#setwd("~/Masters/")

library(tidyverse)

library(ggplot2)

library(car)

library(GGally)

library(effects)

#setwd("~/Masters/")

babies.data <- read.table("babies23.data", header = TRUE)

#since we are working in our directory, I change the directory that I think that

#people use this project can run it.

#observations from data set:

# pluralty is always 5

# outcome is always 1

# there are values of 999 for gestation but readme doc does not clarify if

# these are unknown - CLEANED ANYWAY

# all subjects are male

# for race, I'm unsure why white is assigned six values (0-5) - one unknown

# two unknown ages (mother) - CLEANED

# one unknown education (mother) - CLEANED

# many unknown heights (mother) - CLEANED

# many unkown weights (mother) - CLEANED

# five unknown fathers' races as well as values of 10? - 99s CLEANED -

# many unknown fathers' ages - CLEANED

# many unknown fathers' educations - CLEANED

# many unknown fathers' heights - CLEANED

# many unknown fathers' weights - CLEANED

# no explanation of 0 in marital status - assume unknown?

# many unknown incomes - CLEANED

# ten unknown smokers - CLEANED

# nine unknown quitting times, one not asked - CLEANED

# ten unknown number of cigarettes smoked - CLEANED

##### cleaning the data as per unknown values above #####

clean.data <- babies.data

clean.data$gestation[clean.data$gestation == "999"] <- NA

clean.data$age[clean.data$age == "99"] <- NA

clean.data$ed[clean.data$ed == "9"] <- NA

clean.data$ht[clean.data$ht == "99"] <- NA

clean.data$wt[clean.data$wt == "99"] <- NA

clean.data$drace[clean.data$drace == "99"] <- NA

clean.data$dage[clean.data$dage == "99"] <- NA

clean.data$ded[clean.data$ded == "9"] <- NA

clean.data$dht[clean.data$dht == "99"] <- NA

clean.data$dwt[clean.data$dwt == "999"] <- NA

clean.data$inc[clean.data$inc == "98"] <- NA

clean.data$smoke[clean.data$smoke == "9"] <- NA

clean.data$time[clean.data$time == "99"] <- NA

clean.data$time[clean.data$time == "98"] <- NA

clean.data$number[clean.data$number == "98"] <- NA

clean.data$wt.1[clean.data$wt.1 == "999"] <- NA

#make some factors numeric

clean.data <- clean.data %>% mutate\_each(funs(as.numeric), 5)

clean.data <- clean.data %>% mutate\_each(funs(as.numeric), 7)

clean.data <- clean.data %>% mutate\_each(funs(as.numeric), 10)

clean.data <- clean.data %>% mutate\_each(funs(as.numeric), 12:13)

clean.data <- clean.data %>% mutate\_each(funs(as.numeric), 15)

clean.data <- clean.data %>% mutate\_each(funs(as.numeric), 17:18)

####### Exploration of the birthweight data #######

#normally distributed

hist(clean.data$wt)

summary(clean.data$wt)

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

clean.data.naomit <- na.omit(clean.data)

# select data that does not contain id and data of birth

# consider this two factor does not have effect on baby birth weight

# on the real life

clean.data.naomit <- clean.data.naomit %>% dplyr::select(-id, -date)

#factor(clean.data.naomit$id)

dataModel <- lm(wt ~., data = clean.data.naomit)

summary(dataModel)

#try to use Anova

Anova(dataModel)

#model selection use AIC

dataModel <- step(dataModel)

Anova(dataModel)

#check about normality of dataModel's residual

qqnorm(resid(dataModel))

qqline(resid(dataModel))

#the qq plot looks great but the shapiro test, p value is large than 0.05,

# so the residual of the data Model is normal

shapiro.test(resid(dataModel))

hist(resid(dataModel))

# we track down the extreme residuals

bigResid <- which(abs(resid(dataModel))>5)

clean.data.naomit[bigResid,]

#plot residuals against fitted values

dataResid <- resid(dataModel)

plot(fitted(dataModel),dataResid, ylab= "Residuals", xlab = "Fitted Values")

#it looks good

#https://onlinecourses.science.psu.edu/stat501/node/277/

# do Breusche-Pagan test with respect to fitted model

ncvTest(dataModel)

# null hypothesis: constant error variance. "If we have constant error variance

#then the variation in the residuals should be unrelated to any coveriant."

