**FILE:OnlyNumerical+logY**

setwd('C:/users/mg/Desktop/Data Analytics/HW/HW7/')

getwd()

data = read.table("HW7\_AutoMPG.csv", header=T, sep=',')

head(data)

# In this case, I only used numerical variables (without cylinders, year, origin)

######################################################################################################################

# Purpose of this section : Change discrete values as categorical values. Solve missing values problem.

library(dummies)

mydata = data

# Multi-valued discrete values can be a categorical values.

mydata = dummy.data.frame(mydata,names=c("cylinders"))

head(mydata)

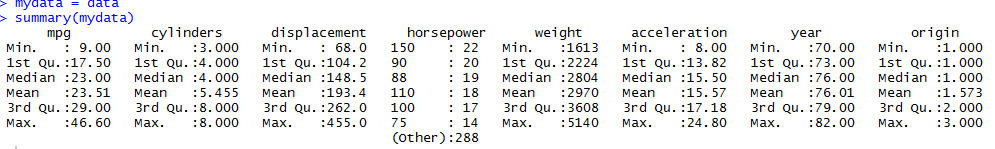
mydata = dummy.data.frame(mydata,names=c("year"))

head(mydata)

mydata = dummy.data.frame(mydata,names=c("origin"))

head(mydata)

summary(mydata)

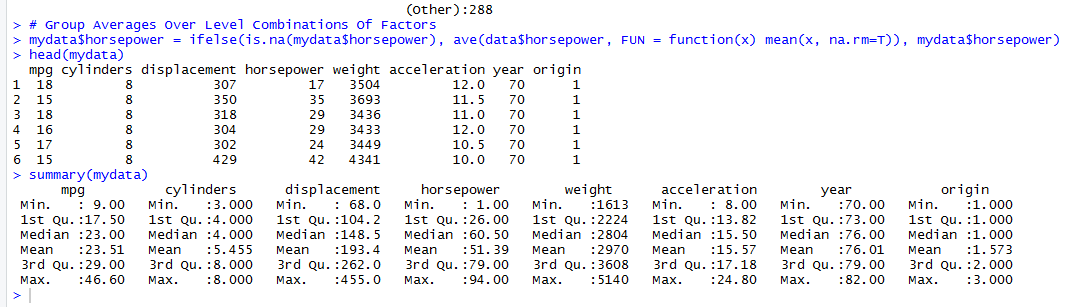


# Group Averages Over Level Combinations Of Factors

mydata$horsepower = ifelse(is.na(mydata$horsepower), ave(data$horsepower, FUN = function(x) mean(x, na.rm=T)), mydata$horsepower)

head(mydata)

summary(mydata)



######################################################################################################################

# Purpose of this section : Variable declarations and examine relationships to create models.

# y variable = mpg, x variables = other 7 variables.

nrow(mydata) # data size = 398, so we should select N-fold evaluation.

# 1. examine the relationship between x and y.

# Because cylinders, year, org are dummy variables == binary variables,

# It is difficult to interpret the correlations, so we compare the correlations except these.

mpg = mydata$mpg

displacement = mydata$displacement

hp = mydata$horsepower

wt = mydata$weight

acc = mydata$acceleration

cylinders = mydata[c(2:6)]

year = mydata[c(11:22)]

org = mydata[c(24:26)]

# Various y tansformations

mpg2 = mpg\*mpg

mpg3 = mpg\*mpg

logmpg = log(mpg)

invmpg = 1/mpg

sqrtmpg = sqrt(mpg)

# examine the correlations between mpg and x variables

cor(mpg,mydata[c(1,7:10)])

# I thought that I could increase the correlation more, so I proceeded to transform Y.

cor(mpg2,mydata[c(1,7:10)])

cor(mpg3,mydata[c(1,7:10)])

cor(logmpg,mydata[c(1,7:10)])

cor(invmpg,mydata[c(1,7:10)])

cor(sqrtmpg,mydata[c(1,7:10)])

# There was a correlation rise in logmpg, invmpg, and sqrtmpg when compared from above.

# I chose logmpg among them because they all have similar values.

plot(displacement, logmpg)

plot(hp, logmpg)

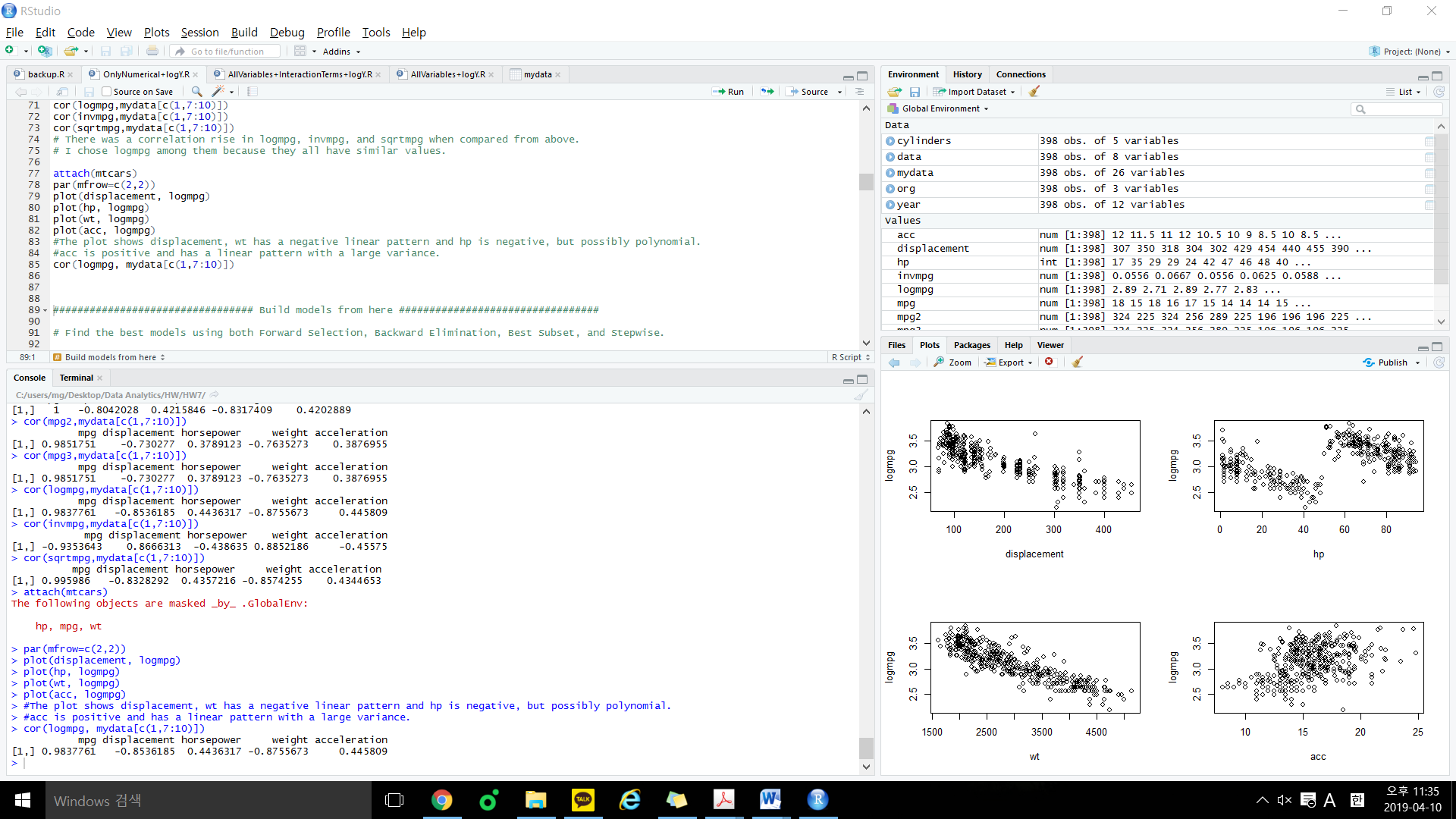
plot(wt, logmpg)

plot(acc, logmpg)

#The plot shows displacement, wt has a negative linear pattern and hp is negative, but possibly polynomial.

