview)
print(GLMoverview)

##        Deviance Accuracy (test)  Kappa Sensitivity Specificity
## Model1   3568.0          0.7145 0.3662      0.4751      0.8703
## Model2   3590.7          0.7115 0.3600      0.4732      0.8666
## Model3   3567.9          0.7145 0.3662      0.4751      0.8703
## Model4   3585.8          0.7062 0.3490      0.4693      0.8603
## Model5   3583.4          0.7130 0.3629      0.4732      0.8691
```

The third model exhibits the lowest deviance, as expected since it represents the full model, and as the number of predictors increases, the deviance tends to decrease across all models.

Similar sensitivities and specificities are observed in all models. The kappa values, ranging from 0.349 to 0.3662 in the five models, fall within the 0.21–0.40 range, indicating a fair to moderate level of agreement beyond chance.

Regarding accuracy, which corresponds to correctly classified observations across all classifications, model 4, comprising four predictors, displays the lowest accuracy at 0.7062, hence the highest error rate, given that accuracy= 1 - the error rate. Models 1 and 3, with 8 and 10 predictors respectively, exhibit the highest accuracy of 0.7145. Model 5, containing five predictors, achieves an accuracy of 0.713, while model 2, with four predictors, reaches an accuracy of 0.7115.

Our preference is for model 2, containing HAEMOGLOBINS, LEUCOCYTE, THROMBOCYTE, and SEX. Its accuracy differs only by 0.003 from the highest accurate models, and it possesses only four predictors. Thus, sacrificing 0.003 accuracy for substantially enhanced interpretability compared to models with 8 or 10 predictors seems a reasonable choice.

Logistic regression doesn't assume any specific data distribution or covariance matrices. In contrast, LDA and QDA assume that each class follows a Gaussian distribution, with LDA further assuming equal covariance matrices for each class.

Let's investigate whether employing LDA and QDA leads to selecting the same model as logistic regression or not. Additionally, we'll explore whether fitting the models with LDA and QDA results in a notable increase in accuracy or a reduction in test error compared to logistic regression.

We'll apply LDA and QDA to models containing 4, 5, and 8 predictors. It's evident that the full model won't be chosen as it lacks interpretability and based on the logistic model, it won't anticipate a significant improvement in prediction accuracy. Hence Model3 will not be included anymore

2- LDA Cross Validation

```
k.lda.mod1 <- train(SOURCE ~
ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+MCHC+MCV+AGE+SEX,
```

```
                tr,method="lda",family = binomial(),
                trControl=ctrl)
kpred1 <- predict(k.lda.mod1,newdata = te)
confusionMatrix(data=kpred1, te$SOURCE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 234  96
##          1 288 706
##
##                Accuracy : 0.71
##                  95% CI : (0.6847, 0.7343)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.559e-15
##
##                   Kappa : 0.3511
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4483
##             Specificity : 0.8803
##          Pos Pred Value : 0.7091
##          Neg Pred Value : 0.7103
##              Prevalence : 0.3943
##          Detection Rate : 0.1767
##    Detection Prevalence : 0.2492
##       Balanced Accuracy : 0.6643
##
##        'Positive' Class : 0
##

k.lda.mod2 <- train(SOURCE ~ HAEMOGLOBINS+LEUCOCYTE+THROMBOCYTE+SEX,
                    tr,method="lda",family = binomial(),
                    trControl=ctrl)
kpred2 <- predict(k.lda.mod2,newdata = te)
confusionMatrix(data=kpred2, te$SOURCE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 236  95
##          1 286 707
##
##                Accuracy : 0.7122
##                  95% CI : (0.687, 0.7365)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 3.734e-16
```

```
##
##                  Kappa : 0.3564
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4521
##             Specificity : 0.8815
##          Pos Pred Value : 0.7130
##          Neg Pred Value : 0.7120
##              Prevalence : 0.3943
##          Detection Rate : 0.1782
##    Detection Prevalence : 0.2500
##       Balanced Accuracy : 0.6668
##
##        'Positive' Class : 0
##
```

```
k.lda.mod3 <- train(SOURCE ~ HAEMATOCRIT+LEUCOCYTE+THROMBOCYTE+SEX,
                    tr,method="lda",family = binomial(),
                    trControl=ctrl)
kpred3 <- predict(k.lda.mod3,newdata = te)
confusionMatrix(data=kpred3, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 228 100
##          1 294 702
##
##                Accuracy : 0.7024
##                  95% CI : (0.677, 0.7269)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.447e-13
##
##                   Kappa : 0.3337
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4368
##             Specificity : 0.8753
##          Pos Pred Value : 0.6951
##          Neg Pred Value : 0.7048
##              Prevalence : 0.3943
##          Detection Rate : 0.1722
##    Detection Prevalence : 0.2477
##       Balanced Accuracy : 0.6560
##
##        'Positive' Class : 0
##
```

