University of Waterloo

Final Project Report

STAT 331 Fall 2018

Group 91:

Gengyao Yuan(20613017)

Shunyu Zhao(20699637)

Contents

1 Summary

The goal of the STAT 331 final project is to explore the relation of healthy male single-fetus birth weight and some explanatory variables. This report will be divided into 4 main sections:
Summary, which will cover the main purpose of the report and give a brief explanation of how the project will analyze the data. Two candidate models will be produced in the model selection section by using the pre-fitting data diagnostic and automated model selection. Model diagnostics section will perform an in-depth comparison of the two candidates models by comparing different types of residual plots, leverage and influence measures and cross-validation(rPMSE). In the end, there will be a discussion section basing on the result of the most likely linear model we get from the previous sections to talk about several topics such like: ???what is the most important factors associated with/influencing birth weight????
After using serial statistical analysis way that we learned in STAT 331 course, we find that

2 Model Selection

2.1 Brief data overview

By view the summary of the data, we notice that there are several illegal data. The domain of "marital"(the mother's marital status) is 1 to 5, but it is clearly showing that there exist 0 in "marital," we replace all the 0 to NA since it is not available data(out of range).
For the categorical predictor "meth"(The self-reported ethnicity of the mother) and "feth"(The self-reported ethnicity of the father), all 0 to 5 is Caucasian meaning that they are in the same group, so we replace the 0-5(Caucasian) to 0, 6(Mexican) to 1, 7(African-American) to 2, 8(Asian) to 3, 9(Mixed) to 4, 10(Other) to 5

2.2 Transform categorical predictors

Since all the categorical predictors should not be treated as continuous variables, although they may look like continuous variables (such like 0,1,2,3,4....), we use "one-hot" encoding scheme to make new factors for them. For example, 'med' means the mother's education, whose domain is 0 to 7, where '0' level means 'elementary school' level, level '1' means 'middle school' level, level '2' means 'high school' .... etc, we just transfer numbers to factors with the same name(for example, number 1 to NEW factor '1'), after successfully transfer all the levels to new factors, dropping all the '0' levels for all categorical predictors by the requirement of one-hot"encoding scheme(we can do that because all predictors have '0' level(meth and feth didn't have, but we already transform them)), and give all the new factors a 0/1 binary variable to show that factor is applied to this data or not.
There is a trick in R code:"as.factor" function, it can automatically transfer variables to new factors, so we applied it on all the categorical predictors(meth, med,feth, fed, marital, smoke, time, number) to factor type instead of continuous variables.

2.3 Drop off NAs and prediction missing data

From the summary report in the previous section, there are lots of NA data points in our data frame. Several methods have been covered in STAT 331 to produce missing data points. However, we use MICE here.
MICE can help us to impute missing values which are drawn from a distribution specifically designed for each missing data points. Don't like replace all NA variables by mean of the data; MICE can also include the 'var' in the data prediction, which can help lessen the bias and make the data close to the original.
Since MCIE is 'prediction' function, thus every time we run this may cause different results, to avoid this, we always set the seed as 1. And to get a closer result similar to the original data, we set the method type of MICE to 'sample,' which means sample any Random sample from observed values.

The result of anyNA is FALSE shows there is no more NA value in our births_mice data frame.
Since there is no +-INF's data basing on our summary, all the data in data frame now is available and meaningful.

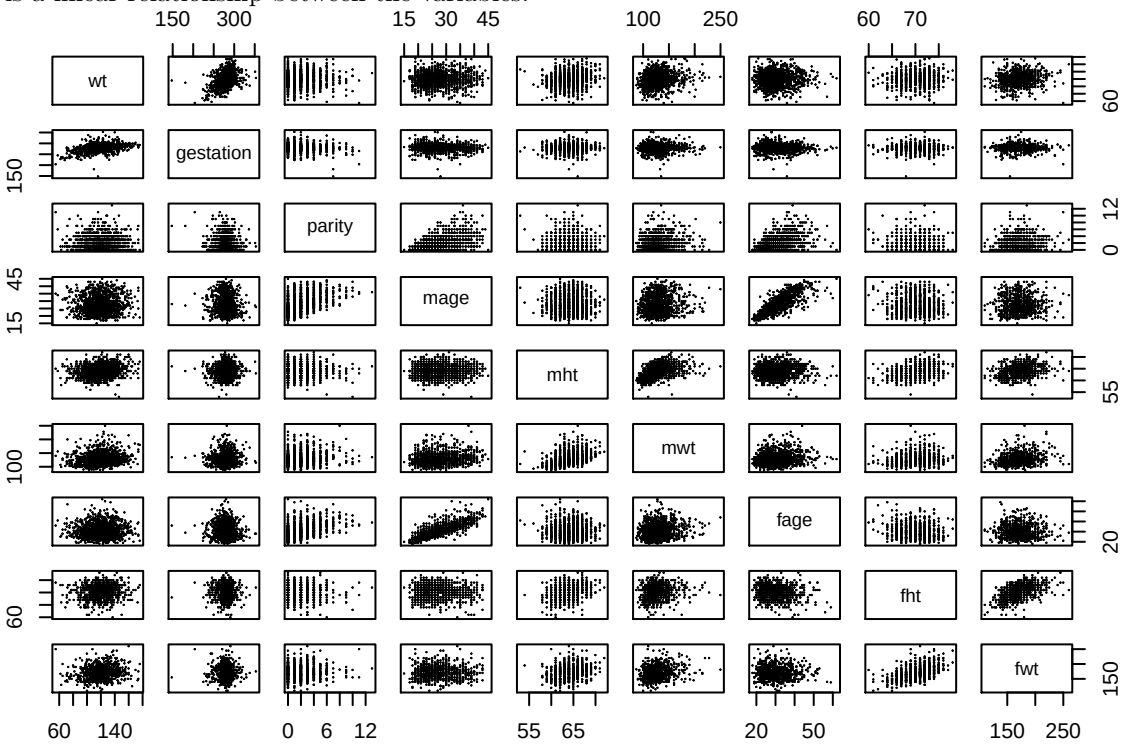
2.4 Revisiting the category variables

After predicting the missing NA data points, it is necessary to revisit categorial variables and factorize the levels of categorical variables into meaningful names; this can be helpful when dueling with interpretation effect of the model.
We also shrink the number of levels for some factors because those levels are significant minorities:
meth & feth: keep 0 as Caucasian, 2 as African-American, shrink all other to other(this change based on the previous shrunk result.) med & fed: keep 1 as middle school, 2 as high school, 3 as high school+trade school, 5 as a college graduate, shrink all other to other marital: keep 1 as married, shrink all the others to other time: keep 0 as never smoke, 1 as still smokes, shrink all the others to other number: keep 0 as never smoked, 1 as (smoke) 1-4 (per day), 2 as 5-9,5 as 20-29, shrink all the others to othera

During we shrinking the varibles, we find that the 'smoke' factor should be exaclty same as the 'time smoke' factor, so we just delete the factor'smoke'