# null hypothesis is rejected since the p value is less than 0.05

#MT5761 notes page 22

# need to write durbinWatsonTest on model

durbinWatsonTest(dataModel)

#null hypothesis: error are uncorrelated, fail to reject the null hypothesis

plot(dataModel, which = 1:2)

#collinearity

numericOnly <- clean.data.naomit %>% select\_if(is.numeric)

#use with caution, picture is sooo huge and difficult to generate

# and do harm to my computer and not useful because we have sooo many variabales

#ggpairs(numericOnly)

vif(dataModel)

# all number is less than 10, do not have to delete any variable

#calculate confidence interval of the model

confint(dataModel)

#add more effect plot if you want and select variable that you

# think is interested

#plot(effect(term="gestation", mod = dataModel))

#plot(effect(term="smoke", mod = dataModel))

#plot(effect(term="number", mod = dataModel))

cols\_to\_change = c(1, 2, 3, 4,6, 8, 9, 11, 14, 16, 19, 20:23)

for(i in cols\_to\_change){

class(clean.data[, i]) = "factor"

}

cols\_to\_change

#create a first order iteraction for every variable

firstorderModel <- lm(wt ~.\*., data = numericOnly)

summary(firstorderModel)

#model selection use AIC

firstorderModel <- step(firstorderModel)

summary(firstorderModel)

Anova(firstorderModel)

qqnorm(resid(firstorderModel))

qqline(resid(firstorderModel))

shapiro.test(resid(firstorderModel))

hist(resid(firstorderModel))

firstorderResid <- resid(firstorderModel)

plot(fitted(firstorderModel),firstorderResid, ylab= "Residuals", xlab = "Fitted Values")

ncvTest(firstorderModel)

durbinWatsonTest(firstorderModel)

plot(firstorderModel, which = 1:2)

# we exam the collinearity of the firstorderModel we find that there are a lot of

# variable that its GVIF number is larger than 10, so in the following step.

# 1. we find the maximum number of GVIF, if it is larger than 10,remove it

# 2. do the vif function again to check the collinearity and get the maximum repeat the step 1

# we do the above two steps until all the variable's collinearity GVIF is less than 10

# or we do not have a collinearity problem anymore

# following just the process of removing every variable that is collinear

k<-vif(firstorderModel)

k[which.max(k)]

alteredModel <-update(firstorderModel,.~.-ht:marital )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-race )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-smoke )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dht:race)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dage)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-age:marital)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-drace)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dht:inc)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-gestation:number)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-wt.1)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-ht:smoke)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-marital:dage )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-ed )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-parity )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-age:dwt )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-marital:race )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-age:race )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dwt:wt.1 )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-gestation:drace )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-ded:dwt )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dwt:dage )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-gestation:smoke )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-ded:time )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-marital:ed )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dage:race )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dwt:ed )

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-gestation:parity)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-ed:smoke)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-age:drace)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dwt:race)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-dwt:smoke)

p<-vif(alteredModel)

p[which.max(p)]

alteredModel <-update(alteredModel,.~.-inc:ed)

p<-vif(alteredModel)

p[which.max(p)]

#finally, we finish deleting collinear variable and we do a AIC do a backward

#model selection and get the finalModel

finalModel <- step(alteredModel)

#check final model colinearity and all of them are less than 10, it works.

vif(finalModel)

#get summary of finalModel

summary(finalModel)

#use qq plot and Shapiro-Wilk normality test to test the normality

# because the p value in Shapiro-Wilk normality test is larger than 0.05,

# the data is normal, the QQ plot show the same result

qqnorm(resid(finalModel))

qqline(resid(finalModel))

shapiro.test(resid(finalModel))

hist(resid(finalModel))

plot(finalModel, which = 1:2)

# do Breusche-Pagan test with respect to fitted model

ncvTest(finalModel)

# null hypothesis: constant error variance. "If we have constant error variance

#then the variation in the residuals should be unrelated to any coveriant."

# null hypothesis is rejected since the p value is less than 0.05

# need to write durbinWatsonTest on model

durbinWatsonTest(finalModel)

#null hypothesis: error variances are uncorrelated, fail to reject the null hypothesis

#MT5761 notes page 22

Anova(finalModel)

#get the confidence interval

confint(finalModel)