Lab06

Yixuan Li, UPI:yil845

20 September 2021

* The purpose of this lab is to get experience for data exploration, linear regression, subset selection, regularisation, and prediction.

* The Data: The data is from UCI Machine Learning Repository about the Automobile data set.

Tasks

Import

1, Import datafile “automobile-original.csv”,check data structure

```
b<-read.csv("automobile-original.csv",stringsAsFactors = FALSE)
```

- Processing data, 1) remove all “?” in dataframe; 2) choose engine.location of “front” only; 3)delete column engine.location; 4) change column class, col2,col21:22,col25 to integer, col18:19 to numeric; 5) write file“automobile.csv” without indexing rownames.

```
btemp<- b[b$normalized.losses!="?" & b$num.of.doors!="?" & b$bore!="?" & b$stroke!="?" & b$horse
power!="?" & b$peak.rpm!="?" & b$engine.location =="front",]
btemp<- btemp[,-9]
btemp[,c(2,21:22,25)]<- sapply(btemp[,c(2,21:22,25)],as.integer)
btemp[,c(18:19)]<-sapply(btemp[,c(18,19)],as.numeric)
write.csv(btemp,"automobile.csv",row.names = FALSE)
```

- Import data “automobile.csv”, and compare it with data imported from “automobile-subset.csv”.

```
b2<- read.csv("automobile.csv",stringsAsFactors = TRUE)
dim(b2)
```

```
## [1] 159  25
```

```
b1<-read.csv("automobile-subset.csv",stringsAsFactors = TRUE)
all(b1 == b2)
```

```
## [1] TRUE
```

Explore

2, Write R code to answer the following questions:

- What is the mean price of all vehicles?

```
mean(b2$price)
```

```
## [1] 11445.73
```

- How many vehicles have 4 doors ?

```
nrow(b2[b2$num.of.doors=="four",])
```

```
## [1] 95
```

- What are the different engine types among the observations?

```
unique(b2$engine.type)
```

```
## [1] ohc  l    dohc ohcv ohcf
## Levels: dohc l ohc ohcf ohcv
```

- How many vehicles have a price higher than $20000?

```
nrow(b2[b2$price > 20000,])
```

```
## [1] 13
```

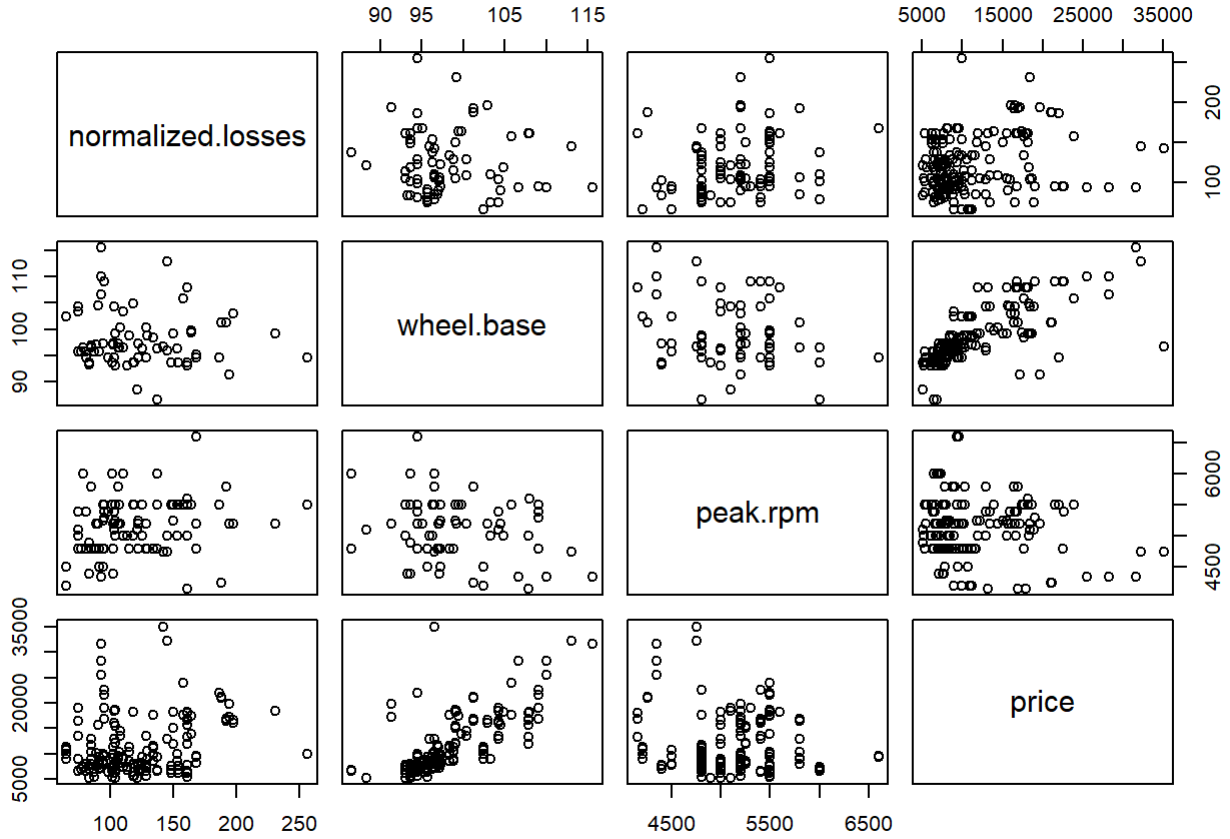
- What is the mean price for “4wd”?

```
m<-b2[b2$drive.wheels=="4wd",]
mean(m$price)
```

```
## [1] 10241
```

