Consider the Cars93 data set. It is of interest to predict the city mileage of a car based on the following predictors

- x_1: number of cylinders
- x_2: horse power
- x_3: RPM
- x_4: number of passengers
- x_5: width

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

1. Fit a full linear regression model
2. 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 5500 RPM, 2950 pounds, 4 passengers, width 62, and 200 horse power. Also construct a 90% confidence interval for that mean city mileage
6. Predict the city mileage of a 6-cylinder car with 6000 RPM, 3150 pounds, 5 passengers, width 75, 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

```
library(PASWR2)   # checking.plots()
library(MASS)     # Cars93()
d0 = Cars93
d1 = subset(d0,select=c(MPG.city, Cylinders, Horsepower, RPM, Passengers, Width))
str(d1)
# 'data.frame':   93 obs. of  6 variables:
# $ MPG.city  : int  25 18 20 19 22 22 19 16 19 16 ...
# $ Cylinders : Factor w/ 6 levels "3","4","5","6",..: 2 4 4 4 2 2 4 4 4 5 ...
# $ Horsepower: int  140 200 172 172 208 110 170 180 170 200 ...
# $ RPM       : int  6300 5500 5500 5500 5700 5200 4800 4000 4800 4100 ...
# $ Passengers: int  5 5 5 6 4 6 6 6 5 6 ...
# $ Width     : int  68 71 67 70 69 69 74 78 73 73 ...

d1$Cylinders = as.numeric(d1$Cylinders)

# correlations
#=============================================================
cor(d1)
#              MPG.city  Cylinders   Horsepower         RPM   Passengers      Width
# MPG.city    1.0000000 -0.7159745 -0.672636151  0.36304513 -0.416855859 -0.7205344
# Cylinders  -0.7159745  1.0000000  0.798169593 -0.32424505  0.235510420  0.7731293
# Horsepower -0.6726362  0.7981696  1.000000000  0.03668821  0.009263668  0.6444134
# RPM         0.3630451 -0.3242451  0.036688212  1.00000000 -0.467137627 -0.5397211
# Passengers -0.4168559  0.2355104  0.009263668 -0.46713763  1.000000000  0.4899786
# Width      -0.7205344  0.7731293  0.644413421 -0.53972113  0.489978637  1.0000000

# Cylinders and Width most highly correlated with MPG.city
# predictors Horsepower - Cylinders correlated

# full model
#==============================================================
m1=lm(MPG.city~.,data=d1)
library(car)
vif(m1)
# Cylinders Horsepower        RPM Passengers      Width
#  4.458918   5.146875   2.426449   1.670997   5.067659

# residuals plots
#==============================================================
checking.plots(m1)

sres = rstandard(m1)
shapiro.test(sres)
#Shapiro-Wilk normality test
#W = 0.87147, p-value = 1.834e-07

# Shapiro concludes normality assumption does not hold
# let us see if it is due to outliers
```

```
# 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

d2=d1[-c(39,42,83),]
m2=lm(MPG.city~.,d2)
sres = rstandard(m2)
shapiro.test(sres)
#Shapiro-Wilk normality test
#W = 0.98061, p-value = 0.1992

# p-value is not small, cannot reject normality assumption

# We will keep outliers, and continue with the analysis, however


# regression equation
#===================================================================
coef(m1)
# (Intercept)    Cylinders   Horsepower          RPM   Passengers        Width
#29.697530356 -0.481561319 -0.064248261  0.002071047 -1.546427032  0.003544324

# yhat = 29.697530356 -0.481561319 Cyl - 0.064248261 HP + 0.002071047 RPM - 1.546427032 Pass + 0.0

# MPG.city decreases by 0.4815 for each additional cylinder
# if all other predictors are held constant


# Test Regression relation
#===================================================================

# from summary() table

# Residual standard error: 3.307 on 87 degrees of freedom
# Multiple R-squared:  0.6726,    Adjusted R-squared:  0.6537
# F-statistic: 35.74 on 5 and 87 DF,  p-value: < 2.2e-16

# Fo = MSR/MSE = 35.74
# p-value small, reject Ho: beta1 = ... = beta-p = 0
# There is enough evidence of a regression relation
```

```
# test individual predictors

#Coefficients:
#             Estimate Std. Error t value Pr(>|t|)
#(Intercept) 29.697530  15.249965   1.947  0.05471 .
#Cylinders   -0.481561   0.616077  -0.782  0.43654
#Horsepower  -0.064248   0.014934  -4.302 4.41e-05 ***
#RPM          0.002071   0.000900   2.301  0.02377 *
#Passengers  -1.546427   0.428957  -3.605  0.00052 ***
#Width        0.003544   0.205381   0.017  0.98627

# Width and Cylinders should be removed from model
# note that they have largest correlation with MPG.city

# model adequacy values - interpret
#====================================================================

# How much variation of Y is explained by m1? 0.6726, the R-squared
# Residual Standard error  S = 3.307  (square root of MSE)

# CI on mean city mileage
#====================================================================
newval=data.frame(Cylinders=4,Horsepower=200,RPM=5500,Passengers=4,Width=62)
predict(m1,newval,interval="conf",level=0.9)
#        fit      lwr      upr
# 1 20.34643 17.25283 23.44003

# PI on city mileage of a new car
#====================================================================
newval=data.frame(Cylinders=6,Horsepower=225,RPM=6000,Passengers=5,Width=75)
predict(m1,newval,interval="pred")
#        fit      lwr    upr
# 1 17.31227 10.04655 24.578

# Select predictors

library(leaps)    # regsubsets()
models=regsubsets(MPG.city~.,d1,nvmax=12)
summary(models)
# Selection Algorithm: exhaustive
#          Cylinders Horsepower RPM Passengers Width
# 1  ( 1 ) " "       " "        " " " "        "*"
# 2  ( 1 ) " "       "*"        " " "*"        " "
# 3  ( 1 ) " "       "*"        "*" "*"        " "
# 4  ( 1 ) "*"       "*"        "*" "*"        " "
# 5  ( 1 ) "*"       "*"        "*" "*"        "*"
```

```
summary(models)$adjr2
# 0.5138860 0.6126458 0.6590680 0.6576678 0.6537342

a=summary(models)$adjr2
which.max(a)   #  3

# best model is in row 3
# best model includes   Horsepower, RPM, Passengers
# these variables are not most correlated with MPG.city

# best model
#===================================================================

m2 = lm(MPG.city~Horsepower+RPM+Passengers,data=d1)
summary(m2)
# Coefficients:
#              Estimate Std. Error t value Pr(>|t|)
#(Intercept) 28.4374921  4.6915814   6.061 3.19e-08 ***
#Horsepower  -0.0728706  0.0065393 -11.144  < 2e-16 ***
#RPM          0.0023631  0.0006491   3.641 0.000456 ***
#Passengers  -1.5867070  0.3725685  -4.259 5.08e-05 ***

#Residual standard error: 3.281 on 89 degrees of freedom
#Multiple R-squared:  0.6702,    Adjusted R-squared:  0.6591
#F-statistic: 60.28 on 3 and 89 DF,  p-value: < 2.2e-16

# best model has adj R2 0.659
# best model explains 67% of MPG.city variability

# plot
#===================================================================
yhat = fitted(m2)
yobs = d1$MPG.city
bounds = c(10,50)
plot(yhat~yobs,pch=19,cex=0.5,ylim=bounds,xlim=bounds)
abline(0,1)
grid()
text(yhat~yobs,labels=rownames(d1),col="red",cex=0.6,pos=1,offset=0.25)
```

```
# Apparent best model (based on correlations with MPG.city)
#===================================================================
d3 = subset(d0,select=c(MPG.city, Cylinders, Width))
d3$Cylinders = as.numeric(d3$Cylinders)
m3=lm(MPG.city~.,data=d3)
vif(m3)
# Cylinders     Width
#  2.485886  2.485886

# Even though they are correlated, the VIFs are small

cor(d3)
#             MPG.city  Cylinders      Width
#MPG.city   1.0000000 -0.7159745 -0.7205344
#Cylinders -0.7159745  1.0000000  0.7731293
#Width     -0.7205344  0.7731293  1.0000000


summary(m3)
# Residual standard error: 3.674 on 90 degrees of freedom
# Multiple R-squared:  0.5819,    Adjusted R-squared:  0.5727
# F-statistic: 62.64 on 2 and 90 DF,  p-value: < 2.2e-16

# Apparent best model with adj R2 0.5727    (not as good as the best model)
```

