Consider the `Cars93` dataframe from `library(MASS)`. It is of interest to predict the city mileage of a car based on the following predictors

- x_1: number of cylinders
- x_2: engine size
- x_3: horse power
- x_4: RPM
- x_5: number of passengers
- x_6: weight

To make such a prediction, build a regression model as follows

1. Find correlation among all variables. Which predictors are more correlated?
2. Fit a full linear regression model. Verify regression assumptions and identify outliers.
3. Interpret the regression equation.
4. Interpret the model adequacy values (MSE, R^2)
5. Estimate the mean city mileage of a 4-cylinder car with 2.3 engine size, 5500 RPM, 2950 pounds, 4 passengers, and 200 horse power. Also construct a 90% confidence interval for that mean city mileage
6. Predict the city mileage of a 4-cylinder car with 3.1 engine size, 6000 RPM, 3150 pounds, 5 passengers, and 225 horse power. Also construct a 95% prediction interval for that exact price.
7. Use `regsubsets()` from library `leaps` to find the best set of predictors suggested by adjusted-R^2. Use a model with these set of predictors to predict the city mileage of the car in item 6.
8. Use `set.seed(12)` to divide the data set into a training and a test set. Compare the MSPE of the full model and the model with the (adjusted-R^2) best predictors
9. Find the best set of predictors suggested by AIC. Compare its MSPE with that of the (adjusted-R^2) best predictors.

```
# cars93.r

library(PASWR2)   # checking.plots()
library(MASS)     # Cars93()
d0 = Cars93

# create data set
d1 = Cars93[,c(7,11,12,13,14,18,25)]    # or
d1 = subset(d0,select=c(MPG.city, Cylinders, EngineSize, Horsepower, RPM, Passengers, Weight))
d1$Cylinders = as.numeric(d1$Cylinders)

# 1) Correlations
#==============================================================
cor(d1)
#              MPG.city  Cylinders EngineSize   Horsepower         RPM   Passengers     Weight
# MPG.city    1.0000000 -0.7159745 -0.7100032 -0.672636151  0.36304513 -0.416855859 -0.8431385
# Cylinders  -0.7159745  1.0000000  0.7969007  0.798169593 -0.32424505  0.235510420  0.7801128
# EngineSize -0.7100032  0.7969007  1.0000000  0.732119730 -0.54789781  0.372721168  0.8450753
# Horsepower -0.6726362  0.7981696  0.7321197  1.000000000  0.03668821  0.009263668  0.7387975
# RPM         0.3630451 -0.3242451 -0.5478978  0.036688212  1.00000000 -0.467137627 -0.4279315
# Passengers -0.4168559  0.2355104  0.3727212  0.009263668 -0.46713763  1.000000000  0.5532730
# Weight     -0.8431385  0.7801128  0.8450753  0.738797516 -0.42793147  0.553272980  1.0000000
# most predictors highly correlated with response

cor(d1[-1,-1])
# predictors horsepower - engine size - weight correlated

# 2) full model
#===============================================================
m1=lm(MPG.city~.,data=d1)

# residuals plots
checking.plots(m1)
# assumptions hold
# outliers (39,42,83)
d1[c(39,42,83),]
#   MPG.city Cylinders EngineSize Horsepower  RPM Passengers Weight
#39       46         1        1.0         55 5700          4   1695
#42       42         2        1.5        102 5900          4   2350
#83       39         1        1.3         70 6000          4   1965

# 3) regression equation
#===================================================================
coef(m1)
 (Intercept)    Cylinders   EngineSize   Horsepower          RPM   Passengers       Weight
35.505593976 -0.450626941  1.554965579 -0.032923383  0.001825616 -0.153179626 -0.006539854

yhat = 35.505593976 -0.450626941 Cyl + 1.554965579 ESize -0.032923383 HP
      + 0.001825616 RPM - 0.153179626 Pass -0.006539854 Weight
```

```
# 4) model adequacy values - interpret
#==================================================================

# from summary() table

# Residual standard error: 3.012 on 86 degrees of freedom
# Multiple R-squared:  0.7314,    Adjusted R-squared:  0.7127
# F-statistic: 39.03 on 6 and 86 DF,  p-value: < 2.2e-16

# How much variation of Y is explained by m1? 0.7314, the R-squared
# Constant variance is estimated by  S = 3.012  (square root of MSE)

# 5) CI on mean city mileage
#==================================================================