```r
k.lda.mod4 <- train(SOURCE ~ ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+SEX,
                    tr,method="lda",family = binomial(),
                    trControl=ctrl)
kpred4 <- predict(k.lda.mod4,newdata = te)
confusionMatrix(data=kpred4, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 232 101
##          1 290 701
##
##                Accuracy : 0.7047
##                  95% CI : (0.6793, 0.7292)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 3.86e-14
##
##                   Kappa : 0.34
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4444
##             Specificity : 0.8741
##          Pos Pred Value : 0.6967
##          Neg Pred Value : 0.7074
##              Prevalence : 0.3943
##          Detection Rate : 0.1752
##    Detection Prevalence : 0.2515
##       Balanced Accuracy : 0.6593
##
##        'Positive' Class : 0
##
```

```r
valuesLDA <- c(0.71,0.3511,0.4483,0.8803,
               0.7122,0.3564,0.4521,0.8815,
               0.7024,0.3337,0.4368,0.8753,
               0.7047,0.34,0.4444,0.8741)
LDAoverview <- matrix(valuesLDA,nrow=4, ncol=4, byrow=TRUE)
colnames(LDAoverview) <- c('Accuracy (test)', 'Kappa',
'Sensitivity','Specificity')
rownames(LDAoverview) <- c('Model1', 'Model2','Model4', 'Model5')
view(LDAoverview)
print(LDAoverview)
```

```
##        Accuracy (test)  Kappa Sensitivity Specificity
## Model1          0.7100 0.3511      0.4483      0.8803
## Model2          0.7122 0.3564      0.4521      0.8815
## Model4          0.7024 0.3337      0.4368      0.8753
## Model5          0.7047 0.3400      0.4444      0.8741
```

There is no remarkable difference between these values and the values obtained previously in Logistic regression Once again we will choose Model2 containing HAEMOGLOBINS, LEUCOCYTE, THROMBOCYTE, and SEX. In LDA this models shows the highest accuracy hence lowest test error.

3- QDA Cross-Validation

```
k.qda.mod1 <- train(SOURCE ~
ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+MCHC+MCV+AGE+SEX,
                    tr,method="qda",family = binomial(),
                    trControl=ctrl)
k.pred1 <- predict(k.qda.mod1,newdata = te)
confusionMatrix(data=k.pred1, te$SOURCE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 269 104
##          1 253 698
##
##                Accuracy : 0.7304
##                  95% CI : (0.7056, 0.7541)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4059
##
##  Mcnemar's Test P-Value : 4.764e-15
##
##             Sensitivity : 0.5153
##             Specificity : 0.8703
##          Pos Pred Value : 0.7212
##          Neg Pred Value : 0.7340
##              Prevalence : 0.3943
##          Detection Rate : 0.2032
##    Detection Prevalence : 0.2817
##       Balanced Accuracy : 0.6928
##
##        'Positive' Class : 0
##

k.qda.mod2 <- train(SOURCE ~ HAEMOGLOBINS+LEUCOCYTE+THROMBOCYTE+SEX,
                    tr,method="qda",family = binomial(),
                    trControl=ctrl)
k.pred2 <- predict(k.qda.mod2,newdata = te)
confusionMatrix(data=k.pred2, te$SOURCE)

## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction   0   1
##          0 241  95
##          1 281 707
##
##                Accuracy : 0.716
##                  95% CI : (0.6909, 0.7402)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.366
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4617
##             Specificity : 0.8815
##          Pos Pred Value : 0.7173
##          Neg Pred Value : 0.7156
##              Prevalence : 0.3943
##          Detection Rate : 0.1820
##    Detection Prevalence : 0.2538
##       Balanced Accuracy : 0.6716
##
##        'Positive' Class : 0
##
```

```
k.qda.mod4 <- train(SOURCE ~ HAEMATOCRIT+LEUCOCYTE+THROMBOCYTE+SEX,
                    tr,method="qda",family = binomial(),
                    trControl=ctrl)
k.pred4 <- predict(k.qda.mod4,newdata = te)
confusionMatrix(data=k.pred4, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 233  95
##          1 289 707
##
##                Accuracy : 0.71
##                  95% CI : (0.6847, 0.7343)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.559e-15
##
##                   Kappa : 0.3507
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4464
##             Specificity : 0.8815
```