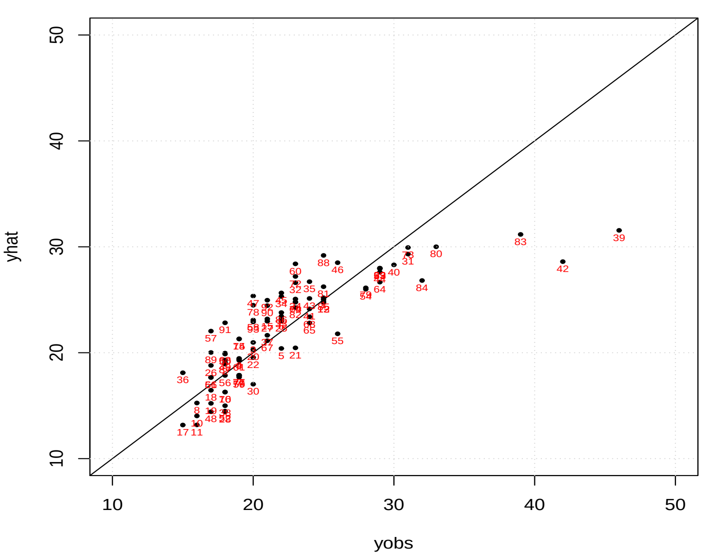

Figure 1: Predicted vs observed mileage for best model