#acc is positive and has a linear pattern with a large variance.

cor(logmpg, mydata[c(1,7:10)])



################################# Build models from here #################################

# Find the best models using both Forward Selection, Backward Elimination, Best Subset, and Stepwise.

# with 95% confidence or significance level.

################################# 1. Feature Selections #################################

# Backward Elimination by manually drop x based on p-value

m1 = glm(logmpg~displacement+hp+wt+acc)

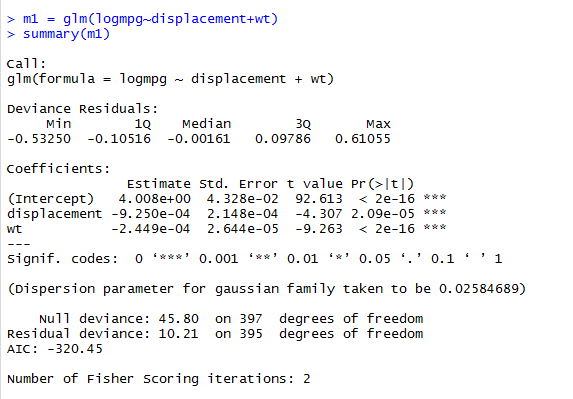
summary(m1) # 1. Remove hp(p-value = 0.5291 > 0.05)

m1 = glm(logmpg~displacement+wt+acc)

summary(m1) # 2. Remove acc(p-value = 0.0552 > 0.05)

m1 = glm(logmpg~displacement+wt)

summary(m1)

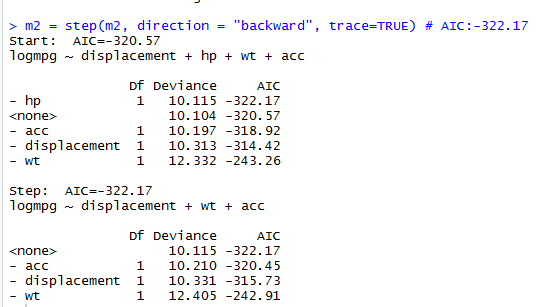


# Backward Elimination by step() based on AIC

m2 = glm(logmpg~displacement+hp+wt+acc)

summary(m2) # AIC:-320.57

m2 = step(m2, direction = "backward", trace=TRUE) # AIC:-322.17



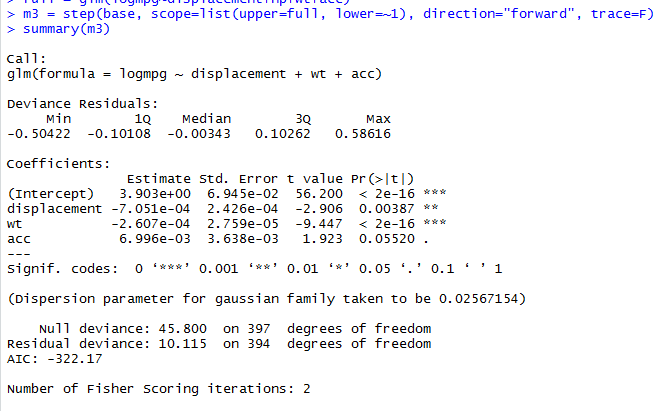
# Forward Selection by step()

base = glm(logmpg~displacement)

full = glm(logmpg~displacement+hp+wt+acc)

m3 = step(base, scope=list(upper=full, lower=~1), direction="forward", trace=F)

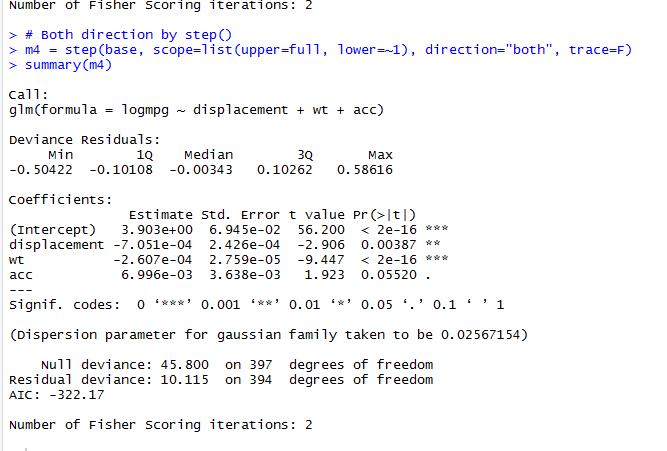
summary(m3)



# Both direction by step()

m4 = step(base, scope=list(upper=full, lower=~1), direction="both", trace=F)

summary(m4)



# Best Subset selection by (Cp,r2,adjr2 with least number of x variables)

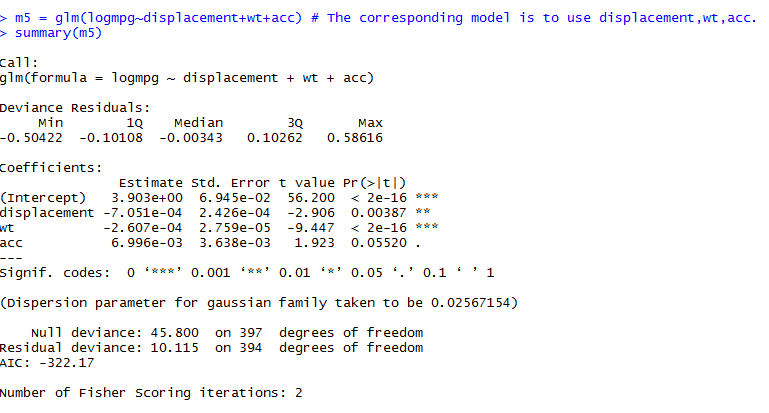
# Install.packages('leaps')

library(leaps)

leaps(y=logmpg,x=mydata[,cbind(7:10)],names=names(mydata[,cbind(7:10)]),method="adjr2") # Criteria:adjr2, index[11] is the biggest.

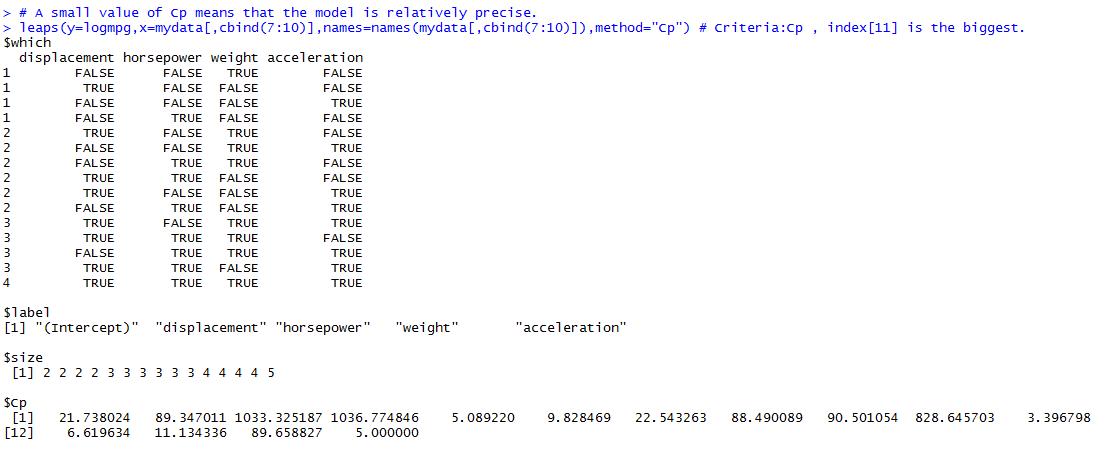
m5 = glm(logmpg~displacement+wt+acc) # The corresponding model is to use displacement,wt,acc.

summary(m5)



# A small value of Cp means that the model is relatively precise.

leaps(y=logmpg,x=mydata[,cbind(7:10)],names=names(mydata[,cbind(7:10)]),method="Cp") # Criteria:Cp , index[11] is the biggest.



# m1 : logmpg~wt+acc

# m2 : logmpg~displacement+wt+acc

# m3 : logmpg~displacement+wt+acc

# m4 : logmpg~displacement+wt+acc

# m5 : logmpg~displacement+wt+acc

# Where m2 to m5 all have the same model, so I will test only one m2.

################################# 2. Examine F-test (To validate models is qualified or not) #################################

# We do not have adjR2 and F-test results.

# It is because we use glm to build the linear regression models.

# So as long as x variable has small p-value in t-test, it is satisfied.

################################# 3. VIF (To solve multi-collinearity problem) #################################

library(car)

vif(m1)

# Since the VIF of displacement and wt in m1 exceeds 4, we make m6 and m7, respectively.

m6 = glm(logmpg~wt)

m7 = glm(logmpg~displacement)

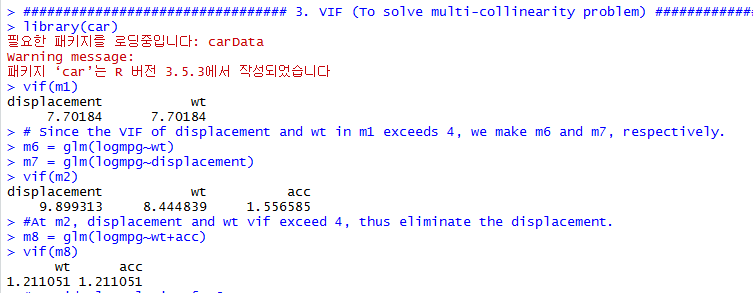
vif(m2)