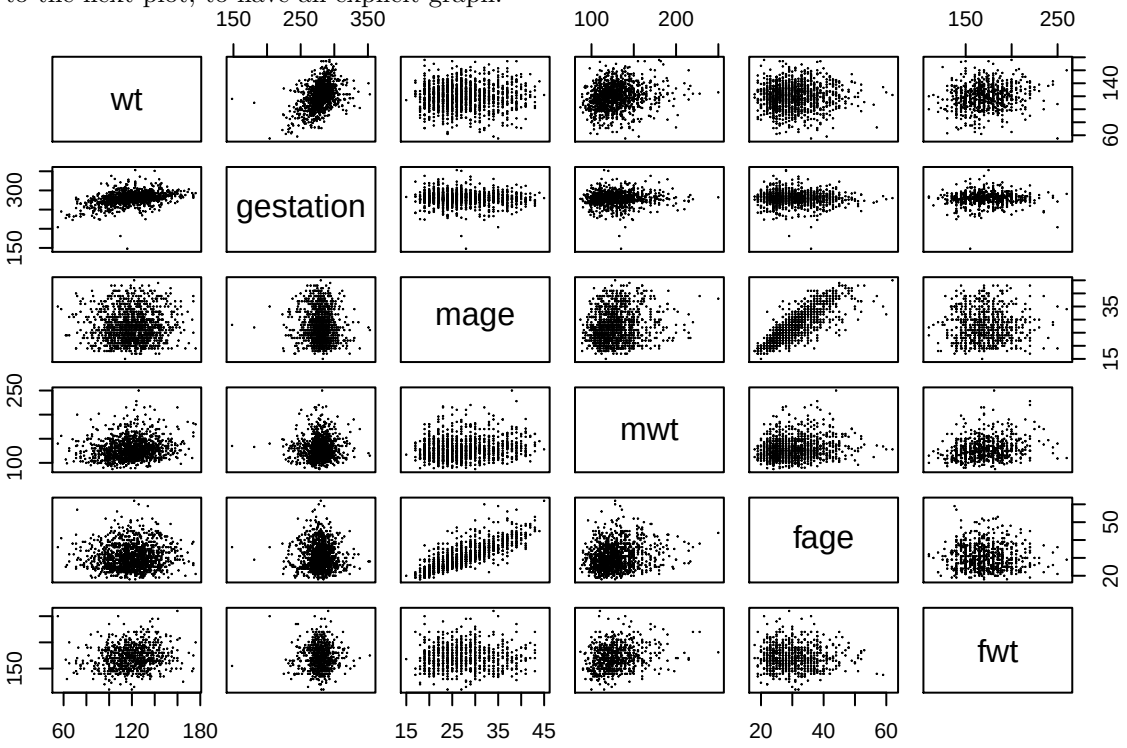
2.5 visually inspect data

By drawing out the pairs plot of the original data, we can have a basic visually inspect to estimation is there is a linear relationship between the variables.



As the reason that we only what to find out if there are linear relationships, we only include 'continuous varibles'(basing on the data definition of the project question) into our graph. The result shows that all the 'continuous varibles' continuously somehow so we don't need to treat any of them as categorical.
Since there are too many variables, we pike up some significant variables that may have a linear relationship

to the next plot, to have an explicit graph.



The only clear linear relationship is between wt and gestation. All the other variables should be further discussed.

2.6 Automated Model Selection

2.6.1 min & max model set up

It is necessary to set up a minimum model and maximum model before using the automatic selections. Firstly, we set up the M0 as min with only interaction and Mmax as the maximum that all variables have interactions with each other.

Since the NA chart shows it is obviose most of the coeficients have NA interactions with marital, fed, feth, number, time, meth, med, thus we delete all the interaction with them in our max in order to have a smaller model. We don't add any quadratic terms because basing on the previous pairs plot, there is no graph seems have quadratic relationship.

```
Mmax <- lm(wt ~ (. -marital -fed -feth -number -time - meth -med)^2
           +marital + fed +feth +number +time + meth +med , data = births_clean)
Mstart <- lm(wt ~ ., data = births_clean)
anyNA(coef(Mmax))
```

```
## [1] FALSE
```

The output of anyNA is FALSE. It shows the Mmax is the minimum model which including as many possible interactions can but can also avoid all NA here.

2.6.2 display covariates in each model

```
invisible(Mfwd <- step(object = M0,
               scope = list(lower = M0, upper = Mmax),
               direction = "forward", trace = FALSE))
```

```
Mback <- step(object = Mmax,
              scope = list(lower = M0, upper = Mmax),
              direction = "backward", trace = FALSE)
```

```
Mstep <- step(object = Mstart,
              scope = list(lower = M0, upper = Mmax),
              direction = "both", trace = FALSE)
```

```
##  fwd back step
##   20   33   25
```

```
## lm(formula = wt ~ gestation + time + mht + meth + parity + number +
##     fwt + mwt + fht + gestation:fwt + gestation:mwt + mht:fwt +
##     gestation:mht + gestation:fht, data = births_clean)
```

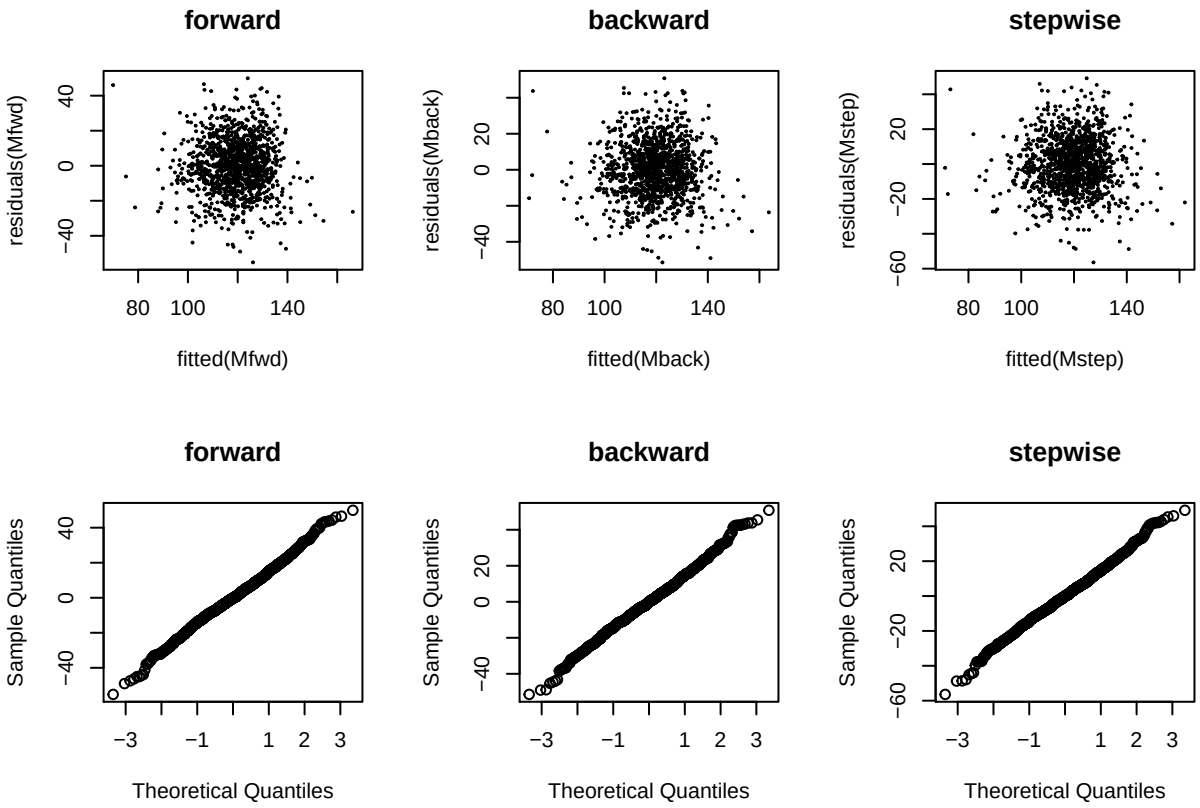
```
## lm(formula = wt ~ gestation + parity + mage + mht + mwt + fage +
##     fht + fwt + income + number + time + meth + gestation:mage +
##     gestation:mht + gestation:mwt + gestation:fht + gestation:fwt +
##     gestation:income + parity:mht + mht:fage + mht:fht + mht:fwt +
##     mht:income + mwt:income + fage:fwt + fage:income + fht:income,
##     data = births_clean)
```

```
## lm(formula = wt ~ gestation + parity + meth + mage + mht + mwt +
##     fht + fwt + income + time + number + gestation:income + mwt:income +
##     gestation:mage + gestation:fwt + gestation:mwt + gestation:fht +
##     gestation:mht + fht:income, data = births_clean)
```