3, Produce pairwise scatterplots between variables normalized.losses, wheel.base, peak.rpm and price. The plot shows a linear dependence between price and wheel.base. However, for normalized.losses and peak.rpm, we hardly see any linear correspondence.

```
bp<-b2[,c(2,9,22,25)]
pairs(bp)
```



Linear regression

4,Produce the full linear regression model with all variables included.

The outcome shows the validness of linear regression model based on ordinary least square (OLS) estimation. It shows fomula, residuals calculated, coefficiants which includes estimated coefficients, standard Error, t values of t test and p value of t test. The stars shows how significant the p values are, * means significant with 0.05< p < 0.01; ** means middle significant with 0.01 < p < 0.001; *** means highly significiant with 0 < p < 0.001; “.” means p value between 0.05 and 0.1. Multiple R^2 is the correlation coefficient of overall fitting (>0.97) and adjusted R^2, indicating a very good linear fitting. We notice “NA” are generated due to poor linear relationships of the relevant parameters. To improve our fitting, we need to remove those parameters.

```
r<-lm(price~.-symboling, data=b2)
summary(r)
```

```
##
## Call:
## lm(formula = price ~ . - symboling, data = b2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2949.1  -587.9     0.0   649.4  2260.2
##
## Coefficients: (3 not defined because of singularities)
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            2.305e+04  1.545e+04   1.492 0.138508
## normalized.losses      5.561e+00  6.186e+00   0.899 0.370596
## makebmw                3.725e+02  1.697e+03   0.220 0.826622
## makechevrolet         -4.733e+03  1.773e+03  -2.669 0.008779 **
## makedodge             -6.198e+03  1.532e+03  -4.046 9.79e-05 ***
## makehonda             -1.574e+03  1.691e+03  -0.931 0.354043
## makejaguar             2.442e+03  2.866e+03   0.852 0.396016
## makemazda             -4.053e+03  1.511e+03  -2.683 0.008442 **
## makemercedes-benz      2.560e+03  1.518e+03   1.686 0.094658 .
## makemitsubishi        -6.321e+03  1.542e+03  -4.098 8.09e-05 ***
## makenissan            -3.680e+03  1.410e+03  -2.611 0.010324 *
## makepeugot            -5.136e+03  4.700e+03  -1.093 0.276957
## makeplymouth          -6.015e+03  1.548e+03  -3.884 0.000177 ***
## makeporsche            4.840e+03  2.043e+03   2.368 0.019642 *
## makesaab              -4.050e+02  1.544e+03  -0.262 0.793625
## makesubaru            -7.306e+03  2.241e+03  -3.260 0.001489 **
## maketoyota            -5.859e+03  1.526e+03  -3.841 0.000207 ***
## makevolkswagen        -4.297e+03  1.349e+03  -3.185 0.001892 **
## makevolvo             -2.855e+03  1.698e+03  -1.682 0.095536 .