#At m2, displacement and wt vif exceed 4, thus eliminate the displacement.

m8 = glm(logmpg~wt+acc)

vif(m8)

# Now we got models m6,m7,m8.



################################# 4. Residual Analysis (To validate models is qualified or not) #################################

# Residual analysis of m6

res=rstandard(m6)

attach(mtcars)

par(mfrow=c(2,2))

#1) Validate the constant variance

plot(fitted(m6),res,main="Predicted vs Residuals plot")

abline(a=0, b=0, col='red')

#2) Validate the linearity relationship

plot(wt, res, main="wt vs residuals plot")

abline(a=0, b=0, col='red')

plot(acc, res, main="acc vs residuals plot")

abline(a=0, b=0, col='red')

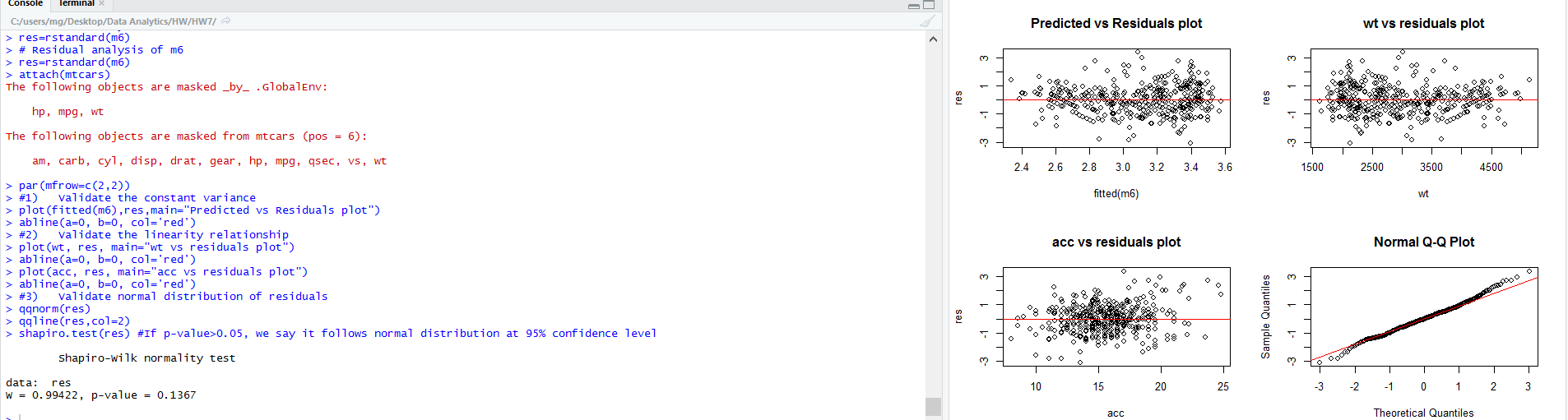
#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value = 0.1367 > 0.05 : Normal distribution



# Residual analysis of m7

res=rstandard(m7)

attach(mtcars)

par(mfrow=c(2,2))

#1) Validate the constant variance

plot(fitted(m7),res,main="Predicted vs Residuals plot")

abline(a=0, b=0, col='red')

#2) Validate the linearity relationship

plot(wt, res, main="wt vs residuals plot")

abline(a=0, b=0, col='red')

plot(acc, res, main="acc vs residuals plot")

abline(a=0, b=0, col='red')

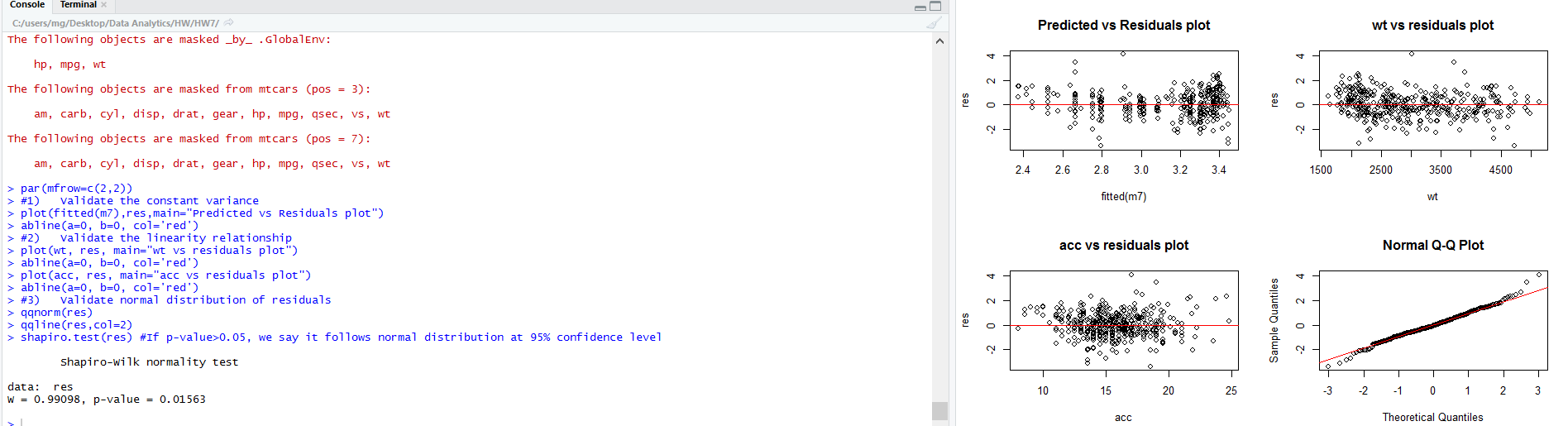
#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value = 0.01563 > 0.05 : No Normal distribution



# Residual analysis of m8

res=rstandard(m8)

attach(mtcars)

par(mfrow=c(2,2))

#1) Validate the constant variance

plot(fitted(m8),res,main="Predicted vs Residuals plot")

abline(a=0, b=0, col='red')

#2) Validate the linearity relationship

plot(wt, res, main="wt vs residuals plot")

abline(a=0, b=0, col='red')

plot(acc, res, main="acc vs residuals plot")

abline(a=0, b=0, col='red')

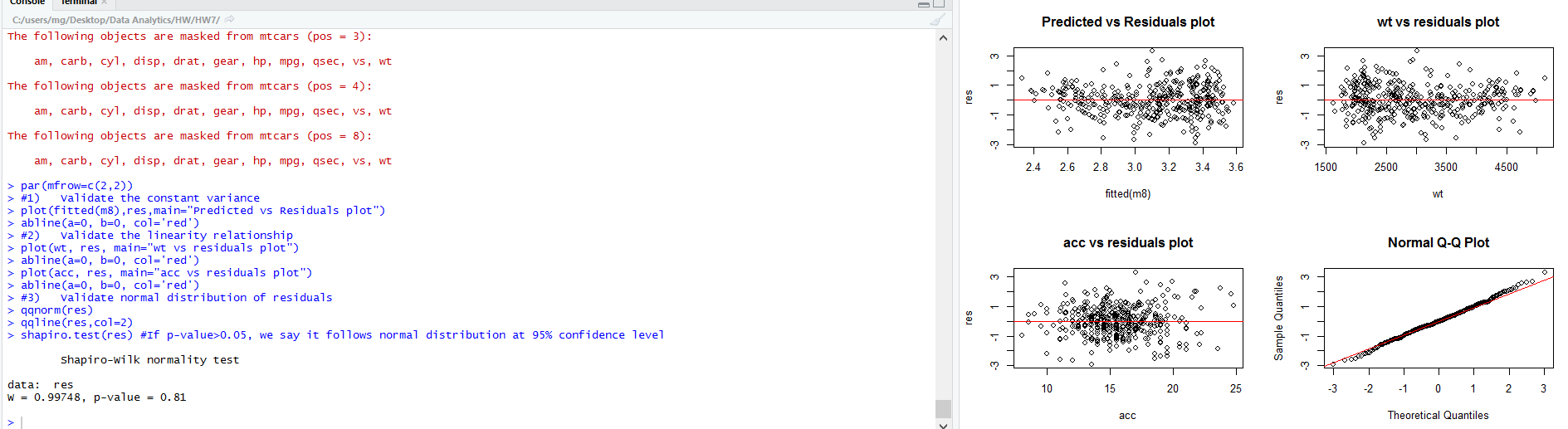
#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value = 0.81 > 0.05 : Normal distribution



# Therefore, only m6 and m8 passed the residual analysis.

#4) Identify potential outliers and (influential points)

# Identify potential outliers and (influential points) about m6.

#The criteria is 4/n = 4/398 = 0.01005025. If cook.d is larger than 4/n, it is influential points.

library(stats)