The first line of output is the number of parameters for the three models. Parameter for every models are showing after that, following the order: forward selection, backward elimination, and stepwise selection.

2.6.3 qqplot for residual distribution

```
par(mfrow=c(2,3))
plot(fitted(Mfwd), residuals(Mfwd), main="forward", cex = .2)
plot(fitted(Mback), residuals(Mback), main="backward", cex = .2)
plot(fitted(Mstep), residuals(Mstep), main="stepwise", cex =.2)
qqnorm(residuals(Mfwd), main="forward")
qqnorm(residuals(Mback), main="backward")
qqnorm(residuals(Mstep), main="stepwise")
```



From the Residual vs Fitted plot, it is clearly showing that points are randomly distributed around the 0 line. And the points in QQ-plot almost line on the diagonal line. Both of the two graphs prove that the residual distribution follow normal distribution.

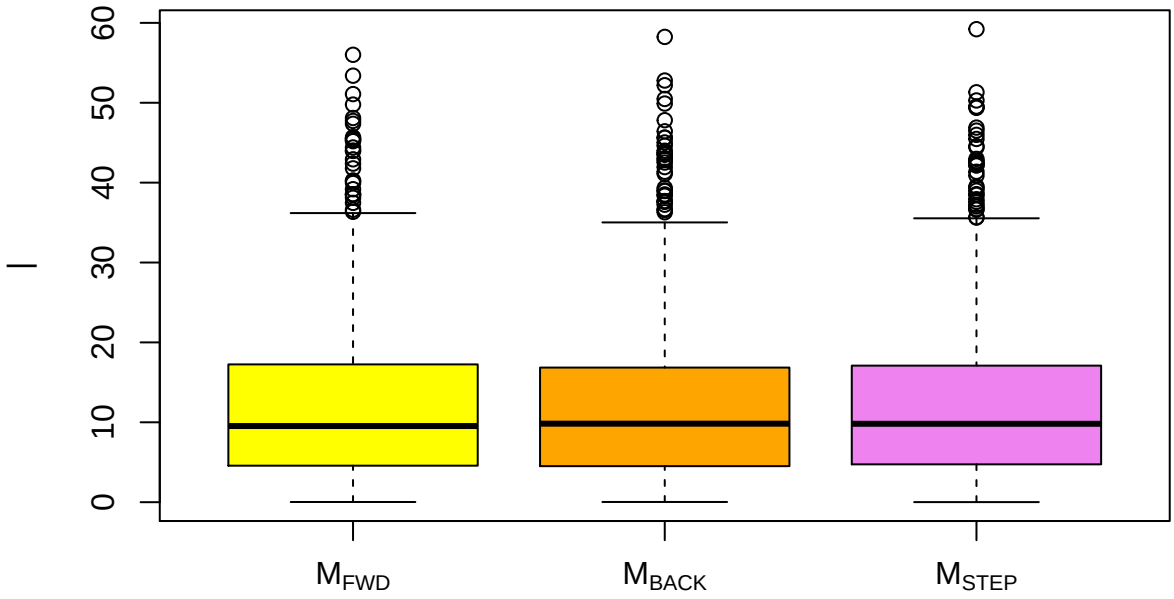
2.6.4 Press AIC and R^2

```
M1 <- Mfwd
M2 <- Mback
M3 <- Mstep
Mnames <- expression(M[FWD], M[BACK], M[STEP])

# press for 3 automated models
press1 <- resid(M1)/(1-hatvalues(M1))
press2 <- resid(M2)/(1-hatvalues(M2))
press3 <- resid(M3)/(1-hatvalues(M3))
PRESS = c(sum(press1^2), sum(press2^2), sum(press3^2))

# R^2 for 3 automated models
r_square1 <- summary(Mfwd)$r.squared
r_square2 <- summary(Mback)$r.squared
r_square3 <- summary(Mstep)$r.squared
R_Squared <- c(r_square1,r_square2,r_square3)

# AIC for 3 automated models
AIC1 <- AIC(M1)
AIC2 <- AIC(M2)
AIC3 <- AIC(M3)
```

```
AIC = c(AIC1,AIC2,AIC3)

# display results
disp <- rbind(AIC,PRESS,R_Squared)
colnames(disp) <- Mnames
disp
```

```
##                 M[FWD]      M[BACK]     M[STEP]
## AIC       1.028290e+04 1.026394e+04  10263.5517
## PRESS     2.986716e+05 2.955627e+05 294200.3280
## R_Squared 3.012018e-01 3.261652e-01      0.3176
```

```
#plot PRESS statistics
boxplot(x = list(abs(press1),abs(press2),abs(press3)), names = Mnames,
        ylab = expression("|", PRESS[i], "|"),col = c("yellow","orange","violet"))
```



Since the press statics equal to the following equation

$$PRESS_i = y_i - y_i = e_i/1 - h_i$$

The 'e' here is the residual error, which means the sum of i PRESSi is the total residual error of the model. Thus the model with lest PRESS is the best model.
Akaike Information Criterion(AIC) equal to

$$AIC = n(1 + log(e^{'}e/n) + log(2pi)) + 2(p + 1)$$

The less AIC means better model with less error because the AIC has the same monotony with 'e'. And by the defination of residual error, R^2 is also less is better. According to the result we get above, Mback(model get by backward elimination) is the best model by automated model section.