## fuel.typegas          -1.066e+04  5.079e+03  -2.099 0.038125 *
## aspirationturbo        2.169e+03  5.793e+02   3.743 0.000293 ***
## num.of.doorstwo       -8.410e+02  3.563e+02  -2.360 0.020057 *
## body.stylehardtop     -5.625e+03  1.367e+03  -4.116 7.56e-05 ***
## body.stylehatchback   -5.732e+03  1.328e+03  -4.316 3.53e-05 ***
## body.stylesedan       -5.698e+03  1.377e+03  -4.137 6.98e-05 ***
## body.stylewagon       -5.644e+03  1.401e+03  -4.028 0.000105 ***
## drive.wheelsfwd       -2.760e+01  6.427e+02  -0.043 0.965823
## drive.wheelsrwd        1.977e+03  9.507e+02   2.079 0.039967 *
## wheel.base             3.188e+02  8.129e+01   3.922 0.000155 ***
## length                -7.656e+01  3.781e+01  -2.025 0.045325 *
## width                  2.435e+02  2.037e+02   1.196 0.234484
## height                -3.349e+02  1.181e+02  -2.836 0.005455 **
## curb.weight            5.210e+00  1.288e+00   4.046 9.81e-05 ***
## engine.typel          -4.677e+03  3.651e+03  -1.281 0.202895
## engine.typeohc        -1.913e+03  9.912e+02  -1.931 0.056160 .
## engine.typeohcf               NA         NA      NA       NA
## engine.typeohcv       -1.334e+03  1.145e+03  -1.165 0.246771
## num.of.cylindersfive  -4.102e+03  2.535e+03  -1.618 0.108500
## num.of.cylindersfour  -4.684e+03  3.221e+03  -1.454 0.148797
## num.of.cylinderssix   -2.973e+03  2.878e+03  -1.033 0.303876
## num.of.cylindersthree         NA         NA      NA       NA
## engine.size           -1.253e+01  2.316e+01  -0.541 0.589662
```

```
## fuel.system2bbl        2.067e+03  1.006e+03   2.055 0.042283 *
## fuel.systemidi                NA         NA      NA       NA
## fuel.systemmfi         3.456e+03  1.885e+03   1.833 0.069502 .
## fuel.systemmpfi        2.600e+03  1.072e+03   2.424 0.016997 *
## fuel.systemspdi        1.081e+03  1.286e+03   0.840 0.402560
## bore                  -8.830e+02  1.420e+03  -0.622 0.535226
## stroke                -5.652e+02  9.422e+02  -0.600 0.549856
## compression.ratio     -7.012e+02  3.791e+02  -1.850 0.067084 .
## horsepower            -2.017e+01  1.899e+01  -1.062 0.290628
## peak.rpm              -5.391e-01  5.600e-01  -0.963 0.337901
## city.mpg              -1.561e+02  1.016e+02  -1.535 0.127596
## highway.mpg            1.281e+02  8.776e+01   1.460 0.147255
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1159 on 108 degrees of freedom
## Multiple R-squared:  0.9734, Adjusted R-squared:  0.9611
## F-statistic: 79.05 on 50 and 108 DF,  p-value: < 2.2e-16
```