#influence.measures(m6)

cooks.distance(m6)

sort(cooks.distance(m6), TRUE)[21] #1 to 21 are influential points, but use only until the 20th.

sort(cooks.distance(m6), TRUE)

# Create new data by removing influential points

m6\_newdata = mydata[-c(112,29,45,323,157,365,27,125,327,388,326,395,299,330,113,26,245,328,72,167),]

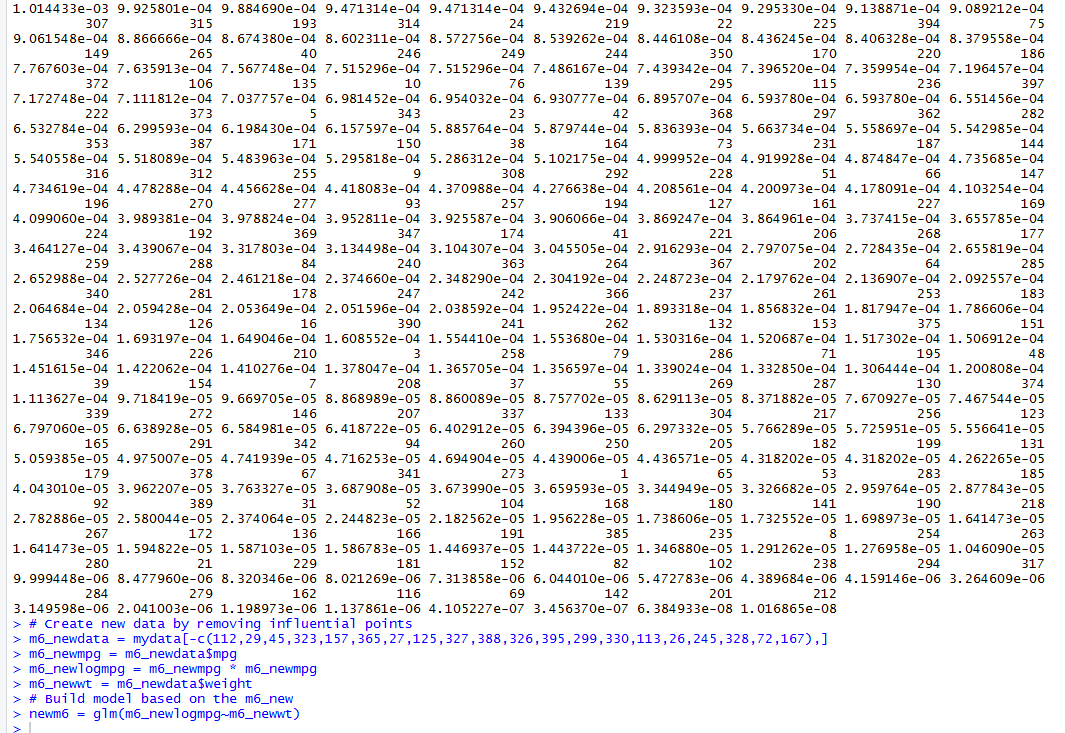
m6\_newmpg = m6\_newdata$mpg

m6\_newlogmpg = m6\_newmpg \* m6\_newmpg

m6\_newwt = m6\_newdata$weight

# Build model based on the m6\_new

newm6 = glm(m6\_newlogmpg~m6\_newwt)



# Identify potential outliers and (influential points) about m8.

# The criteria is 4/n = 4/398 = 0.01005025. If cook.d is larger than 4/n, It is influential points.

cooks.distance(m8)

sort(cooks.distance(m8), TRUE)[20] #1 to 20 are influentil points.

sort(cooks.distance(m8), TRUE)

# Create new data by removing influential points

m8\_newdata = mydata[-c(327,29,395,60,329,112,365,328,14,334,155,125,326,156,330,167,157,361,300,298),]

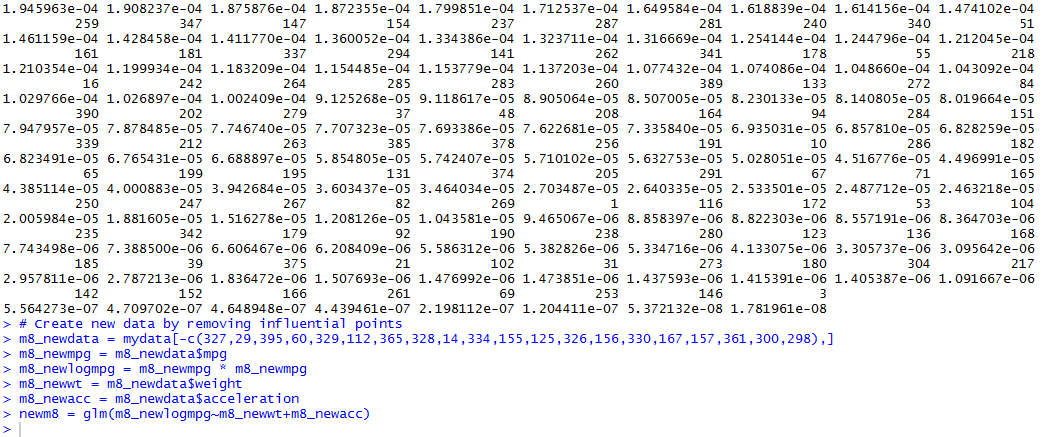
m8\_newmpg = m8\_newdata$mpg

m8\_newlogmpg = m8\_newmpg \* m8\_newmpg

m8\_newwt = m8\_newdata$weight

m8\_newacc = m8\_newdata$acceleration

newm8 = glm(m8\_newlogmpg~m8\_newwt+m8\_newacc)



################################# 5. Evaluate the model #################################

#5) Evaluate the model.

#install.packages('boot')

library('boot')

# N-fold cross validation

mse6 = cv.glm(mydata, m6, K=5)$delta

mse8 = cv.glm(mydata, m8, K=5)$delta

mse\_newm6 = cv.glm(m6\_newdata, newm6, K=5)$delta

mse\_newm8 = cv.glm(m8\_newdata, newm8, K=5)$delta

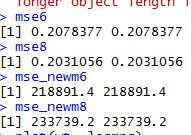
mse6

mse8

mse\_newm6

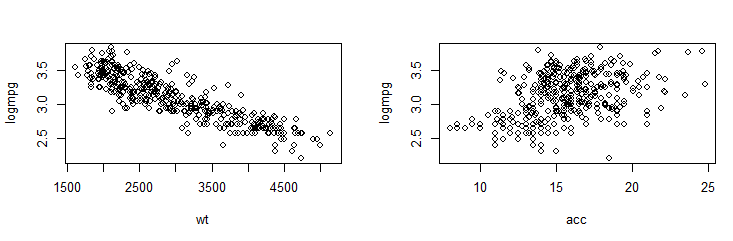
mse\_newm8

# When compared, m8 is the most suitable model because the error is the smallest. mse8 = 0.203276



plot(wt, logmpg)

plot(acc, logmpg)



# It is not necessary to apply the high-order term because the curve is not visible in the plot of m8.

**FILE:AllVariables+logY**

getwd()

setwd('C:/users/mg/Desktop/Data Analytics/HW/HW7/')

mydata=read.table('HW7\_AutoMPG.csv',header=T,',')

# In this case, I used all variables.

######################################################################################################################

# Purpose of this section : Change discrete values as categorical values. Solve missing values problem.

# Multi-valued discrete values can be a categorical values.

#all data is numerical

summary(mydata) #it tells if missing values exist or not

# horsepower missing data preprocessing

mydata$horsepower = ifelse(is.na(mydata$horsepower), ave(data$horsepower, FUN = function(x) mean(x, na.rm=T)), mydata$horsepower)

summary(mydata)

######################################################################################################################

# Purpose of this section : Variable declarations and examine relationships to create models.

# Make Multiple Linear Regression Model.

# y variable = mpg, x variables = other 7 variables.

nrow(mydata) # data size = 398, so we should select N-fold evaluation.