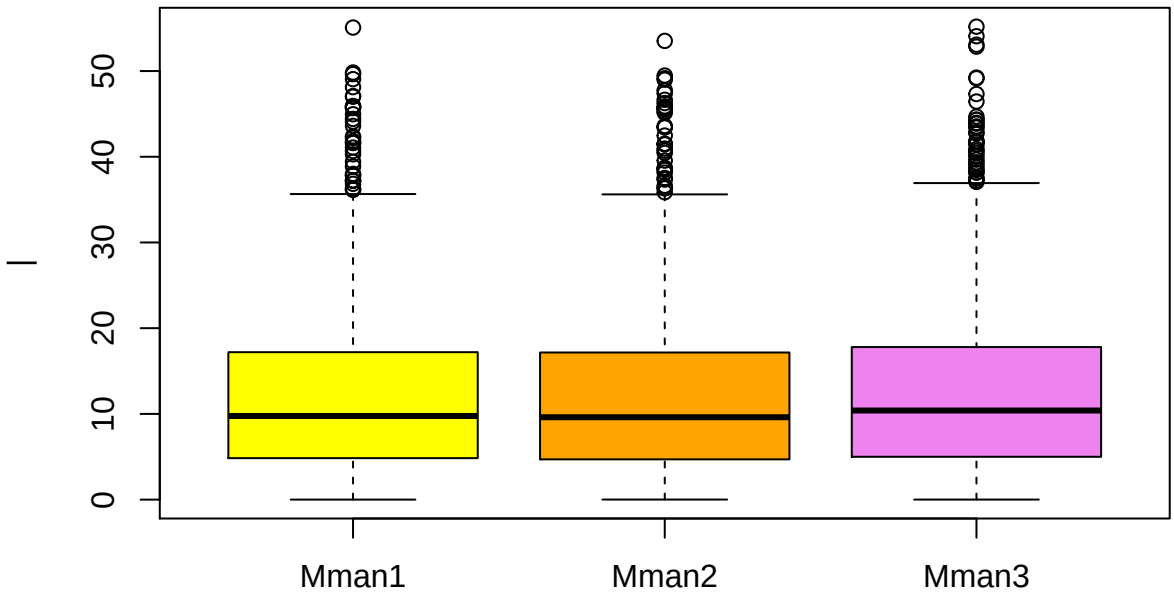
2.6.5 Manual Model

```
Mman1 <- lm(formula = wt ~ gestation + time + mht + meth + parity + number +
    fwt + mwt + fht,  data = births_clean)
Mman2 <- lm(formula = wt ~ gestation + parity + meth + mage + mht + mwt +
```

```
    fht + fwt + income + time + number,  data = births_clean)
Mman3 <- lm(formula = wt ~ gestation + mage + fage  + time + number + meth + feth,
            data = births_clean)
Mnames <- expression(Mman1, Mman2, Mman3)

# press for 3 automated models
press1 <- resid(Mman1)/(1-hatvalues(Mman1))
press2 <- resid(Mman2)/(1-hatvalues(Mman2))
press3 <- resid(Mman3)/(1-hatvalues(Mman3))
PRESS = c(sum(press1^2), sum(press2^2), sum(press3^2))

# R^2 for 3 automated models
r_square1 <- summary(Mman1)$r.squared
r_square2 <- summary(Mman2)$r.squared
r_square3 <- summary(Mman3)$r.squared
R_Squared <- c(r_square1,r_square2,r_square3)

# AIC for 3 automated models
AIC1 <- AIC(Mman1)
AIC2 <- AIC(Mman2)
AIC3 <- AIC(Mman3)
AIC = c(AIC1,AIC2,AIC3)

# display results
disp <- rbind(AIC,PRESS,R_Squared)
colnames(disp) <- Mnames
disp
```

```
##                  Mman1        Mman2        Mman3
## AIC       1.029514e+04 1.029669e+04 1.038430e+04
## PRESS     3.002646e+05 3.006585e+05 3.221790e+05
## R_Squared 2.885142e-01 2.899263e-01 2.340586e-01
```

```
#plot PRESS statistics
boxplot(x = list(abs(press1),abs(press2),abs(press3)), names = Mnames,
        ylab = expression("|", PRESS[i], "|"),col = c("yellow","orange","violet"))
```



3 Model Diagnostics

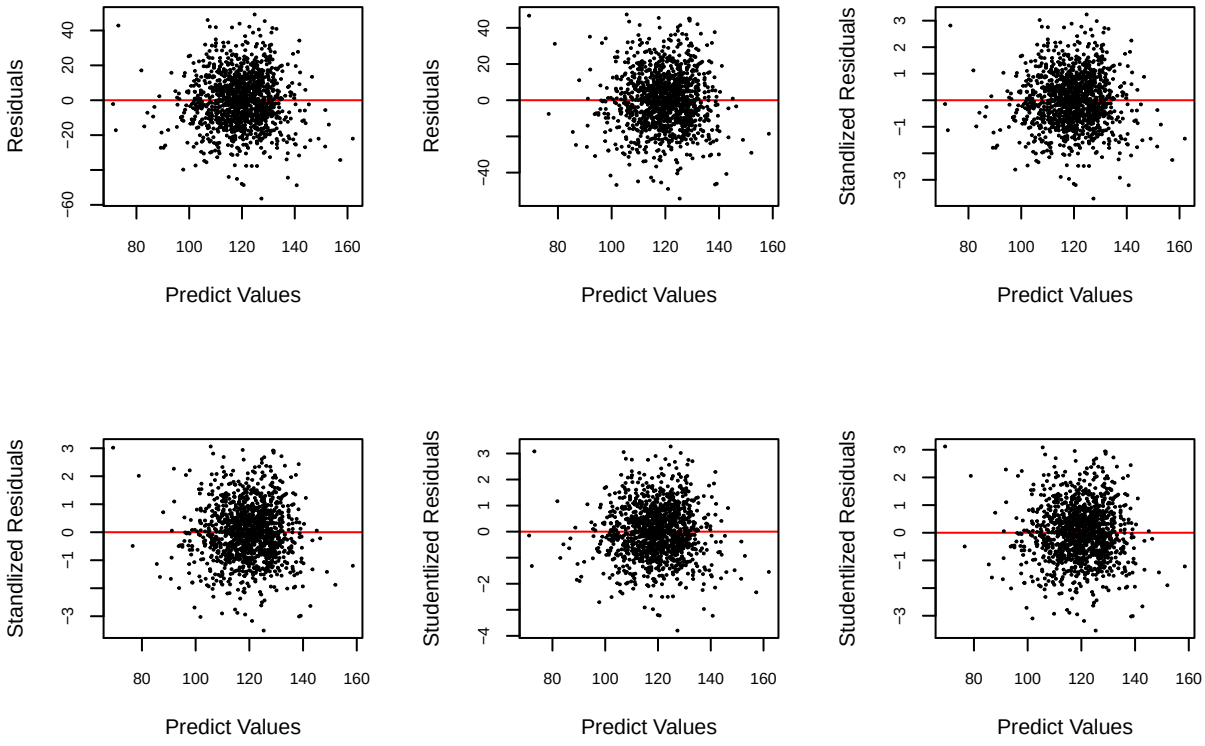
```
Model1 <- Mstep
Model2 <- Mman1

h1 <- hatvalues(Model1)
h2 <- hatvalues(Model2)

y1.hat <- predict(Model1)
y2.hat <- predict(Model2)
```