5, Remove NA values & count significant P values (< 0.05), additionally normalized.losses removed, because of the plot and t value in Q4. A better linear regression is generated after removing failed parameters.

```
lr<-lm(price~.-symboling -normalized.losses -engine.type -num.of.cylinders -fuel.system, data=b
2)
summary(lr) ## comment on outcome
```

```
##
## Call:
## lm(formula = price ~ . - symboling - normalized.losses - engine.type -
##     num.of.cylinders - fuel.system, data = b2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2862.24  -713.33   -25.37   771.77  3019.25
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         13368.2796 14319.5060   0.934 0.352415
## makebmw              1302.9182  1339.3589   0.973 0.332630
## makechevrolet       -4437.0773  1325.1646  -3.348 0.001089 **
## makedodge           -6196.0728  1044.6147  -5.931 3.02e-08 ***
## makehonda           -4187.1955   945.7129  -4.428 2.13e-05 ***
## makejaguar           2699.1243  2068.1020   1.305 0.194369
## makemazda           -3832.8735   919.7958  -4.167 5.88e-05 ***
## makemercedes-benz    2950.4883  1277.5168   2.310 0.022637 *
## makemitsubishi      -6886.4754   984.0162  -6.998 1.64e-10 ***
## makenissan          -3848.5948   922.6569  -4.171 5.79e-05 ***
## makepeugot          -6397.2324  1310.1481  -4.883 3.29e-06 ***
## makeplymouth        -6302.8597  1071.8056  -5.881 3.83e-08 ***
## makeporsche          3888.8119  1716.5909   2.265 0.025295 *
## makesaab             -412.3341  1222.0152  -0.337 0.736394
## makesubaru          -4997.9941  1408.4027  -3.549 0.000555 ***
## maketoyota          -5478.0484   964.1106  -5.682 9.63e-08 ***
## makevolkswagen      -4148.5040   912.1549  -4.548 1.31e-05 ***
## makevolvo           -2797.8073  1123.0805  -2.491 0.014110 *
## fuel.typegas        -2859.1271  4212.7345  -0.679 0.498654
## aspirationturbo      1658.3804   584.1557   2.839 0.005324 **
## num.of.doorstwo      -569.8647   356.0451  -1.601 0.112130
## body.stylehardtop   -6511.8512  1177.1217  -5.532 1.91e-07 ***
## body.stylehatchback -6034.1063  1078.1612  -5.597 1.42e-07 ***
## body.stylesedan     -5772.8527  1149.8562  -5.020 1.83e-06 ***
## body.stylewagon     -5832.2150  1209.3778  -4.822 4.24e-06 ***
## drive.wheelsfwd      -264.5284   669.1118  -0.395 0.693298
## drive.wheelsrwd      1409.0973   929.4784   1.516 0.132169
## wheel.base            283.2453    75.8379   3.735 0.000290 ***
## length                -78.0576    36.3489  -2.147 0.033785 *
## width                 205.6299   189.9011   1.083 0.281075
## height               -365.4615   110.1231  -3.319 0.001201 **
## curb.weight             5.2188     1.2756   4.091 7.85e-05 ***
## engine.size             9.9592    16.9817   0.586 0.558671
## bore                -2286.1326   940.0653  -2.432 0.016510 *
## stroke               -628.0364   845.7784  -0.743 0.459215
## compression.ratio    -259.5834   317.9962  -0.816 0.415954
## horsepower             10.1422    15.3965   0.659 0.511340
## peak.rpm               -0.1438     0.4844  -0.297 0.767159
## city.mpg              -72.3725   101.5658  -0.713 0.477509
## highway.mpg            62.0067    89.0011   0.697 0.487352
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1230 on 119 degrees of freedom
## Multiple R-squared:  0.967,  Adjusted R-squared:  0.9562
## F-statistic: 89.47 on 39 and 119 DF,  p-value: < 2.2e-16
```

```
# count how many P values show significance, in total 24.
sum(summary(lr)$coefficients[,4]<0.05)
```

```
## [1] 24
```