```
##          Pos Pred Value : 0.7104
##          Neg Pred Value : 0.7098
##              Prevalence : 0.3943
##          Detection Rate : 0.1760
##    Detection Prevalence : 0.2477
##       Balanced Accuracy : 0.6640
##
##        'Positive' Class : 0
##
```

```
k.qda.mod5 <- train(SOURCE ~ ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+SEX,
                    tr,method="qda",family = binomial(),
                    trControl=ctrl)
k.pred5 <- predict(k.qda.mod5,newdata = te)
confusionMatrix(data=k.pred5, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 238  93
##          1 284 709
##
##                Accuracy : 0.7153
##                  95% CI : (0.6901, 0.7394)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3632
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4559
##             Specificity : 0.8840
##          Pos Pred Value : 0.7190
##          Neg Pred Value : 0.7140
##              Prevalence : 0.3943
##          Detection Rate : 0.1798
##    Detection Prevalence : 0.2500
##       Balanced Accuracy : 0.6700
##
##        'Positive' Class : 0
##
```

```
valuesQDA <- c(0.7304,0.4059,0.5153,0.8703,
               0.716,0.366,0.4617,0.8815,
               0.71,0.3507,0.4464,0.8815,
               0.7153,0.3632,0.4559,0.884)
QDAoverview <- matrix(valuesQDA,nrow=4, ncol=4, byrow=TRUE)
colnames(QDAoverview) <- c('Accuracy (test)', 'Kappa',
```

```
'Sensitivity','Specificity')
rownames(QDAoverview) <- c('Model1', 'Model2','Model4', 'Model5')
view(QDAoverview)
print(QDAoverview)

##        Accuracy (test)  Kappa Sensitivity Specificity
## Model1          0.7304 0.4059      0.5153      0.8703
## Model2          0.7160 0.3660      0.4617      0.8815
## Model4          0.7100 0.3507      0.4464      0.8815
## Model5          0.7153 0.3632      0.4559      0.8840
```

We noticed a marginal enhancement in values when employing QDA the highest accuracy rose to 0.7304 in Model 1, which includes eight predictors. Like LDA and Logistic regression, the sensitivity and specificity values among the models are closely aligned and don't exhibit significant deviations from those observed in LDA and Logistic regression. We will still choose Model2 containing the four predictors and not Model1 We're willing to trade off a 0.02 increase in accuracy for a significant gain in interpretability.

We perform LDA and QDA using the validation set approach to compare the results with those obtained through cross-validation.

LDA validation-set approach

```
lda.mod1 <- lda(SOURCE ~
ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+MCHC+MCV+AGE+SEX,data=tr)
lda.pred1 <- predict(lda.mod1, newdata=te)
confusionMatrix(data=lda.pred1$class, te$SOURCE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 234  96
##          1 288 706
##
##                Accuracy : 0.71
##                  95% CI : (0.6847, 0.7343)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.559e-15
##
##                   Kappa : 0.3511
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4483
##             Specificity : 0.8803
##          Pos Pred Value : 0.7091
##          Neg Pred Value : 0.7103
##              Prevalence : 0.3943
##          Detection Rate : 0.1767
```

```
##    Detection Prevalence : 0.2492
##       Balanced Accuracy : 0.6643
##
##        'Positive' Class : 0
##
```

```
lda.mod2 <- lda(SOURCE ~ HAEMOGLOBINS+LEUCOCYTE+THROMBOCYTE+SEX,data=tr)
lda.pred2 <- predict(lda.mod2, newdata=te)
confusionMatrix(data=lda.pred2$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 236  95
##          1 286 707
##
##                Accuracy : 0.7122
##                  95% CI : (0.687, 0.7365)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 3.734e-16
##
##                   Kappa : 0.3564
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4521
##             Specificity : 0.8815
##          Pos Pred Value : 0.7130
##          Neg Pred Value : 0.7120
##              Prevalence : 0.3943
##          Detection Rate : 0.1782
##    Detection Prevalence : 0.2500
##       Balanced Accuracy : 0.6668
##
##        'Positive' Class : 0
##
```

```
lda.mod4 <- lda(SOURCE ~ HAEMATOCRIT+LEUCOCYTE+THROMBOCYTE+SEX,data=tr)
lda.pred4 <- predict(lda.mod4, newdata=te)
confusionMatrix(data=lda.pred4$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 228 100
##          1 294 702
##
##                Accuracy : 0.7024
##                  95% CI : (0.677, 0.7269)
```