3.1 Different types of residual plots

3.2 Residuals studentlized residuals and standlized residuals

```
Re1 <- residuals(Model1)
Re2 <- residuals(Model2)

StanRe1 <- Re1/sigma(Model1)
StanRe2 <- Re2/sigma(Model2)

StudRe1 <- StanRe1 / sqrt(1-hatvalues(Model1))
StudRe2 <- StanRe2 / sqrt(1-hatvalues(Model2))


par(mfrow=c(2,3))
## Residual plots
plot(predict(Model1), Re1, xlab = "Predict Values", ylab = "Residuals", cex.axis = .8, cex = .2,
     abline(h = mean(Re1), col = "red"))
plot(predict(Model2), Re2, xlab = "Predict Values", ylab = "Residuals", cex.axis = .8, cex = .2,
     abline(h = mean(Re2), col = "red"))
```

```r
## Standlized residual plots
plot(predict(Model1), StanRe1, xlab = "Predict Values", ylab = "Standlized Residuals", cex.axis = .8,
     cex = .2, abline(h = mean(StanRe1), col = "red"))
plot(predict(Model2), StanRe2, xlab = "Predict Values", ylab = "Standlized Residuals", cex.axis = .8,
     cex = .2, abline(h = mean(StanRe2), col = "red"))

## Studentlized residuals plots
plot(predict(Model1), StudRe1, xlab = "Predict Values", ylab = "Studentlized Residuals", cex.axis = .8,
     cex = .2, abline(h = mean(StudRe1), col = "red"))
plot(predict(Model2), StudRe2, xlab = "Predict Values", ylab = "Studentlized Residuals", cex.axis = .8,
     cex = .2, abline(h = mean(StudRe2), col = "red"))
```



3.3 PRESS Residuals

```r
press_model1 <- Re1/(1 - hatvalues(Model1))
press_model2 <- Re2/(1 - hatvalues(Model2))
```

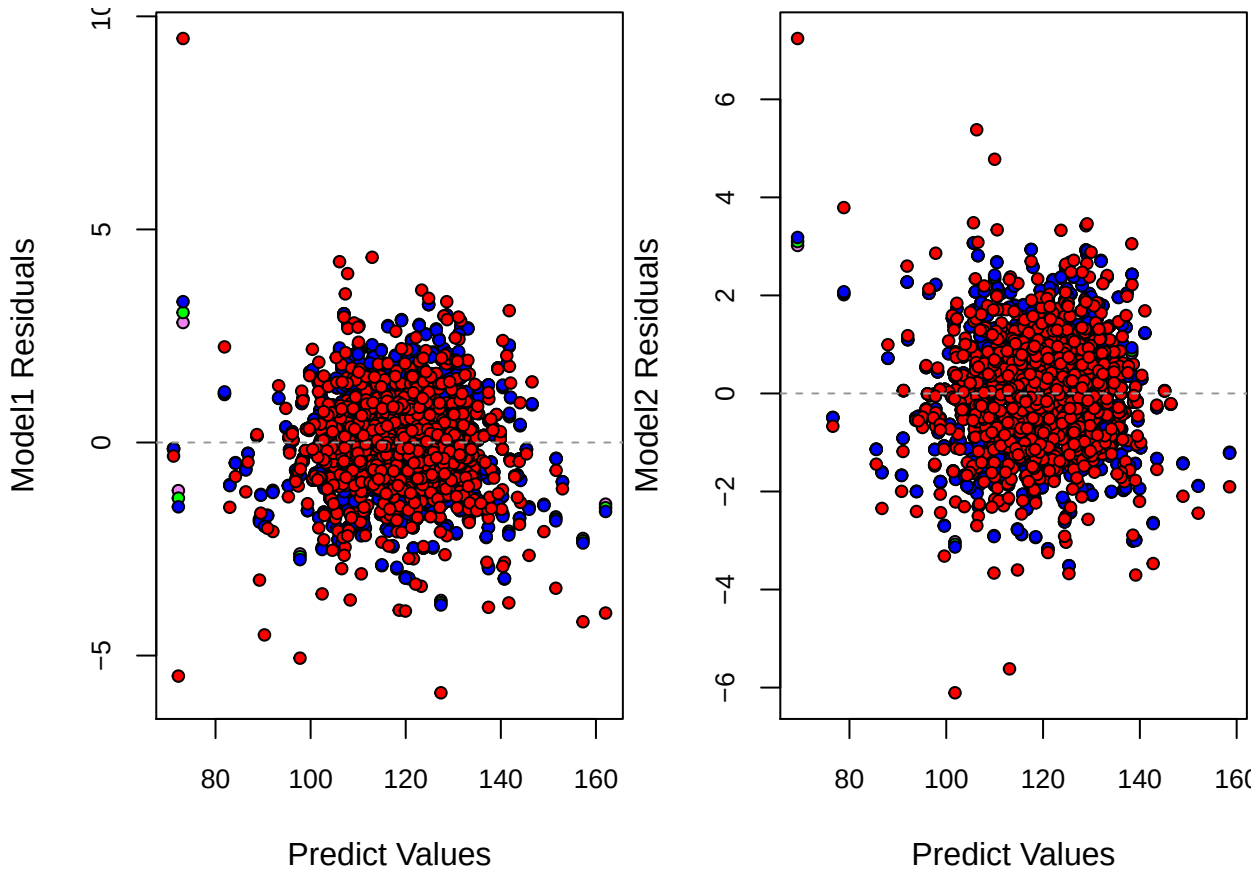
3.4 DFFITS Residuals

```r
dfts1 <- dffits(Model1)
dfts2 <- dffits(Model2)
```