newval=data.frame(Cylinders=4,EngineSize=2.3,Horsepower=200,RPM=5500,Passengers=4,Weight=2950)
predict(m1,newval,interval="conf",level=0.9)
#       fit      lwr      upr
# 1 20.83043 19.06601 22.59485

# 6) PI on city mileage of a new car
#==================================================================
newval=data.frame(Cylinders=4,EngineSize=2.3,Horsepower=200,RPM=5500,Passengers=4,Weight=2950)
predict(m1,newval,interval="pred")
#      fit     lwr      upr
#1 20.83043 14.4813 27.17956
```

```
# 7) Best set of predictors
#==================================================================
library(leaps)    # regsubsets()

# Select predictors
models=regsubsets(MPG.city~.,d1,nvmax=12)
summary(models)
# Selection Algorithm: exhaustive
         Cylinders EngineSize Horsepower RPM Passengers Weight
1  ( 1 ) " "       " "        " "        " " " "        "*"
2  ( 1 ) "*"       " "        " "        " " " "        "*"
3  ( 1 ) "*"       "*"        " "        " " " "        "*"
4  ( 1 ) " "       "*"        "*"        "*" " "        "*"
5  ( 1 ) "*"       "*"        "*"        "*" " "        "*"
6  ( 1 ) "*"       "*"        "*"        "*" "*"        "*"

# best predictor is Weight
# worst predictor is n. of Passengers

summary(models)$adjr2
# [1] 0.7077055 0.7133132 0.7123930 0.7166129 0.7157038 0.7126693
a=summary(models)$adjr2
which.max(a)   #  4

# best model is in row 4
# best model includes   EngineSize, Horsepower, RPM, Weight
# these variables are highly correlated with MPG.city

# compare to a non-optimal regression model
m0 = lm(MPG.city~Horsepower,d1)
summary(m0)

# Coefficients:
#              Estimate Std. Error t value Pr(>|t|)
# (Intercept) 32.746279   1.273229  25.719  < 2e-16 ***
# Horsepower  -0.072174   0.008323  -8.671 1.54e-13 ***

# Residual standard error: 4.181 on 91 degrees of freedom
# Multiple R-squared:  0.4524,    Adjusted R-squared:  0.4464
# F-statistic: 75.19 on 1 and 91 DF,  p-value: 1.537e-13

# This R-squared much smaller
```

```
# function predict.regsubsets()

predict.regsubsets <- function(object, newdata, id, ...)
{
  form <- as.formula(object$call[[2]])
  mat <- model.matrix(form, newdata)
  coefi = coef(object, id = id)
  xvars <- names(coefi)
  mat[, xvars]%*%coefi
}

newval=data.frame(MPG.city=66,Cylinders=4,EngineSize=2.3,Horsepower=200,
                  RPM=5500,Passengers=4,Weight=2950)

predict.regsubsets(models,newval,id = 4)
#          [,1]
# [1,] 21.06858

# 8) Validation and MSPE
#===================================================================

# full model
n = nrow(d1)    # 93
n/2             # [1] 46.5
ceiling(n/2)   # [1] 47

set.seed(12)
train = sample(1:n,47)       # train row numbers

d1train = d1[train,]
d1test  = d1[-train,]
dim(d1train)     # [1] 47  5
dim(d1test)      # [1] 46  5

# model with all 6 variables
m1 = lm(MPG.city~.,d1train)
yhat1 = predict(m1,d1test)
ytest1 = d1test$MPG.city
# mspe
mean((yhat1-ytest1)^2)       # 9.736857
```

```
# best model
d2 = subset(d1,select=c(MPG.city, EngineSize, Horsepower, RPM, Weight))
d2train = d2[train,]
d2test  = d2[-train,]

m2 = lm(MPG.city~.,d2train)
yhat2  = predict(m2,d2test)
ytest2 = d2test$MPG.city
# mspe
mean((yhat2-ytest2)^2)       # 9.389211

# If summary(m1) is compared against summary(m2)
# comparisons are for training R-squared values

# plot
plot(yhat2~ytest2,pch=19,cex=0.5,ylim=c(10,50),xlim=c(10,50))
abline(0,1)
grid()
text(yhat2~ytest2,labels=rownames(d2test),cex=0.6,pos=1,offset=0.25)



# 9) stepAIC
#=================================================================

step1 = stepAIC(m1)
coef(step1)
#  (Intercept)   Horsepower          RPM       Weight
# 35.280443949 -0.026385811  0.001349840 -0.005293273

# AIC model
d3 = subset(d1,select=c(MPG.city, Horsepower, RPM, Weight))
d3train = d3[train,]
d3test  = d3[-train,]

m3 = lm(MPG.city~.,d3train)
yhat3  = predict(m3,d3test)
ytest3 = d3test$MPG.city
# mspe
mean((yhat3-ytest3)^2)       # 9.602604

coef(m3)
#  (Intercept)   Horsepower          RPM       Weight
# 35.280443949 -0.026385811  0.001349840 -0.005293273

# thus, adj-R2 suggested a better model than AIC
```