```
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.447e-13
##
##                   Kappa : 0.3337
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4368
##             Specificity : 0.8753
##          Pos Pred Value : 0.6951
##          Neg Pred Value : 0.7048
##              Prevalence : 0.3943
##          Detection Rate : 0.1722
##    Detection Prevalence : 0.2477
##       Balanced Accuracy : 0.6560
##
##        'Positive' Class : 0
##
```

```
lda.mod5 <- lda(SOURCE ~ ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+SEX,data=tr)
lda.pred5 <- predict(lda.mod5, newdata=te)
confusionMatrix(data=lda.pred5$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 232 101
##          1 290 701
##
##                Accuracy : 0.7047
##                  95% CI : (0.6793, 0.7292)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 3.86e-14
##
##                   Kappa : 0.34
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4444
##             Specificity : 0.8741
##          Pos Pred Value : 0.6967
##          Neg Pred Value : 0.7074
##              Prevalence : 0.3943
##          Detection Rate : 0.1752
##    Detection Prevalence : 0.2515
##       Balanced Accuracy : 0.6593
##
##        'Positive' Class : 0
##
```

QDA validation-set approach

```
qda.mod1 <- qda(SOURCE ~
ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+MCHC+MCV+AGE+SEX,data=tr)
qda.pred1 <- predict(qda.mod1, newdata=te)
confusionMatrix(data=qda.pred1$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 269 104
##          1 253 698
##
##                Accuracy : 0.7304
##                  95% CI : (0.7056, 0.7541)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4059
##
##  Mcnemar's Test P-Value : 4.764e-15
##
##             Sensitivity : 0.5153
##             Specificity : 0.8703
##          Pos Pred Value : 0.7212
##          Neg Pred Value : 0.7340
##              Prevalence : 0.3943
##          Detection Rate : 0.2032
##    Detection Prevalence : 0.2817
##       Balanced Accuracy : 0.6928
##
##        'Positive' Class : 0
##
```

```
qda.mod2 <- qda(SOURCE ~ HAEMOGLOBINS+LEUCOCYTE+THROMBOCYTE+SEX,data=tr)
qda.pred2 <- predict(qda.mod2, newdata=te)
confusionMatrix(data=qda.pred2$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 241  95
##          1 281 707
##
##                Accuracy : 0.716
##                  95% CI : (0.6909, 0.7402)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                   Kappa : 0.366
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4617
##             Specificity : 0.8815
##          Pos Pred Value : 0.7173
##          Neg Pred Value : 0.7156
##              Prevalence : 0.3943
##          Detection Rate : 0.1820
##    Detection Prevalence : 0.2538
##       Balanced Accuracy : 0.6716
##
##        'Positive' Class : 0
##
```

```
qda.mod4 <- qda(SOURCE ~ HAEMATOCRIT+LEUCOCYTE+THROMBOCYTE+SEX,data=tr)
qda.pred4 <- predict(qda.mod4, newdata=te)
confusionMatrix(data=qda.pred4$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 233  95
##          1 289 707
##
##                Accuracy : 0.71
##                  95% CI : (0.6847, 0.7343)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : 1.559e-15
##
##                   Kappa : 0.3507
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4464
##             Specificity : 0.8815
##          Pos Pred Value : 0.7104
##          Neg Pred Value : 0.7098
##              Prevalence : 0.3943
##          Detection Rate : 0.1760
##    Detection Prevalence : 0.2477
##       Balanced Accuracy : 0.6640
##
##        'Positive' Class : 0
##
```

