lab6

YILIN LI

2019/11/3

Lab 6

Q1

```
library(ISLR)
library(tidyverse)
```

```
## -- Attaching packages -------------------------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.1     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts -------------------------------------------------------------------------------------- t
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(DAAG)
```

```
## Loading required package: lattice
```

```
# 1st part
d <- read.csv("https://bit.ly/36kibHZ")
d["MAPE"] <- d$Price/d$Earnings_10MA_back
summary(d)
```

```
##       Date          Price            Earnings      Earnings_10MA_back
##  Min.   :1871   Min.   :  64.76   Min.   :  4.01   Min.   : 8.51
##  1st Qu.:1907   1st Qu.: 161.84   1st Qu.: 11.94   1st Qu.:13.89
##  Median :1943   Median : 237.53   Median : 18.89   Median :17.48
##  Mean   :1943   Mean   : 442.82   Mean   : 27.02   Mean   :25.70
##  3rd Qu.:1979   3rd Qu.: 553.62   3rd Qu.: 36.59   3rd Qu.:37.19
##  Max.   :2015   Max.   :2056.51   Max.   :103.17   Max.   :77.00
##                                                    NA's   :120
##   Return_cumul       Return_10_fwd          MAPE
##  Min.   :    0.99   Min.   :-0.05925   Min.   : 4.785
##  1st Qu.:   15.13   1st Qu.: 0.03400   1st Qu.:11.708
##  Median :   95.89   Median : 0.06827   Median :15.947
##  Mean   : 1456.57   Mean   : 0.06788   Mean   :16.554
##  3rd Qu.: 1060.67   3rd Qu.: 0.10481   3rd Qu.:19.959
##  Max.   :12950.92   Max.   : 0.19960   Max.   :44.196
##                     NA's   :120        NA's   :120
```

```
d <-na.omit(d)

# 2nd part
m1 <- lm(Return_10_fwd ~ MAPE, data = d)
summary(m1)
```

```
##
```

```
## Call:
## lm(formula = Return_10_fwd ~ MAPE, data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.116777 -0.029650  0.004347  0.028478  0.093157
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.1383475  0.0029889   46.29   <2e-16 ***
## MAPE        -0.0045885  0.0001727  -26.57   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04321 on 1482 degrees of freedom
## Multiple R-squared:  0.3226, Adjusted R-squared:  0.3221
## F-statistic: 705.8 on 1 and 1482 DF,  p-value: < 2.2e-16
```

```r
# 3rd part
k = 5
partition_index =  sample(1:5,nrow(d),replace = TRUE)
MSE_i = rep(NA, k)

for(i in 1:k) {
  train = d[partition_index!=i,]
  test = d[partition_index==i,]
  m1_cv = lm(Return_10_fwd ~ MAPE, data = train)
  pd = predict(m1_cv,newdata = test)
  MSE_i[i] = mean((pd - test$Return_10_fwd)^2)
}
mean(MSE_i)
```

```
## [1] 0.001866734
```

1. There are exactly 120 NAs because the new column “MAPE” is created from “Price” and “Earnings_10MA_back”, where “Earnings_10MA_back” has 120 NAs.
2. The coefficient is -0.0045885 and its standard error is 0.0001727. It’s significant because of its small p-value.
3. Clearly, the MSE of this model under five-fold CV is 0.00187.

Q2

```r
# 1st part
d["inverse_MAPE"] = 1/d$MAPE
m2 <- lm(Return_10_fwd ~ inverse_MAPE, data = d)
summary(m2)
```

```
##
## Call:
## lm(formula = Return_10_fwd ~ inverse_MAPE, data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.106298 -0.030839  0.002955  0.028179  0.103866
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.007659   0.002878  -2.661  0.00788 **
## inverse_MAPE  0.995904   0.036513  27.275  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04284 on 1482 degrees of freedom
## Multiple R-squared:  0.3342, Adjusted R-squared:  0.3338
## F-statistic: 743.9 on 1 and 1482 DF,  p-value: < 2.2e-16
```

```r
# 2nd part
k = 5
partition_index =  sample(1:5,nrow(d),replace = TRUE)
MSE_i = rep(NA, k)

for(i in 1:k) {
  train = d[partition_index!=i,]
  test = d[partition_index==i,]
  m2_cv = lm(Return_10_fwd ~ inverse_MAPE, data = train)
  pd = predict(m2_cv,newdata = test)
  MSE_i[i] = mean((pd - test$Return_10_fwd)^2)
}
mean(MSE_i)
```

```
## [1] 0.001839632
```

1. The coefficient is 0.99590 and its standard error is 0.03651. It's significant because of its small p-value.
2. Clearly, the MSE of this model under five-fold CV is 0.00184, which is less than the previous one.