6, calculate MSE based on full OLS model.

```
MSE <- function(o, p) mean((o - p)^2)
MSE(predict(lr,b2),b2$price)
```

```
## [1] 1132209
```

```
##RMSE
sqrt(MSE(predict(lr,b2),b2$price))
```

```
## [1] 1064.053
```

subset selection

7,produce a subset linear regression model, using the backward selection and the AIC.

```
rl.bac<- regsubsets(price~.-symboling -normalized.losses -engine.type -num.of.cylinders -fuel.sy
stem, data=b2, nvmax=50, method="backward")
rl.b=summary(rl.bac)
```

calculate AIC based on formula: BIC(k)-log(n)*k*+2k

```
# count number of fitting parameters
k<-1:(ncol(rl.b$which)-1)
aic<-rl.b$bic-log(159)*k+2*k
aic
```

```
##  [1] -247.7153 -257.4944 -268.2075 -271.0805 -278.4920 -286.5273 -292.8073
##  [8] -298.9789 -307.4123 -312.6586 -321.7452 -340.5615 -351.6461 -366.2102
## [15] -374.7714 -401.7675 -409.4260 -411.8684 -413.0535 -426.6856 -445.2621
## [22] -455.8051 -457.2931 -461.9682 -464.8567 -470.9015 -472.8157 -474.7675
## [29] -474.1080 -472.7594 -472.2000 -471.2258 -470.0883 -468.1549 -466.7159
## [36] -464.9864 -463.2177 -461.3062 -459.4239
```

8,Apply the AIC-selected model to the data and compute the resulting MSE.

```
# model matrix
b2.matrix = model.matrix(price~. -symboling -normalized.losses -engine.type -num.of.cylinders -f
uel.system, data=b2)

j<-which.min(aic)
beta=coef(rl.bac,j)
b2j.matrix=b2.matrix[,names(beta)]
yhat = drop(b2j.matrix %*% beta)
resid = b2$price - yhat
## MSE
mean(resid^2)
```

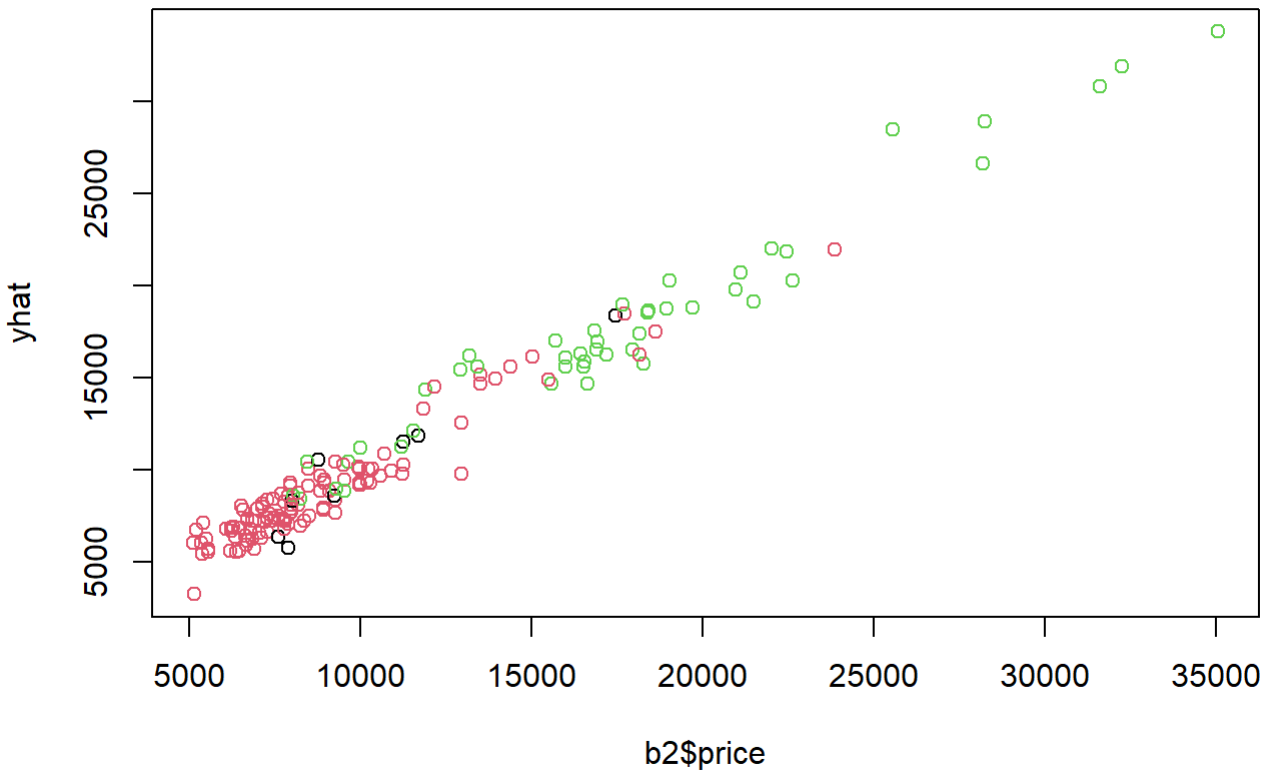
```
## [1] 1180614
```

```
##RMSE
sqrt(mean(resid^2))
```

```
## [1] 1086.561
```