```
qda.mod5 <- qda(SOURCE ~ ERYTHROCYTE+LEUCOCYTE+THROMBOCYTE+MCH+SEX,data=tr)
qda.pred5 <- predict(qda.mod5, newdata=te)
confusionMatrix(data=qda.pred5$class, te$SOURCE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 238  93
##          1 284 709
##
##                Accuracy : 0.7153
##                  95% CI : (0.6901, 0.7394)
##     No Information Rate : 0.6057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3632
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4559
##             Specificity : 0.8840
##          Pos Pred Value : 0.7190
##          Neg Pred Value : 0.7140
##              Prevalence : 0.3943
##          Detection Rate : 0.1798
##    Detection Prevalence : 0.2500
##       Balanced Accuracy : 0.6700
##
##        'Positive' Class : 0
##
```

We're seeing identical outcomes in both cross-validation and the validation set, potentially because the number of observation is relatively large compared to the number of predictors, which could be a contributing factor to this consistency.

We split the data differently, and repeated the LDA and QDA processes, generating testing errors using both the validation set approach and cross-validation and we didn't specify any seed. Each time,we consistently obtained identical results.

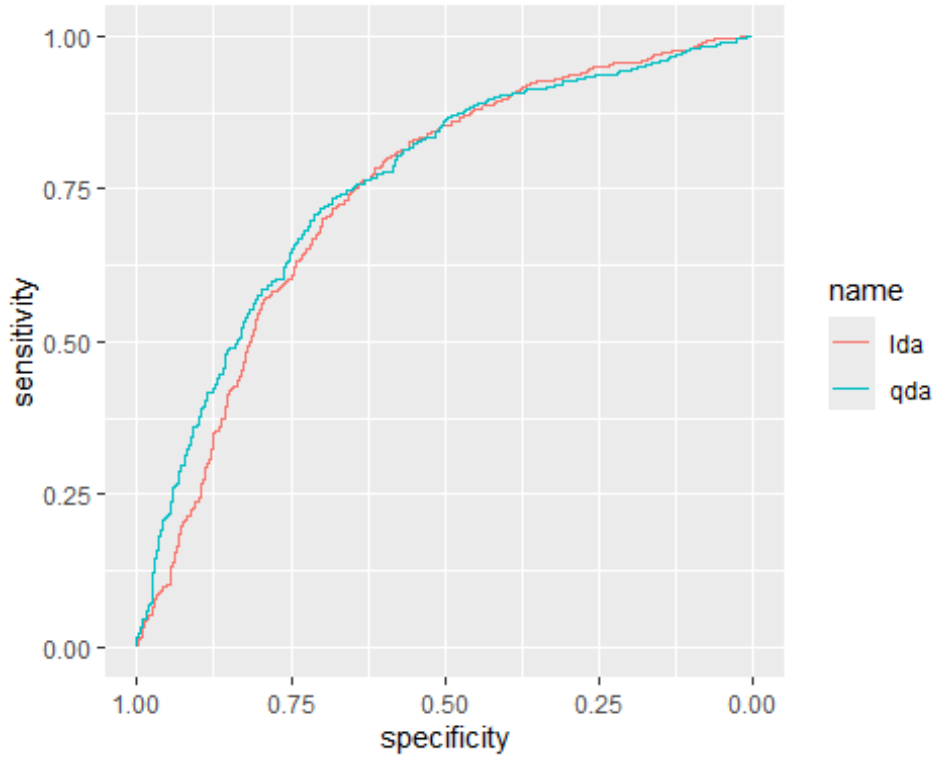
The aim of this roc-curve is to see which metod LAD or QDA performed better on our chosen model (Model2)

```
roc_lda= roc(response=te$SOURCE,
             predictor= lda.pred2$posterior[,2])
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
roc_qda= roc(response=te$SOURCE,
             predictor= qda.pred2$posterior[,2])

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

ggroc(list(lda=roc_lda,
           qda=roc_qda))
```




```
auc(roc_lda)

## Area under the curve: 0.7421

auc(roc_qda)

## Area under the curve: 0.7577
```

The area under the curve in LDA 0.7421 is lower than the area under the curve in QDA 0.7577 Hence QDA is preferable in this case

From the roc curves we concluded that QDA performs better than LDA on Model2, but we want to compare Logistic regression to these two methods as well Hence we do this following graph

```
values <- c(0.7145, 0.71, 0.7304, 0.7115, 0.7122, 0.716, 0.7062, 0.7024,
0.71, 0.713, 0.7047, 0.7153)
models <- matrix(values, nrow = 4, ncol = 3, byrow = TRUE)
colnames(models) <- c( 'GLM', 'LDA','QDA')
```

```r
rownames(models) <- c('ERY,LEU,THR,MCH,MCHC,MCV,AGE,SEX',
'HAEMO,LEU,THR,SEX','HAEMA,LEU,THR,SEX', 'ERY,LEU,THR,MCH,SEX')

lda_error <- mean(lda.pred2$class != te$SOURCE)
qda_error <- mean(qda.pred2$class != te$SOURCE)
glm_error <- mean(pred2 != te$SOURCE)

error_data <- data.frame(
  Method = c("LDA", "QDA", "GLM"),
  Error = c(lda_error, qda_error, glm_error)
)

boxplot(Error ~ Method, data = error_data, main = "Test Error Rates", ylab =
"Error Rate")
```

Our findings indicate that QDA exhibits the lowest test error rate at 0.284, while LDA and Logistic Regression demonstrate error rates of 0.2875 and 0.2885, respectively. The difference in error rate is so minimal that the use of a simpler linear method is also possible in our case.

A notable observation is that none of the pairs of predictors in Model2 (“HAEMOGLOBINS+LEUCOCYTE+THROMBOCYTE+SEX”) match the pairs we previously identified as correlated.