Q3

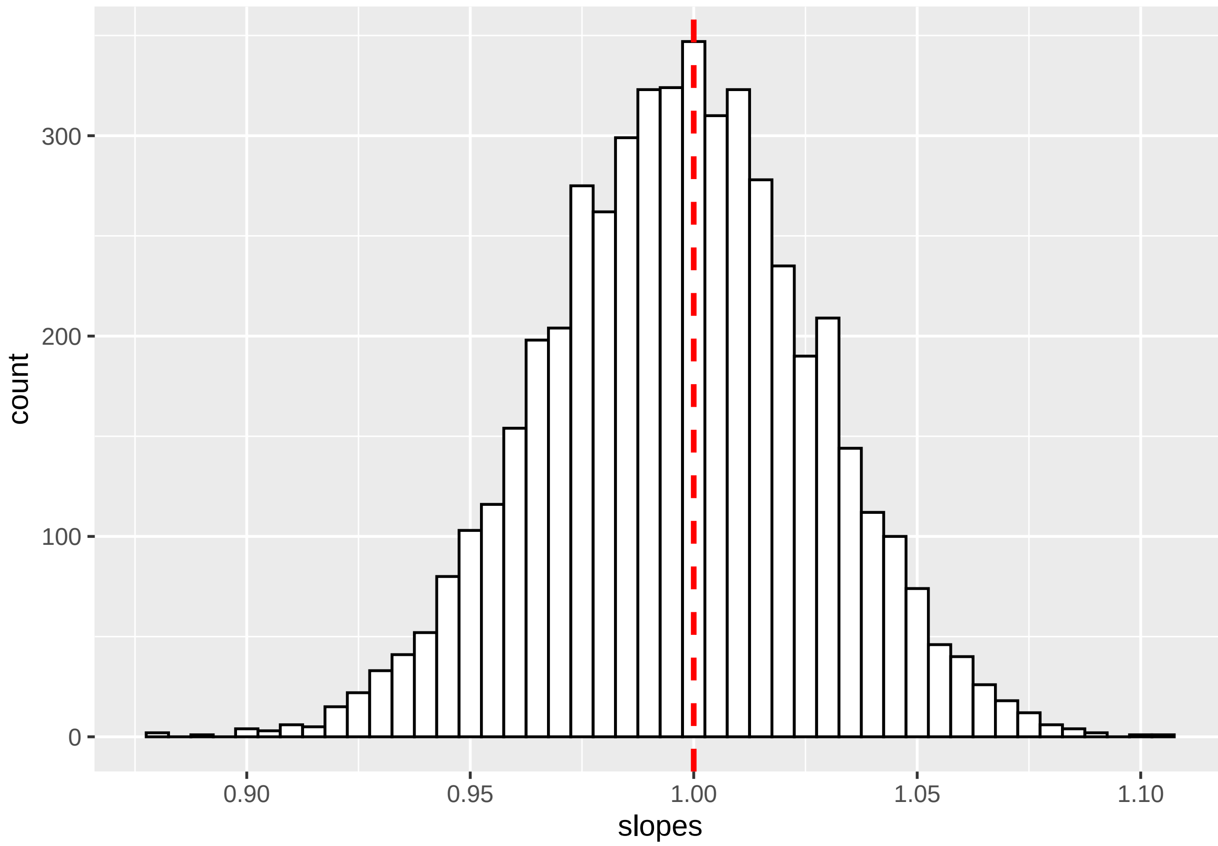
```r
# 1st part
training_MSE = mean((d$inverse_MAPE-d$Return_10_fwd)^2)
training_MSE
```

```
## [1] 0.001896346
```

1. The training MSE is 0.0019.
2. It illustrates that this simple model is a good model to approximate the return.

Q4

```r
# 1st part
library(ggplot2)
slopes = rep(NA,5000)
for (i in 1:5000){
  boot_ind = sample(1:nrow(d),size = nrow(d),replace = TRUE)
  d_boot = d[boot_ind,]
  slopes[i] = coef(lm(Return_10_fwd ~ inverse_MAPE,data=d_boot))[2]
}
ggplot(data.frame("slopes" = slopes), aes(x = slopes)) +
  geom_histogram(binwidth = .005,color = 'black',fill='white') +
  geom_vline(aes(xintercept=1), color="red", linetype="dashed", size=1)
```