9, The following codes create a plot that shows the predictions of min(AIC)-selected model against the response variable (price), using different colors for different levels of drive.wheels.

```
# the majority of red spot - forward-wheels-drive cars (fwd) dominated at lower price range,wher
eas rear-wheels-drive (rwd) cars at higher price range.
plot(b2$price,yhat,col=as.numeric(factor(b2$drive.wheels)),)
```



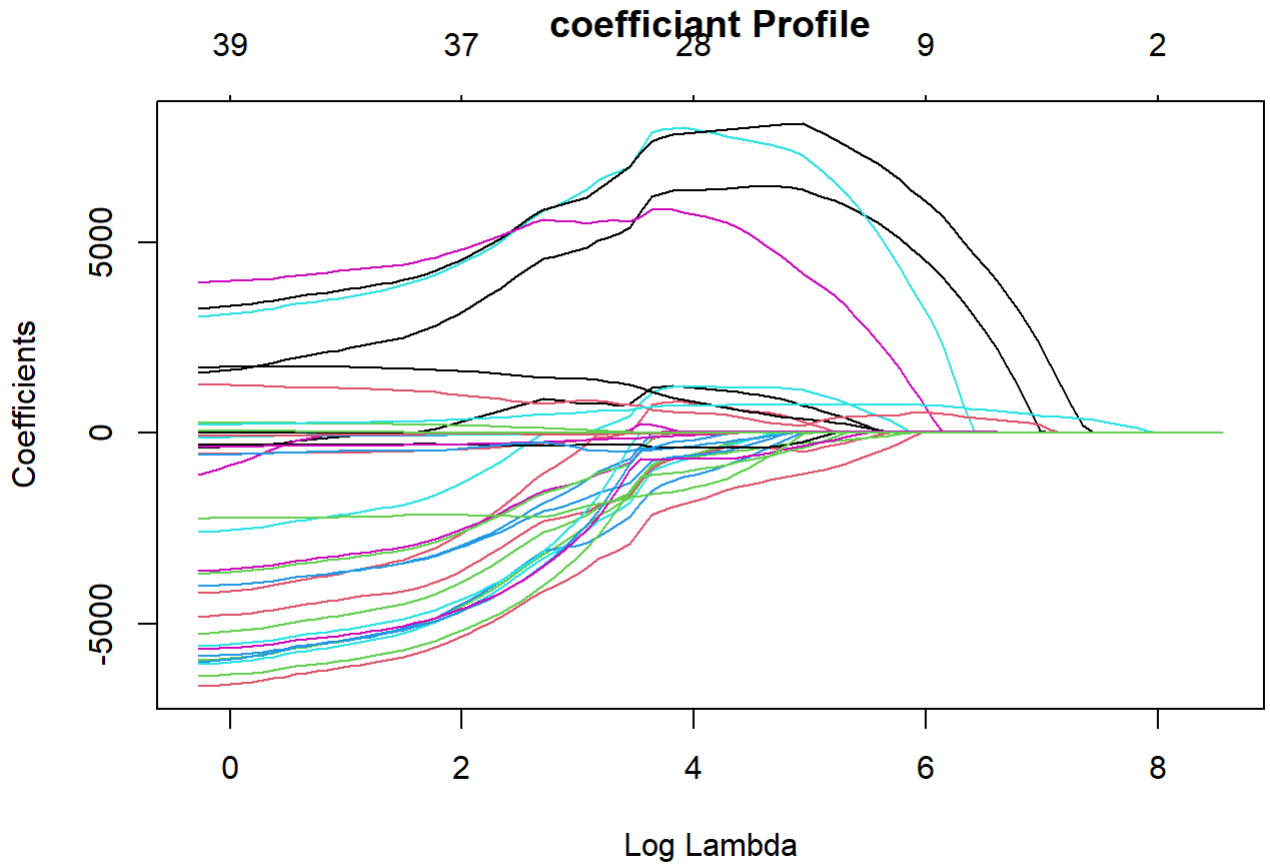

```
#Levels(b2$drive.wheels); 4wd(black),fwd(red),rwd(green)
```