3.5 Comparison of different residual plots

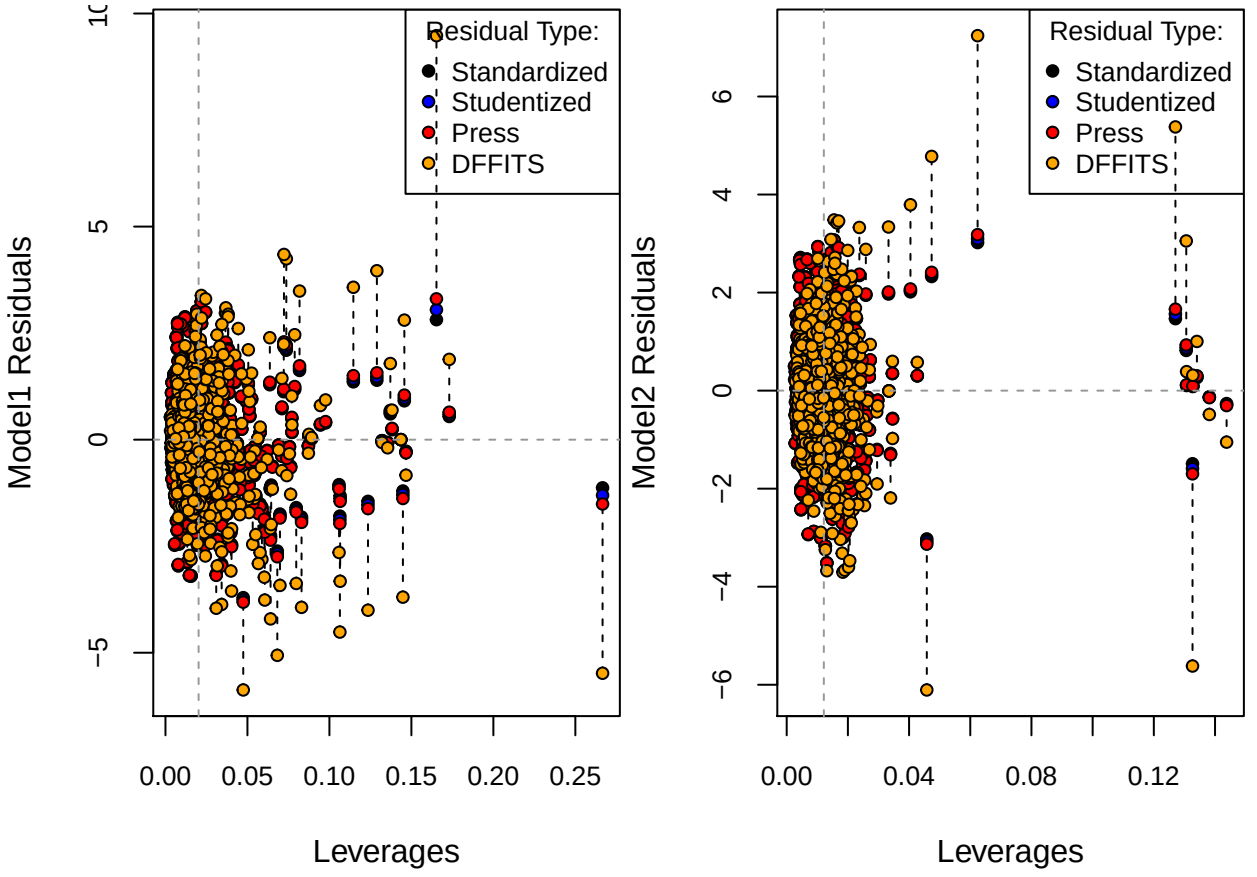
```
# standlize each of these
p1 <- length(coef(Model1))
n1 <- nobs(Model1)
hbar1 <- p1/n1
StudRe1.stan <- StudRe1 * sqrt(1-hbar1)
press_model1.stan <- press_model1*(1-hbar1)/sigma(Model1)
dfts1.stan <- dfts1*(1-hbar1)/sqrt(hbar1)


# plots all
par(mfrow = c(1,2), mar = c(4,4,.1,.1))
plot(predict(Model1), rep(0, length(predict(Model1))),
     type = "n",
     ylim = range(StanRe1,StudRe1.stan,dfts1.stan,press_model1.stan),
     xlab = "Predict Values",
     ylab = "Model1 Residuals",
     cex.axis = .8)
segments(x0 = h1,
        y0 = pmin(StanRe1, StudRe1.stan, press_model1.stan, dfts1.stan),
        y1 = pmax(StanRe1, StudRe1.stan, press_model1.stan, dfts1.stan),
        lty = 2)
points(predict(Model1), StanRe1, pch = 21, bg = "violet", cex = .8)
points(predict(Model1), StudRe1.stan, pch = 21, bg = "green", cex = .8)
points(predict(Model1), press_model1.stan, pch = 21, bg = "blue", cex = .8)
points(predict(Model1), dfts1.stan, pch = 21, bg = "red", cex = .8)
abline(h = 0, col = "grey60", lty =2) #horizontal line

## model2
# standlize each of these
p2 <- length(coef(Model2))
n2 <- nobs(Model2)
hbar2 <- p2/n2
StudRe2.stan <- StudRe2 * sqrt(1-hbar2)
press_model2.stan <- press_model2*(1-hbar2)/sigma(Model2)
dfts2.stan <- dfts2*(1-hbar2)/sqrt(hbar2)

# plots all
plot(predict(Model2), rep(0, length(predict(Model2))),
     type = "n",
     ylim = range(StanRe2,StudRe2.stan,dfts2.stan,press_model2.stan),
     xlab = "Predict Values",
     ylab = "Model2 Residuals",
     cex.axis = .8)
segments(x0 = h2,
        y0 = pmin(StanRe2, StudRe2.stan, press_model2.stan, dfts2.stan),
        y1 = pmax(StanRe2, StudRe2.stan, press_model2.stan, dfts2.stan),
        lty = 2)
points(predict(Model2), StanRe2, pch = 21, bg = "violet", cex = .8)
points(predict(Model2), StudRe2.stan, pch = 21, bg = "green", cex = .8)
points(predict(Model2), press_model2.stan, pch = 21, bg = "blue", cex = .8)
points(predict(Model2), dfts2.stan, pch = 21, bg = "red", cex = .8)
```

```
abline(h = 0, col = "grey60", lty =2) #horizontal line
```




```
# plots Residuals vs Leverages
# model1
plot(h1, rep(0, length(y1.hat)), type = "n", cex.axis = .8,
     ylim = range(StanRe1, StudRe1.stan, press_model1.stan, dfts1.stan),
     xlab = "Leverages", ylab = "Model1 Residuals")
segments(x0 = h1,
        y0 = pmin(StanRe1, StudRe1.stan, press_model1.stan, dfts1.stan),
        y1 = pmax(StanRe1, StudRe1.stan, press_model1.stan, dfts1.stan),
        lty = 2)
points(h1, StanRe1, pch = 21, bg = "black", cex =.8)
points(h1, StudRe1.stan, pch = 21, bg = "blue", cex =.8)
points(h1,press_model1.stan, pch = 21, bg = "red", cex = .8)
points(h1,dfts1.stan, pch = 21, bg = "orange", cex = .8)
abline(v = hbar1, col = "grey60", lty = 2)
abline(h = 0, col = "grey60", lty =2) #horizontal line
legend("topright",legend = c("Standardized", "Studentized","Press","DFFITS"), pch = 21, pt.bg = c("blac

# model2
plot(h2, rep(0, length(y2.hat)), type = "n", cex.axis = .8,
     ylim = range(StanRe2, StudRe2.stan, press_model2.stan, dfts2.stan),
     xlab = "Leverages", ylab = "Model2 Residuals")
segments(x0 = h2,
        y0 = pmin(StanRe2, StudRe2.stan, press_model2.stan, dfts2.stan),
        y1 = pmax(StanRe2, StudRe2.stan, press_model2.stan, dfts2.stan),
```