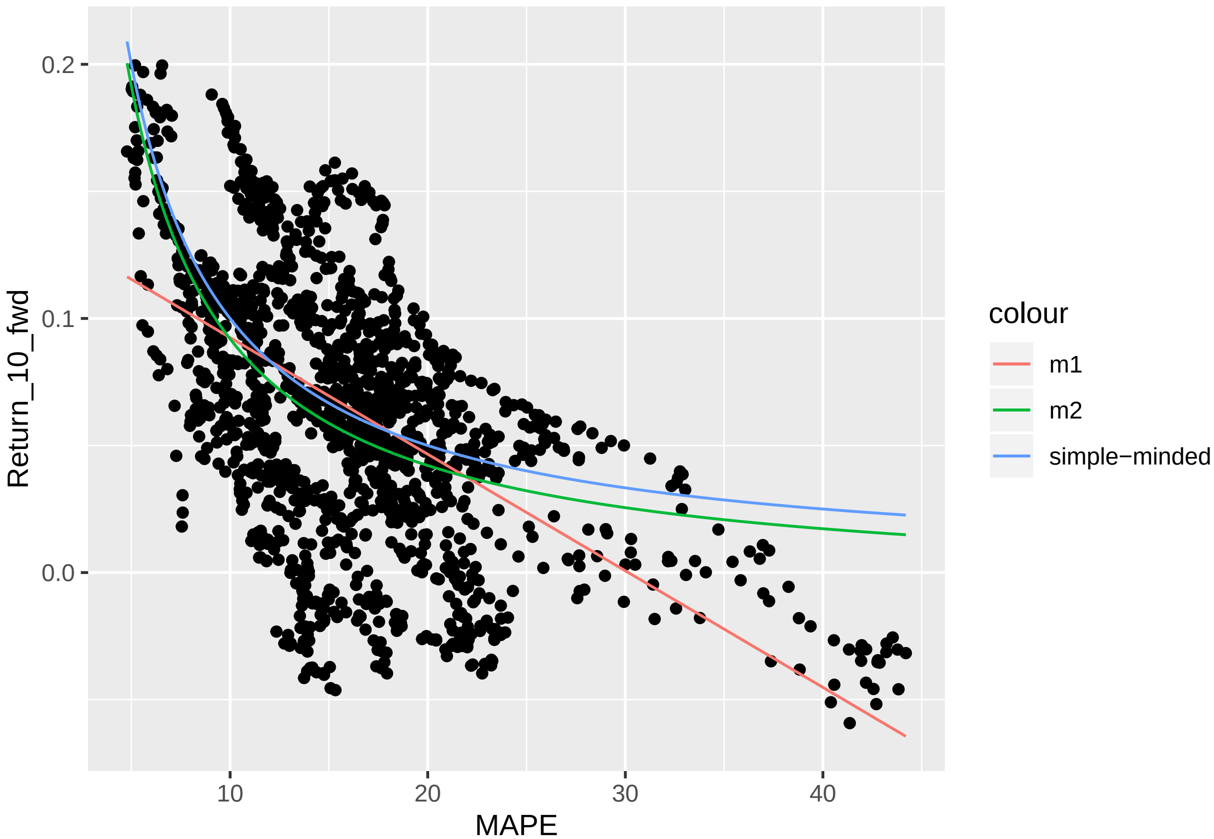

```r
# 2nd part
SE = sqrt(var(slopes))
up = mean(slopes)+ SE
down = mean(slopes) - SE

confint(m2)
```

```
##                    2.5 %       97.5 %
## (Intercept)  -0.01330433 -0.002013051
## inverse_MAPE  0.92428102  1.067526198
```

So the 95% confident interval by bootstrapping is (0.965, 1.03) and the result from the model in question 2 is (0.9243, 1.06753). It's worth noticing that they both centered at 0.996, but the later interval has a greater range, approximately twice of the first interval.

Q5

```r
d["pd1"] = predict(m1)
d["pd2"] = predict(m2)
ggplot(data = d, aes(x = MAPE, y = Return_10_fwd)) +
  geom_point() +
  geom_line(aes(x = MAPE, y = pd1, color = "m1")) +
  geom_line(aes(x = MAPE, y = pd2, color = "m2")) +
  geom_line(aes(x = MAPE, y = inverse_MAPE, color = "simple-minded"))
```



The big picture

1. I will use the second model m2 to make predictions. From the plot above, it shows that it performs well with lower MAPEs but badly on large MAPEs.
2. Yes, it's a plausible model. One obvious evidence is that the confidence interval of it is included in the m2 model as we illustrated above.

Exercises

Q4

We could use the bootstrap method to approximate it. We can bootstrap a large number of times B and fit B models with different bootstrap dataset. Then predict Y with each model and then we have a distribution of the predicted result, and thus can find the standard deviation of our prediction.