# 1. examine the relationship between x and y.

mpg = mydata$mpg

dis = mydata$displacement

hp = mydata$horsepower

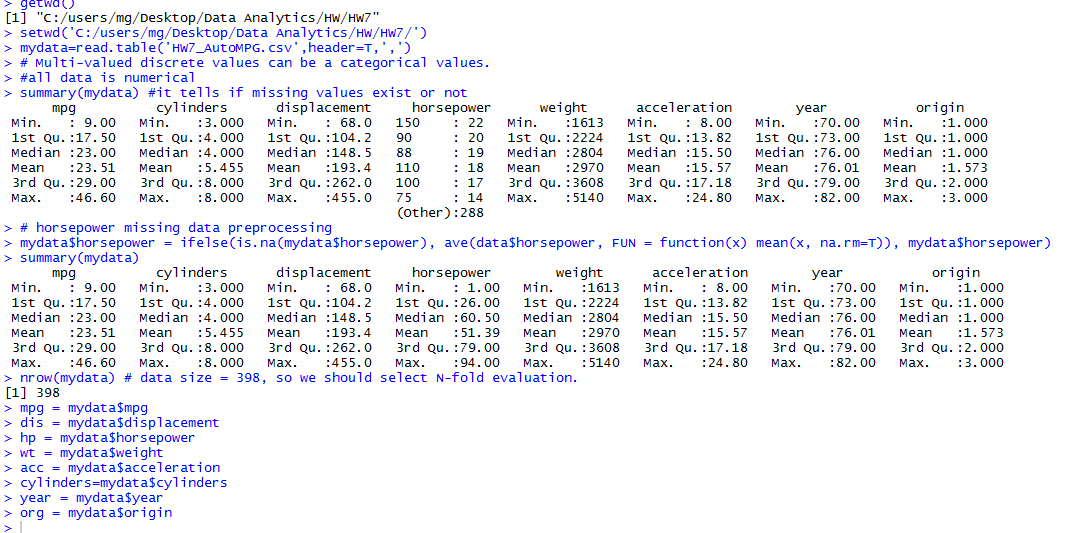
wt = mydata$weight

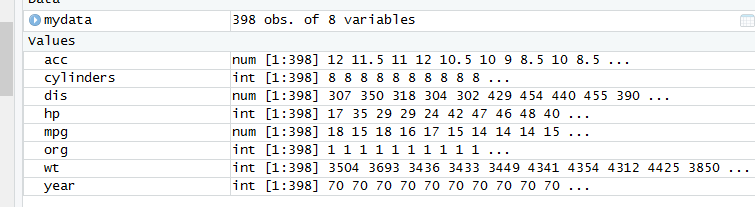
acc = mydata$acceleration

cylinders=mydata$cylinders

year = mydata$year

org = mydata$origin



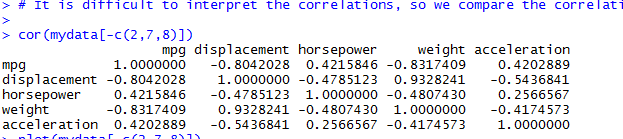


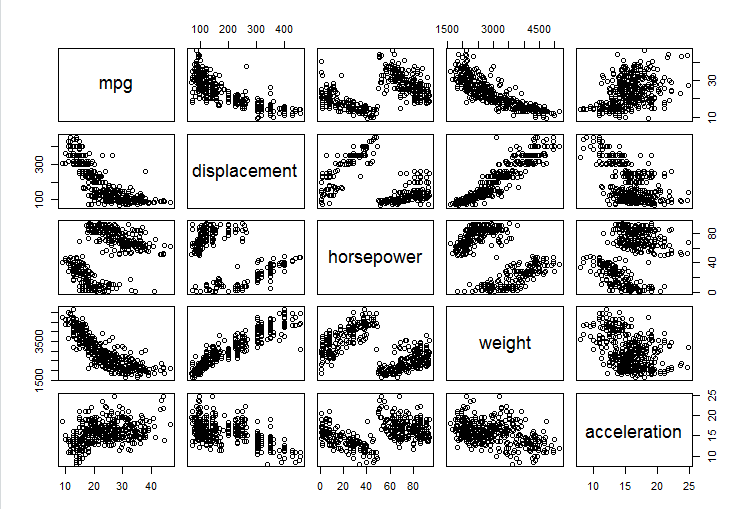
# Because cylinders, year, org are dummy variables == binary variables,

# It is difficult to interpret the correlations, so we compare the correlations except these.

cor(mydata[-c(2,7,8)])

plot(mydata[-c(2,7,8)])





# I thought that I could increase the correlation more, so I proceeded to transform Y.

logmpg = log(mpg)

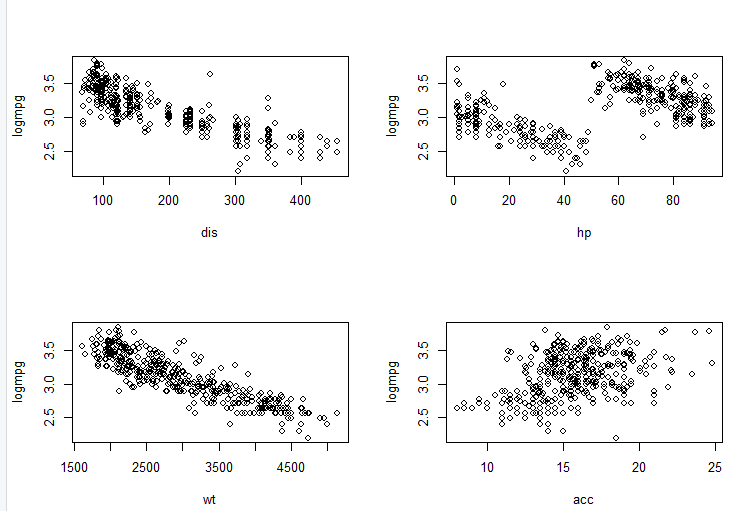
cor(logmpg, mydata[-c(2,7,8)])

plot(dis, logmpg)

plot(hp, logmpg)

plot(wt, logmpg)

plot(acc, logmpg)



#The plot shows displacement, wt has a negative linear pattern and hp is negative, but possibly polynomial.

#acc is positive and has a linear pattern with a large variance.

################################# Build models from here #################################

# Find the best models using both Forward Selection, Backward Elimination, Best Subset, and Stepwise.

# with 95% confidence or significance level.

################################# 1. Feature Selections #################################

# Backward Elimination by manually drop x based on p-value

hp2 = hp\*hp

hp3 = hp\*hp2

loghp = log(hp)

sqrthp = sqrt(hp)

m0 = glm(logmpg ~ dis+ hp + wt + acc + as.factor(cylinders)+ as.factor(year) + as.factor(org) + hp2 + hp3 + loghp + sqrthp, data=mydata)

summary(m0)

m0 = glm(logmpg ~ dis+ hp + wt + as.factor(cylinders)+ as.factor(year) + as.factor(org) + hp2 + hp3 + loghp + sqrthp, data=mydata)

summary(m0)

m0 = glm(logmpg ~ hp + wt + as.factor(cylinders)+ as.factor(year) + as.factor(org) + hp2 + hp3 + loghp + sqrthp, data=mydata)

summary(m0)

m0 = glm(logmpg ~ hp + wt + as.factor(cylinders)+ as.factor(year) + as.factor(org) + hp2 + loghp + sqrthp, data=mydata)

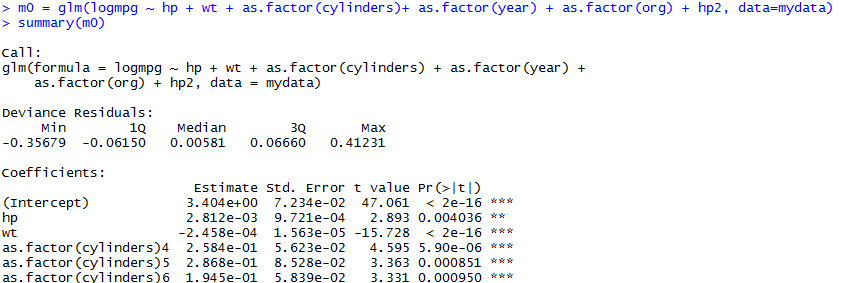
summary(m0)

m0 = glm(logmpg ~ hp + wt + as.factor(cylinders)+ as.factor(year) + as.factor(org) + hp2 + sqrthp, data=mydata)

summary(m0)

m0 = glm(logmpg ~ hp + wt + as.factor(cylinders)+ as.factor(year) + as.factor(org) + hp2, data=mydata)

summary(m0)



# I found that the relationship between logmpg and hp is a quadratic from this.

# So now on i will use hp2 as new variable.

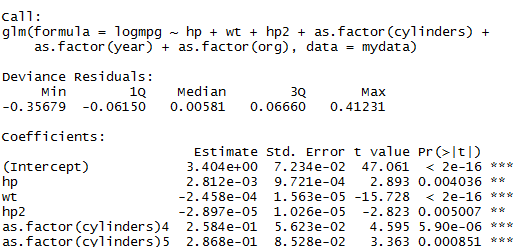
# Backward Elimination by step() based on AIC

m1 = glm(logmpg ~ dis + hp + wt + acc + hp2 + as.factor(cylinders)+ as.factor(year) + as.factor(org), data=mydata)

summary(m1) # AIC: -629.18

m1 = step(m1, direction = "backward", trace=TRUE) # AIC:-632.27

summary(m1)