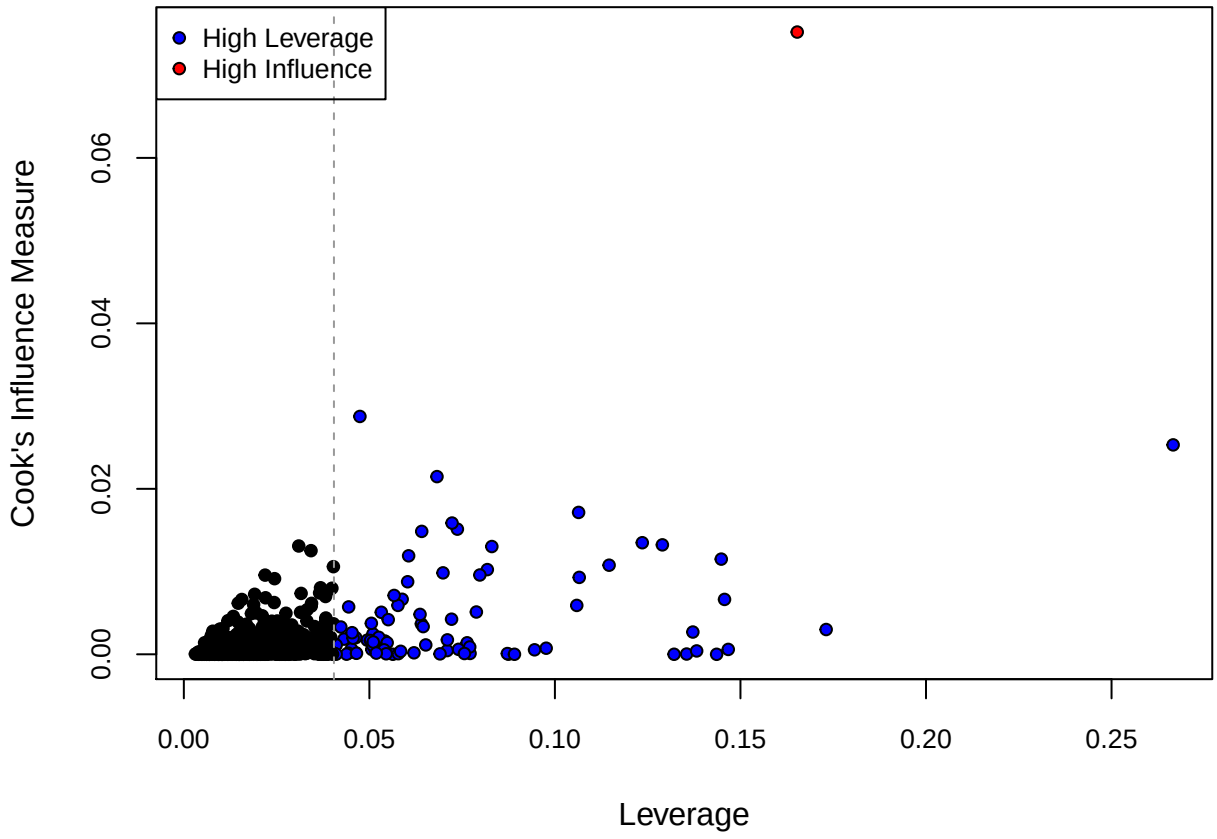
```
       lty = 2)
points(h2, StanRe2, pch = 21, bg = "black", cex =.8)
points(h2, StudRe2.stan, pch = 21, bg = "blue", cex =.8)
points(h2,press_model2.stan, pch = 21, bg = "red", cex = .8)
points(h2,dfts2.stan, pch = 21, bg = "orange", cex = .8)
abline(v = hbar2, col = "grey60", lty = 2)
abline(h = 0, col = "grey60", lty =2) #horizontal line
legend("topright",legend = c("Standardized", "Studentized","Press","DFFITS"), pch = 21, pt.bg = c("blac
```



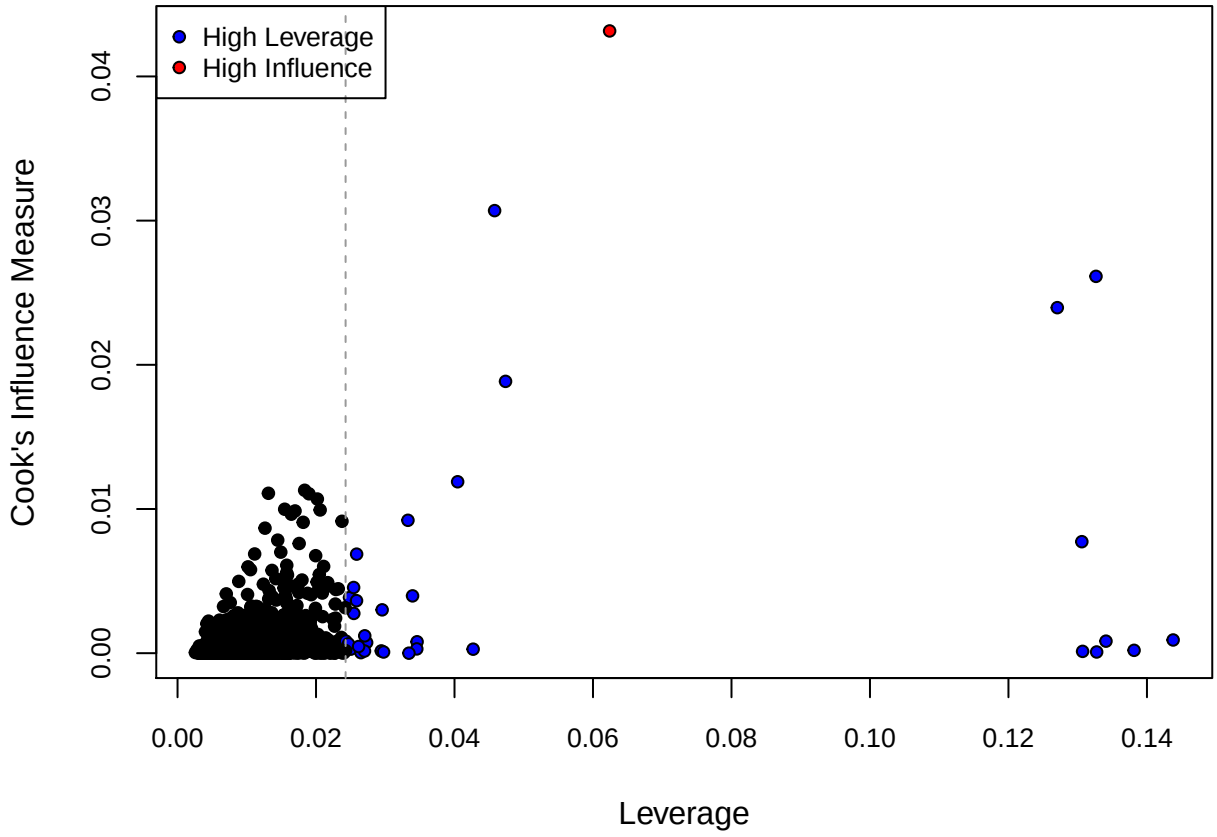
3.6 Leverage and influence measures

```
# compute leverage

h1 <- hatvalues(Model1)
h2 <- hatvalues(Model2)

D1 <- cooks.distance(Model1)
D2 <- cooks.distance(Model2)

infl1.ind <- which.max(D1)
infl2.ind <- which.max(D2)

lev1.ind <- h1 > 2*hbar1
lev2.ind <- h2 > 2*hbar2
```

```
clrs <- rep("black", len = nobs(Model1))
clrs[lev1.ind] <- "blue"
clrs[infl1.ind] <-"red"
par(mfrow = c(1,1), mar = c(4,4,1,1))
cex <- .8
plot(h1,D1,xlab = "Leverage", ylab = "Cook's Influence Measure", pch = 21, bg = clrs, cex = cex, cex.axi
abline(v = 2*hbar1, col = "grey60", lty = 2)
legend("topleft", legend = c("High Leverage", "High Influence"), pch = 21, pt.bg = c("blue", "red"),cex
```