10,For the full data selected, compute the Lasso model.

```
x<-b2.matrix[,-1] # remove intercept
y<-b2$price
lr.lasso<-glmnet(x,y,alpha=1)
#lr.lasso
```

11, these codes Create a coefficient profile plot of the coefficient price that varies with the value of log(λ).

```
plot(lr.lasso,xvar="lambda" ,main="coefficiant Profile")
```



12, Choose 5 different λ-values within a seemingly reasonable range (with roughly 5 to 30 variables included) and compute the MSEs of the corresponding 5 Lasso subset models. Write R code to find out how many variables (excluding the intercept) are included in each Lasso subset model.

```
## the chosen lambda range is lambda=c(894.0, 424.7, 243.0, 139.1, 87.3)
j= c(20,28,34,40,45)
coef(lr.lasso)[,j]
```

```
## 40 x 5 sparse Matrix of class "dgCMatrix"
##                               s19          s27           s33           s39
## (Intercept)         -38280.344076 -50215.28738 -51473.492373 -4.960554e+04
## makebmw               1229.596283   4370.48436   5669.295331  6.364890e+03
## makechevrolet            .             .            .         3.035551e+02
## makedodge                .             .            .         .
## makehonda                .             .            .         .
## makejaguar               .          2916.95732   5618.922665  7.278362e+03
## makemazda                .             .            .         .
## makemercedes-benz     3235.559209   5934.48009   7220.299810  8.096481e+03
## makemitsubishi           .             .         -604.874979 -1.080761e+03
## makenissan               .             .            .         .
## makepeugot               .             .            .        -5.557147e-01
## makeplymouth             .             .            .         .
## makeporsche              .           406.58087   2729.053098  4.175821e+03
## makesaab                 .             .          184.208102  7.253933e+02
## makesubaru               .             .         -124.102807 -4.938159e+02
## maketoyota               .             .         -130.009143 -2.593423e+02
## makevolkswagen           .             .            .         .
## makevolvo                .             .          567.914622  1.123092e+03
## fuel.typegas             .             .            .         .
## aspirationturbo          .             .          142.706188  3.651684e+02
## num.of.doorstwo          .             .            .         .
## body.stylehardtop        .             .            .        -3.982314e+02
## body.stylehatchback      .             .            .         .
## body.stylesedan          .             .            .         .
## body.stylewagon          .             .            .        -3.445836e+02
## drive.wheelsfwd          .             .            .        -2.439144e+02
## drive.wheelsrwd        291.549049    522.19358    441.236599  1.860512e+02
## wheel.base               .             .            .         .
## length                   .             .            .         .
## width                  521.399405    720.09394    748.434023  7.291552e+02
## height                   .             .            .         .
## curb.weight              5.033388      4.50707      4.080575  3.918258e+00
## engine.size             13.273400      .            .         .
## bore                     .             .            .        -1.828736e+01
## stroke                   .             .            .         .
## compression.ratio        .             .            .         .
## horsepower              13.887254     29.80442     33.660370  3.425659e+01
## peak.rpm                 .             .            .         .
## city.mpg                 .             .            .         .
## highway.mpg              .             .            .         .
##                             s44
## (Intercept)         -47299.60596
## makebmw               6442.54440
## makechevrolet          578.13209
## makedodge             -163.39803
## makehonda                .
## makejaguar            7668.08316
## makemazda                .
## makemercedes-benz     7993.89365
## makemitsubishi       -1360.67778
```

```
## makenissan              .
## makepeugot           -612.97492
## makeplymouth         -185.07675
## makeporsche          5211.92044
## makesaab             1037.94142
## makesubaru           -373.80240
## maketoyota           -436.49050
## makevolkswagen       -321.34242
## makevolvo            1191.37814
## fuel.typegas            .
## aspirationturbo       558.52255
## num.of.doorstwo         .
## body.stylehardtop    -713.37095
## body.stylehatchback  -156.07578
## body.stylesedan         .
## body.stylewagon      -608.47415
## drive.wheelsfwd      -394.54989
## drive.wheelsrwd       364.46944
## wheel.base              .
## length                  .
## width                 736.61807
## height                  .
## curb.weight             4.13992
## engine.size             .
## bore                 -907.56693
## stroke                  .
## compression.ratio       .
## horsepower            31.89615
## peak.rpm                .
## city.mpg                .
## highway.mpg             .
```

```
#lr.lasso
yhat.lasso = drop(predict(lr.lasso, s=j, alpha=1, newx=x))
# residue matrix at five chosen lambdas
resid.lasso = b2$price - yhat.lasso

## compute MSE of five lasso subset models based on the chosen lambda
mse.lasso = sapply(1:5, function(i) mean(resid.lasso[,i]^2))
mse.lasso
```

```
## [1] 1407611 1584839 1730780 1807329 1836023
```

```
##RMSE
sqrt(mse.lasso)
```

```
## [1] 1186.428 1258.904 1315.591 1344.369 1354.999
```