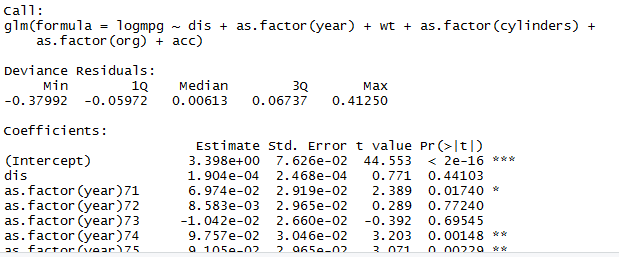
# Forward Selection by step()

base = glm(logmpg~dis)

full = glm(logmpg~dis+ hp + wt + acc + hp2 + as.factor(cylinders)+ as.factor(year) + as.factor(org), data=mydata)

m2 = step(base, scope=list(upper=full, lower=~1), direction="forward", trace=F)

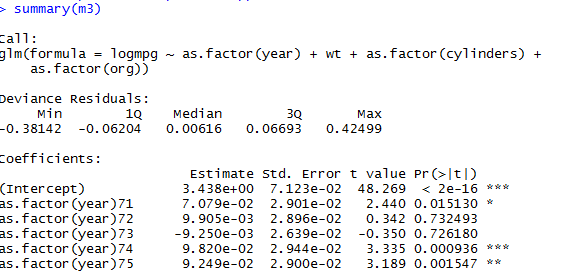
summary(m2)



# Both direction by step()

m3 = step(base, scope=list(upper=full, lower=~1), direction="both", trace=F)

summary(m3)



# Best Subset selection by (Cp,r2,adjr2 with least number of x variables) (W6\_01\_37p)

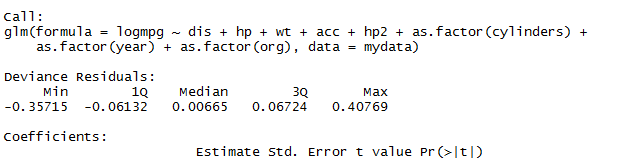
#install.packages('leaps')

library(leaps)

leaps(y=logmpg,x=mydata[,cbind(2:8)],names=names(mydata[,cbind(2:8)]),method="adjr2") #adjr2 index[55] is the biggest.

m4 = full

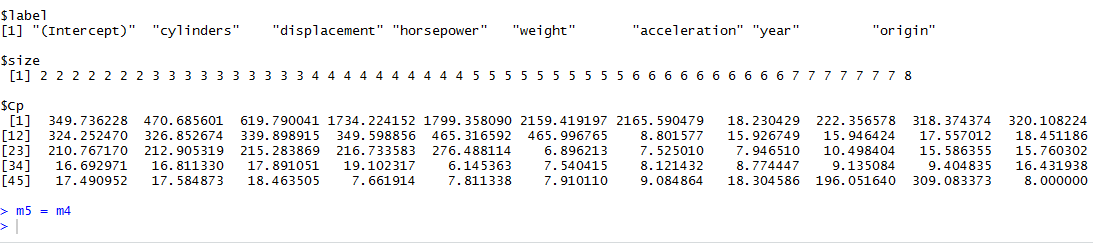
summary(m4)



#A small value of Cp means that the model is relatively precise.

leaps(y=logmpg,x=mydata[,cbind(2:8)],names=names(mydata[,cbind(2:8)]),method="Cp") # index[55] is the biggest.

m5 = m4



# m1 = glm(formula = logmpg ~ hp + wt + hp2 + as.factor(cylinders) + as.factor(year) + as.factor(org), data = mydata)

# m2 = glm(formula = logmpg ~ dis + as.factor(year) + wt + as.factor(cylinders) + as.factor(org) + acc)

# m3 = glm(formula = logmpg ~ as.factor(year) + wt + as.factor(cylinders) + as.factor(org))

# m4 = glm(logmpg ~ dis + hp + wt + acc + as.factor(cylinders) + as.factor(year) + as.factor(org), data = mydata)

# m5 = glm(logmpg ~ dis + hp + wt + acc + as.factor(cylinders) + as.factor(year) + as.factor(org), data = mydata)

# Therefore, we obtain models m1, m2, m3, m4, and m5 (m4 and m5 are the same so i use m4 only.).

################################# 2. Examine F-test (To validate models is qualified or not) #################################

# We do not have adjR2 and F-test results.

# It is because we use glm to build the linear regression models.

################################# 3. VIF (To solve multi-collinearity problem) #################################

library(car)

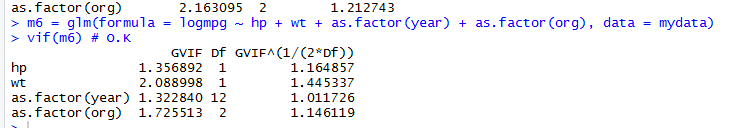
vif(m1) # Remove hp2 because gvif > 4

m6 = glm(formula = logmpg ~ hp + wt + as.factor(cylinders) + as.factor(year) + as.factor(org), data = mydata)

vif(m6) # Remove cylinders because of same reason

m6 = glm(formula = logmpg ~ hp + wt + as.factor(year) + as.factor(org), data = mydata)

vif(m6) # O.K



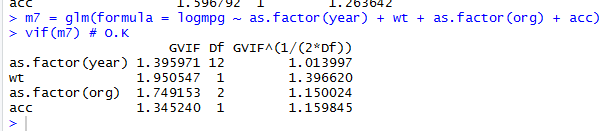
vif(m2) # Remove dis

m7 = glm(formula = logmpg ~ as.factor(year) + wt + as.factor(cylinders) + as.factor(org) + acc)

vif(m7) # Remove cylinders

m7 = glm(formula = logmpg ~ as.factor(year) + wt + as.factor(org) + acc)

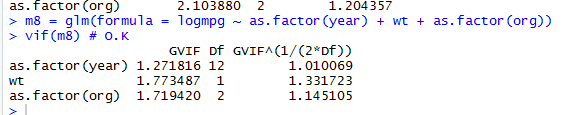
vif(m7) # O.K



vif(m3) # Remove cylinders

m8 = glm(formula = logmpg ~ as.factor(year) + wt + as.factor(org))

vif(m8) # O.K



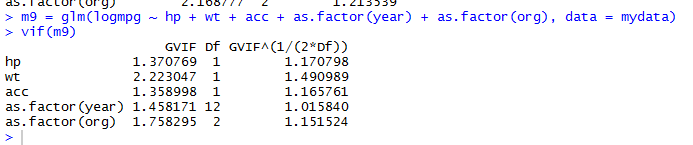
vif(m4) # Remove dis

m9 = glm(logmpg ~ hp + wt + acc + as.factor(cylinders) + as.factor(year) + as.factor(org), data = mydata)

vif(m9) # Remove cylinders

m9 = glm(logmpg ~ hp + wt + acc + as.factor(year) + as.factor(org), data = mydata)

vif(m9)



# m6 = glm(formula = logmpg ~ hp + wt + as.factor(year) + as.factor(org), data = mydata)

# m7 = glm(formula = logmpg ~ as.factor(year) + wt + as.factor(org) + acc)

# m8 = glm(formula = logmpg ~ as.factor(year) + wt + as.factor(org))

# m9 = glm(logmpg ~ hp + wt + acc + as.factor(year) + as.factor(org), data = mydata)

################################# 4. Residual Analysis (To validate models is qualified or not) #################################

#m6 residual analysis

res=rstandard(m6)

#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value > 0.05, Not a Normal distribution

#m7 residual analysis

res=rstandard(m7)

#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value > 0.05, Not a Normal distribution

#m8 residual analysis

res=rstandard(m8)

#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value > 0.05, Not aNormal distribution

#m9 residual analysis

res=rstandard(m9)

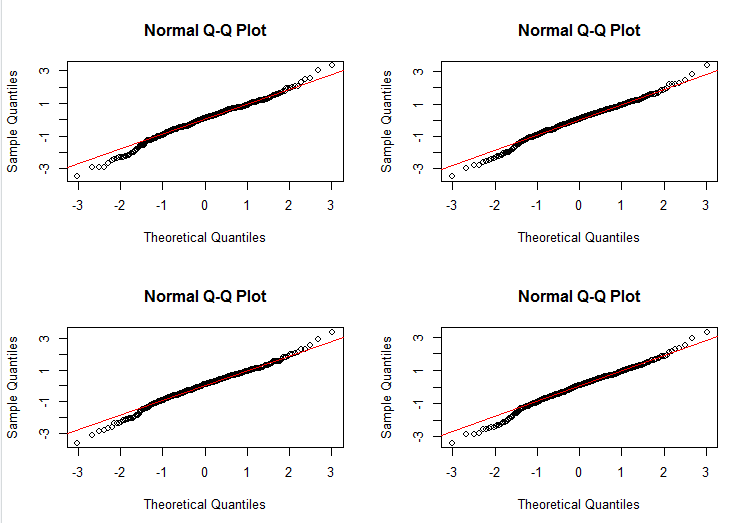
#3) Validate normal distribution of residuals