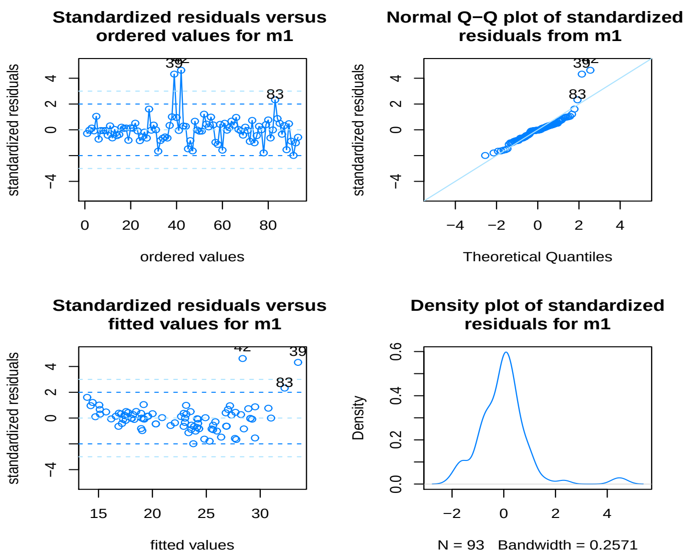

Figure 1: Residual analysis for model with all 6 predictors

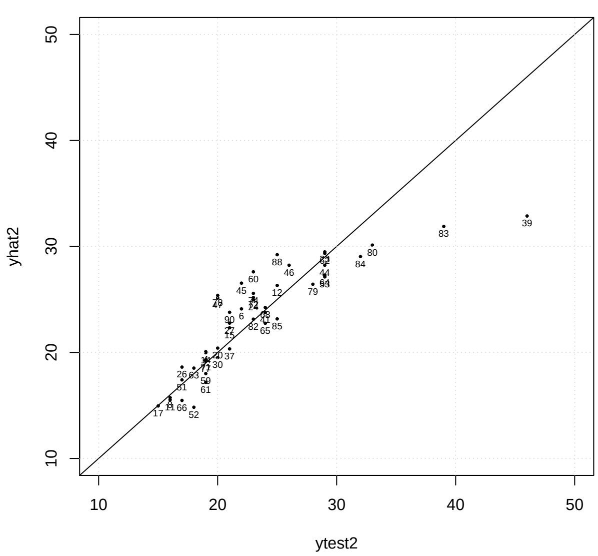

Figure 2: Scatterplot of yhat vs. y values from test set for model with best 4 predictors