Summary:

In this lab, we have learn multiple linear regression based on different models,OLS ordinary least squares model, backward stepwise fitting and lasso regularization method. Firstly, we read and subset data automobile-orginal file to remove NA values, select column based on chosen value, remove this column and change column class in order that the obtained data is the same as automobile-subset. Secondly, we used this data for linear fitting based on least square model lm(), and discarded invalid linear fitting parameters,“symboling”,“engine.type”, “num.of.cylinders” and “fuel.system”; thirdly we use backward stepwise method regsubsets(),use obtained BIC values to compute AIC and calculate residule matrix based on min(AIC)(minimize coefficient) in order to compute MSE; lastly we use Lasso regression glmnet(..alpha=1) for linear fitting, and we choose five lambdas which includes 7,8,15,19,25 variables in the regression for MSEs calculation. We have three plots generated; 1, from pairwise scatterplots of variatbles “normalized.losses”, “wheel.base”, “peak.rpm” and “price”, visually we can see clear linear dependence of wheel.base and price, ; 2, from the plot of the predictions of min(AIC) model fitted with the orginal price, we can see forward-wheel-drive cars(“rwd”,red) is dominant in lower price range, and rear-wheel-drive (“rwd”,green) in the upper price range, model fitting is pretty well; 3,a coefficient profile plot of the coefficient price that varies with the value of log(λ) in Lasso regression, it shows when log(lambda) is small, the coefficient is large& highly diversed; when log(lambda) is bigger, the coefficient is getting smaller and close to zero. From the MSEs generated from 3 models, we see the lowest MSE of least squares fitting is ~1.1M, the second is backward fitting ~1.18M, the third one is Lasso fitting, ~ 1.4-1.8M respectively.In this case, we do not see improvement of accuracy of OLS fitting, indicating parameters donot have cross correlation and OLS ($R^2>0.97$) is the best linear fitting model sofar.