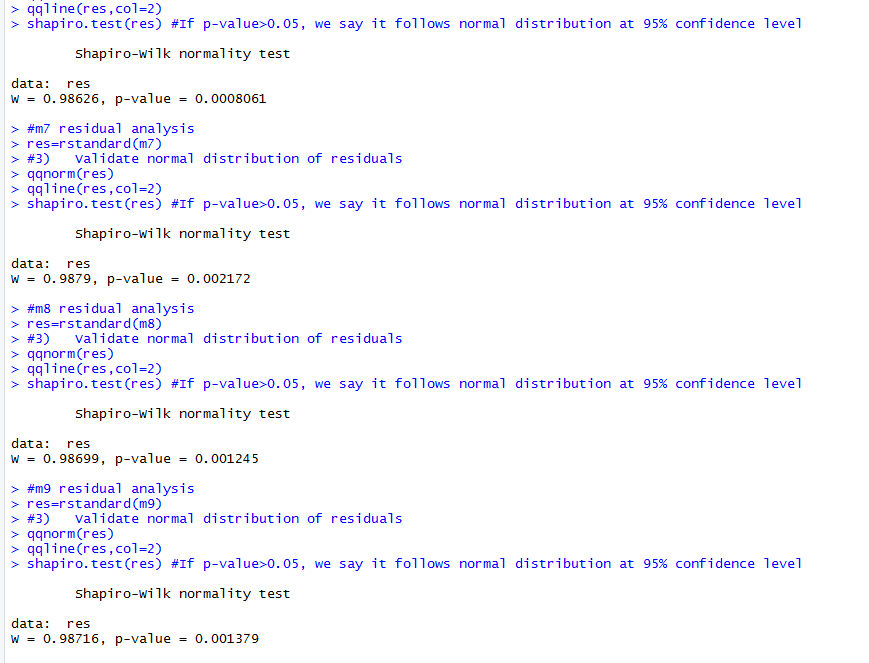
qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level

#p-value > 0.05 Not a Normal distribution





# Therefore, there is no suitable model.

**FILE:AllVariable+InteractionTerms+logY**

getwd()

setwd('C:/users/mg/Desktop/Data Analytics/HW/HW7/')

mydata=read.table('HW7\_AutoMPG.csv',header=T,',')

# In this case, I used all variables and interaction terms

######################################################################################################################

# Purpose of this section : Change discrete values as categorical values. Solve missing values problem.

# Multi-valued discrete values can be a categorical values.

#all data is numerical

summary(mydata) #it tells if missing values exist or not

# horsepower missing data preprocessing

mydata$horsepower = ifelse(is.na(mydata$horsepower), ave(data$horsepower, FUN = function(x) mean(x, na.rm=T)), mydata$horsepower)

summary(mydata)

######################################################################################################################

# Purpose of this section : Variable declarations and examine relationships to create models.

# Make Multiple Linear Regression Model.

# y variable = mpg, x variables = other 7 variables.

nrow(mydata) # data size = 398, so we should select N-fold evaluation.

# 1. examine the relationship between x and y.

mpg = mydata$mpg

dis = mydata$displacement

hp = mydata$horsepower

wt = mydata$weight

acc = mydata$acceleration

cylinders=mydata$cylinders

year = mydata$year

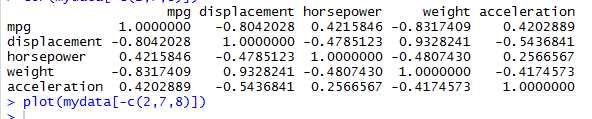
org = mydata$origin

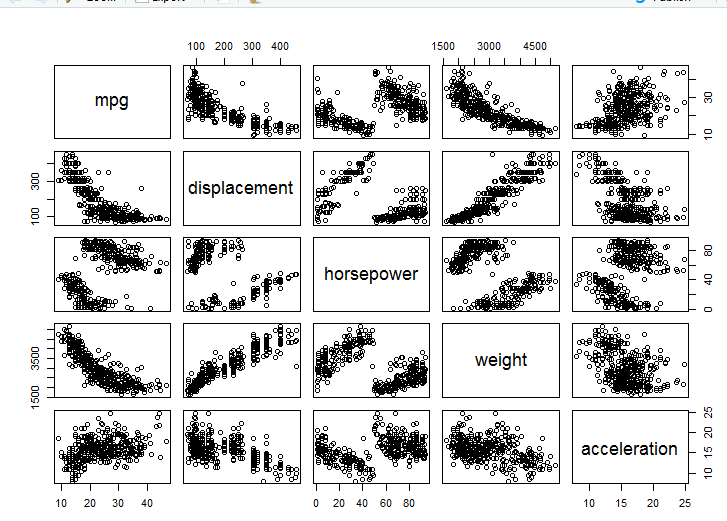
# Because cylinders, year, org are dummy variables == binary variables,

# It is difficult to interpret the correlations, so we compare the correlations except these.

cor(mydata[-c(2,7,8)])

plot(mydata[-c(2,7,8)])





# I thought that I could increase the correlation more, so I proceeded to transform Y.

logmpg = log(mpg)

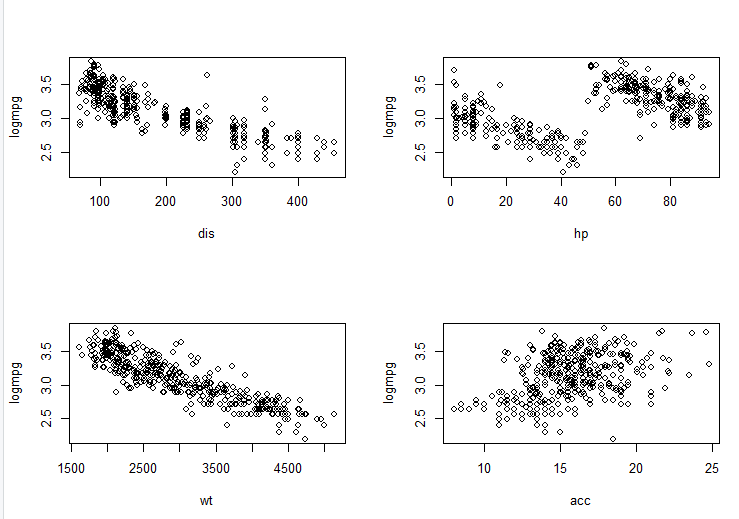
cor(logmpg, mydata[-c(2,7,8)])

plot(dis, logmpg)

plot(hp, logmpg)

plot(wt, logmpg)

plot(acc, logmpg)



#The plot shows displacement, wt has a negative linear pattern and hp is negative, but possibly polynomial.

#acc is positive and has a linear pattern with a large variance.

################################# Build models from here #################################

# Find the best models using both Forward Selection, Backward Elimination, Best Subset, and Stepwise.

# with 95% confidence or significance level.

################################# 1. Feature Selections #################################

hp2 = hp\*hp

# Interaction terms variables.

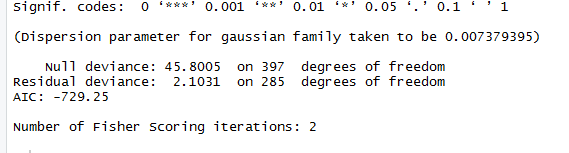
# Backward Elimination by step() based on AIC

model = glm(logmpg ~ (dis+hp+wt+acc+hp2+as.factor(cylinders)+ as.factor(year) + as.factor(org))^2, data=mydata)

summary(model) # AIC:-692.07

model = step(model, direction = "backward", trace=TRUE)

summary(model) # AIC:-729.25



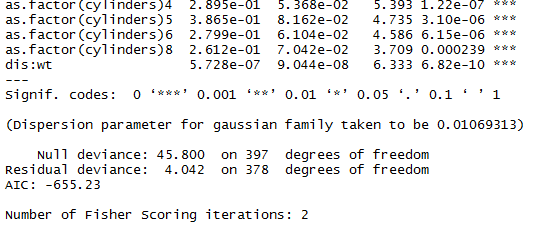
# Both direction by step()

full = glm(logmpg ~ (dis+hp+wt+acc+hp2+as.factor(cylinders)+ as.factor(year) + as.factor(org))^2, data=mydata)

base = glm(logmpg ~ dis)

model2 = step(base, scope=list(upper=model, lower=~1), direction="both", trace=F)

summary(model2) # Since -692 > -655, discard this model.