Q8

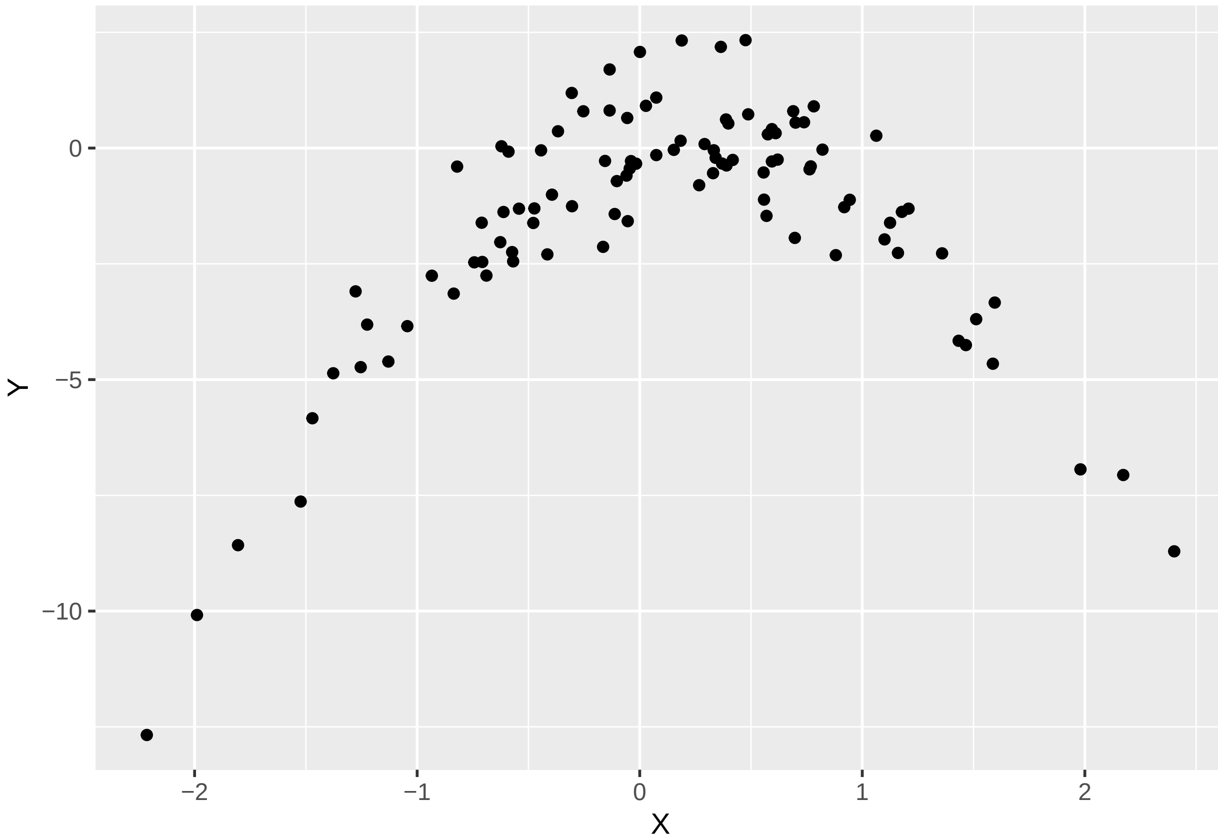
a.

```
set.seed(1)
x = rnorm(100)
y = x-2*x^2+rnorm(100)
```

$n = 100, p = 2$ $Y = X - 2X^2 + \epsilon$

b.

```r
d2 = data.frame("X" = x,"Y" = y)
ggplot(data = d2, aes(x = X, y = Y)) +
  geom_point()
```



It seems like the shape of this data is in quadratic form.

c.

```r
library(boot)
```

```
##
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':
##
##     melanoma
```

```r
set.seed(2)
cv.error = rep(0,4)
for(i in 1:4){
  glm.fit=glm(Y~poly(X,i) ,data=d2)
  cv.error[i] = cv.glm(d2,glm.fit)$delta[1]
}
cv.error
```

```
## [1] 7.2881616 0.9374236 0.9566218 0.9539049
```

d.

```r
library(boot)
set.seed(3)
```

```r
cv.error = rep(0,4)
for(i in 1:4){
  glm.fit=glm(Y~poly(X,i) ,data=d2)
  cv.error[i] = cv.glm(d2,glm.fit)$delta[1]
}
cv.error
```

```
## [1] 7.2881616 0.9374236 0.9566218 0.9539049
```

They are the same because LOOCV evaluates n folds of 1 observation, no randomness here.

e. The quadratic polynomial had the lowest LOOCV test error rate. This is what we expected.

f.

```r
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Y ~ poly(X, i), data = d2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0550  -0.6212  -0.1567   0.5952   2.2267
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.55002    0.09591 -16.162  < 2e-16 ***
## poly(X, i)1   6.18883    0.95905   6.453 4.59e-09 ***
## poly(X, i)2 -23.94830    0.95905 -24.971  < 2e-16 ***
## poly(X, i)3   0.26411    0.95905   0.275    0.784
## poly(X, i)4   1.25710    0.95905   1.311    0.193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9197797)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  87.379  on 95  degrees of freedom
## AIC: 282.3
##
## Number of Fisher Scoring iterations: 2
```

Clearly, polynomials of 3 and 4 are not significant which agrees with the result of CV.