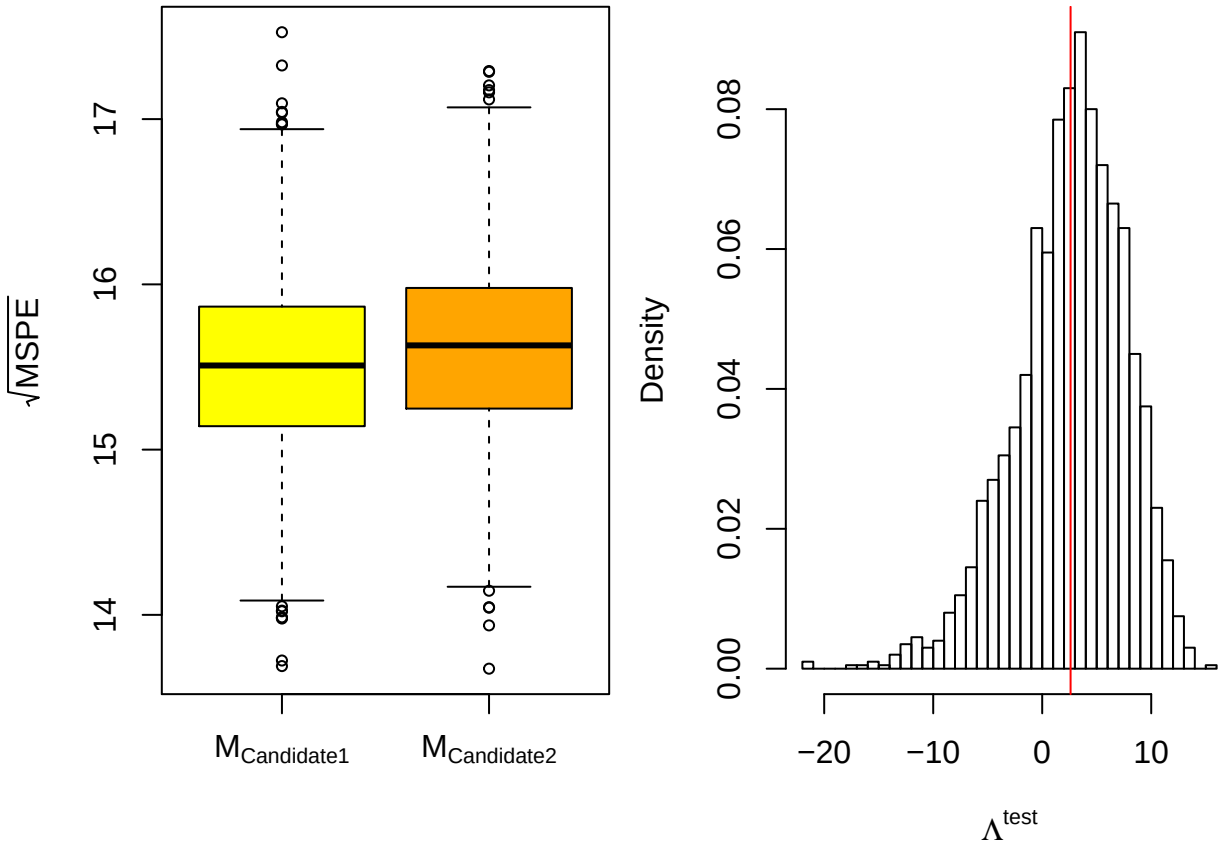

```
clrs <- rep("black", len = nobs(Model1))
clrs[lev2.ind] <- "blue"
clrs[infl2.ind] <-"red"
par(mfrow = c(1,1), mar = c(4,4,1,1))
cex <- .8
plot(h2,D2,xlab = "Leverage", ylab = "Cook's Influence Measure", pch = 21, bg = clrs, cex = cex, cex.axi
abline(v = 2*hbar2, col = "grey60", lty = 2)
legend("topleft", legend = c("High Leverage", "High Influence"), pch = 21, pt.bg = c("blue", "red"),cex
```



3.7 Cross-validation

```
M1 <- Model1
M2 <- Model2
Mnames_new <- expression(M[Candidate1],M[Candidate2])
# Cross-validation setup
nreps <- 2e3 # number of replications
ntot <- nrow(births_clean) # total number of observations
ntrain <- floor(0.7 * ntot) # size of training set
ntest <- ntot-ntrain # size of test set
mspe1 <- rep(NA, nreps) # sum-of-square errors for each CV replication
mspe2 <- rep(NA, nreps)
logLambda <- rep(NA, nreps) # log-likelihod ratio statistic for each replication
for(ii in 1:nreps) {
  # randomly select training observations
  train.ind <- sample(ntot, ntrain) # training observations
  # refit the models on the subset of training data; ?update for details!
  M1.cv <- update(M1, subset = train.ind)
  M2.cv <- update(M2, subset = train.ind)
  # out-of-sample residuals for both models
  # that is, testing data - predictions with training parameters
  M1.res <- births_clean$wt[-train.ind] -
            predict(M1.cv, newdata = births_clean[-train.ind,])
  M2.res <- births_clean$wt[-train.ind] -
            predict(M2.cv, newdata = births_clean[-train.ind,])
```

```r
  # mean-square prediction errors
  mspe1[ii] <- mean(M1.res^2)
  mspe2[ii] <- mean(M2.res^2)
  # out-of-sample likelihood ratio
  M1.sigma <- sqrt(sum(resid(M1.cv)^2)/ntrain) # MLE of sigma
  M2.sigma <- sqrt(sum(resid(M2.cv)^2)/ntrain)
  # since res = y - pred, dnorm(y, pred, sd) = dnorm(res, 0, sd)
  logLambda[ii] <- sum(dnorm(M1.res, mean = 0, sd = M1.sigma, log = TRUE))
  logLambda[ii] <- logLambda[ii] -
                   sum(dnorm(M2.res, mean = 0, sd = M2.sigma, log = TRUE))
}
# plot rMSPE and out-of-sample log
par(mfrow = c(1,2))
par(mar = c(4.5, 4.5, .1, .1))
boxplot(x = list(sqrt(mspe1), sqrt(mspe2)),
        names = Mnames_new, cex = .7,
        ylab = expression(sqrt(MSPE)), col = c("yellow", "orange"))
hist(logLambda, breaks = 50, freq = FALSE,
     xlab = expression(Lambda^{test}),
     main = "", cex = .7)
abline(v = mean(logLambda), col = "red") # average value
```



4 Discussion

4.1 What are the most important factors associated with/influencing birth weight?

The most important factors are

4.2 Low birth weight is considered to be 88 ounces or less. Based on this analysis, wouldyou be able to recommend behavioral changes to parents in order to avoid low birthweight? If so, please carefully formulate your recommendation.

4.3 Are there any coecients with high p-values retained in the final model? If so, why?

YES, there are sererval coecients with high p-value remain in the model, because we considering the interaction between
the variables, so it-self may have high p-value but its interaction with others is fine. \subsection{Are there any outlying observations that might be appropriate to remove?

4.4 Are any of the regression assumptions of the final model violated? If so, which ones?

4.5 What are the possible deficiencies of the final model? how do these deficienciesnuance your conclusions/recommendations above? conclusions/recommendations above?