################################# 2. Examine F-test (To validate models is qualified or not) #################################

# We do not have adjR2 and F-test results.

# It is because we use glm to build the linear regression models.

################################# 3. VIF (To solve multi-collinearity problem) #################################

vif(model) # Error in vif.default(model) : there are aliased coefficients in the model



################################# 4. Residual Analysis (To validate models is qualified or not) #################################

# Residual analysis of m6

res=rstandard(model)

attach(mtcars)

par(mfrow=c(3,3))

#1) Validate the constant variance

plot(fitted(model),res,main="Predicted vs Residuals plot")

abline(a=0, b=0, col='red')

#2) Validate the linearity relationship

plot(dis, res, main="dis vs residuals plot")

abline(a=0, b=0, col='red')

plot(hp, res, main="hp vs residuals plot")

abline(a=0, b=0, col='red')

plot(wt, res, main="wt vs residuals plot")

abline(a=0, b=0, col='red')

plot(acc, res, main="acc vs residuals plot")

abline(a=0, b=0, col='red')

plot(hp2, res, main="hp2 vs residuals plot")

abline(a=0, b=0, col='red')

plot(as.factor(cylinders), res, main="as.factor(cylinders) vs residuals plot")

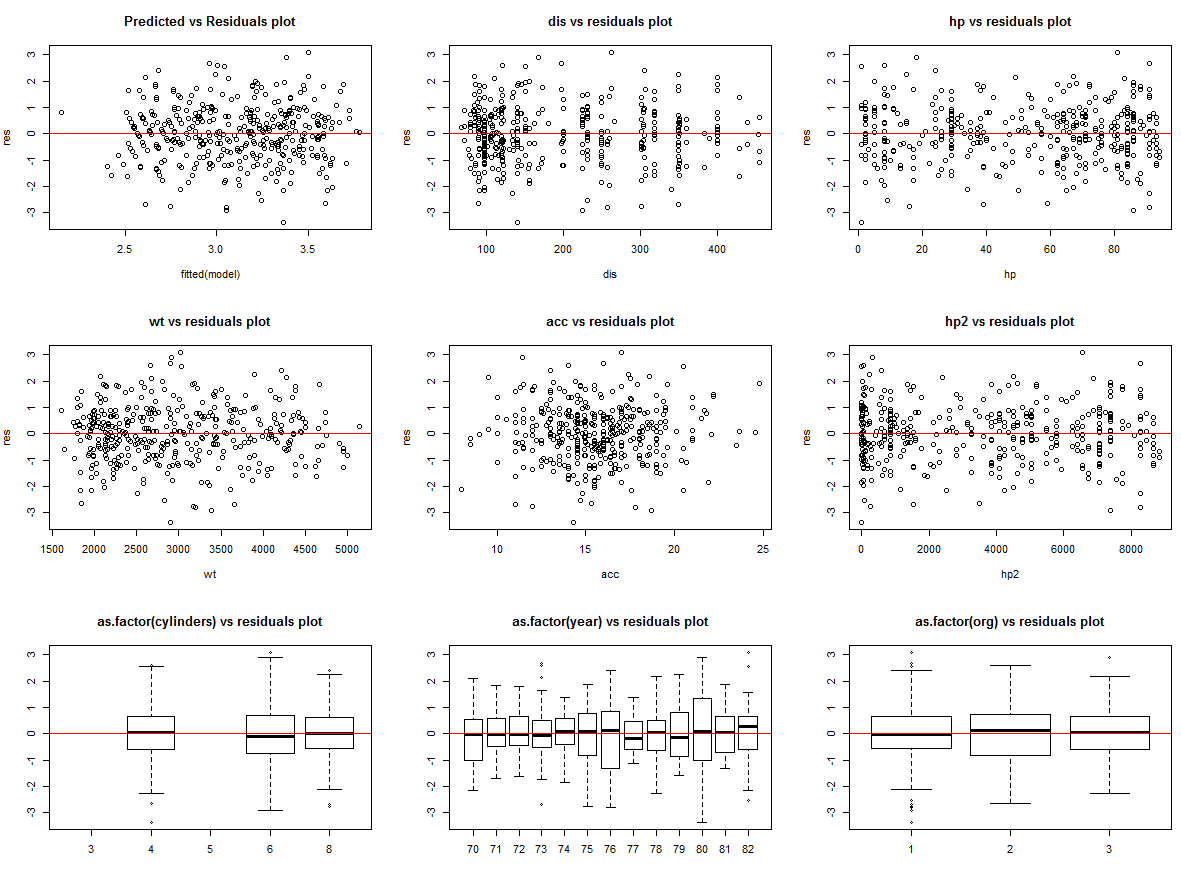
abline(a=0, b=0, col='red')

plot(as.factor(year), res, main="as.factor(year) vs residuals plot")

abline(a=0, b=0, col='red')

plot(as.factor(org), res, main="as.factor(org) vs residuals plot")

abline(a=0, b=0, col='red')

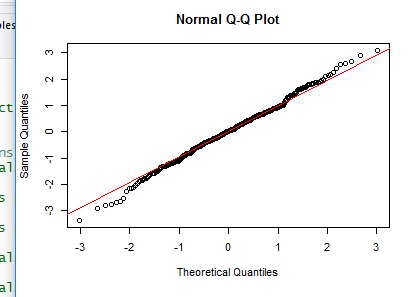


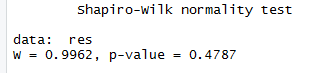
#3) Validate normal distribution of residuals

qqnorm(res)

qqline(res,col=2)

shapiro.test(res) #If p-value>0.05, we say it follows normal distribution at 95% confidence level





#p-value = 0.4787 > 0.05 : Normal distribution

#4) Identify potential outliers and (influential points)

# Identify potential outliers and (influential points) about model.

#The criteria is 4/n = 4/398 = 0.01005025. If cook.d is larger than 4/n, it is influential points.

library(stats)

#influence.measures(model)

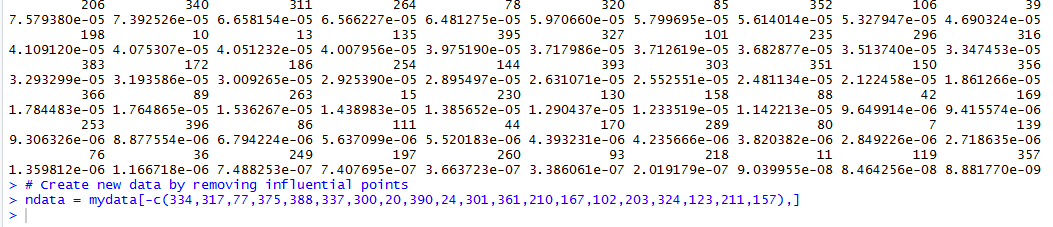
cooks.distance(model)

sort(cooks.distance(model), TRUE)[46] #1 to 46 are influential points, but use only until the 20th.

sort(cooks.distance(model), TRUE)

# Create new data by removing influential points

ndata = mydata[-c(334,317,77,375,388,337,300,20,390,24,301,361,210,167,102,203,324,123,211,157),]



nmpg = ndata$mpg

nlogmpg = log(nmpg)

ndis = ndata$displacement

nhp = ndata$horsepower

nwt = ndata$weight

nacc = ndata$acceleration

nhp2 = nhp\*nhp

ncy = ndata$cylinders

nyr = ndata$year

norg = ndata$origin

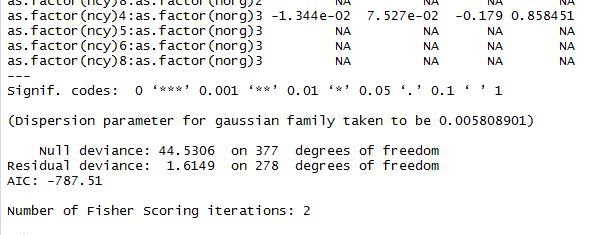
# Build model based on the model\_new

model\_new = glm(nlogmpg ~ (ndis+nhp+nwt+nacc+nhp2+as.factor(ncy)+ as.factor(nyr) + as.factor(norg))^2, data=ndata)

summary(model\_new) # AIC:-734.34

model\_new = step(model\_new, direction = "backward", trace=TRUE)

summary(model\_new) # AIC:-787.51



################################# 5. Evaluate the model #################################

#5) Evaluate the model.

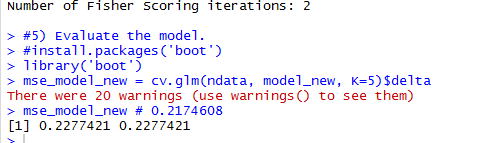
#install.packages('boot')

library('boot')

# N-fold cross validation

mse\_model\_new = cv.glm(ndata, model\_new, K=5)$delta

mse\_model\_new # 0.2277421



Conclusion:

|  |  |  |
| --- | --- | --- |
|  | AIC | MSE |
| AllVariables+InteractionTerms+logY | -787.51 | 0.2277421 |
| OnlyNumerical+logY | -322.17 | 0.203276 |

If you look at the table, you can see that the mse8 model is better because MSE is more lower.