Lab 6

Hands off my stack

Alice Chang

Inventing a variable

```
#load data
library(MASS)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)

d <- read.csv("https://bit.ly/36kibHZ")
#add column MAPE to dataframe
MAPE <- d$Price / d$Earnings_10MA_back
d$MAPE <- MAPE
summary(MAPE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   4.785  11.708  15.947  16.554  19.959  44.196     120
```

```
#Remove all rows with missing data
d <- na.omit(d)

#build linear model with MAPE
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
#Calculate MSE for five-fold CV
set.seed(42)
d <- d[-sample(1:nrow(d), 4), ]
k <- 5
partition_index <- rep(1:k, each = nrow(d)/k) %>%
sample()
MSE_i <- rep(NA, k)
d$partition_index <- partition_index
for(i in 1:k) {
  test <- d[d$partition_index == i, ]
  train <- d[d$partition_index != i,]
  lm1 <- lm(Return_10_fwd ~ MAPE, data = train)
  pd <- predict(lm1, newdata = test, type = "response")
  MSE_i[i] <- mean((test$Return_10_fwd-pd)^2)

  }


MSE = mean(MSE_i)
MSE
```

```
## [1] 0.001868484
```

1. There are exactly 120 NAs because "Earnings_10_MA_back" measures the ten-year moving average of earnings looking back from the current date. We do not have any earnings data for the previous ten years before 1871 that allows us to have information for the ten-year moving average for the first ten years. There is no data available for the ten-year moving average of earnings until 10 years later in 1871. Each row in the dataset represents one month of each of the 12 months in the year, and there are 10 years of mising data. Accordingly, 12 X 10 = 120, so there are exactly 120 NAs in both "Earnings_10MA_back" and "MAPE."
2. As evident in the linear model, the coefficient of MAPE is -0.004602 and its standard error is 0.000173 . This means that for every unit increase in "MAPE" (the ratio of "Price," the index of U.S. stocks, to "Earnings_10MA_back," the ten year moving average of earnings), the average rate of return over the next ten years from the current data, "Return_10_fwd," should decrease by 0.004602 units. Moreovr, this rate of return can vary by 0.000173. Since $P < 2e-16$, the effect of MAPE is statistically significant.
3. The MSE of the model under a five-fold CV is 0.001868484.

Inverting a variable

```r
#Create inverted variable
Inv_MAPE <- 1 / d$MAPE
d$Inv_MAPE <- Inv_MAPE



#Fit model with Inv_MAPE
```

```
m2 <- lm(Return_10_fwd ~ Inv_MAPE, data = d)
summary(m2)
```

```
##
## Call:
## lm(formula = Return_10_fwd ~ Inv_MAPE, data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.106266 -0.030816  0.002843  0.028174  0.103910
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.007815   0.002883  -2.711  0.00679 **
## Inv_MAPE     0.997627   0.036567  27.282  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04286 on 1478 degrees of freedom
## Multiple R-squared:  0.3349, Adjusted R-squared:  0.3345
## F-statistic: 744.3 on 1 and 1478 DF,  p-value: < 2.2e-16
```

```
#Find five-fold CV MSE
MSE_i <- rep(NA, k)
for(i in 1:k) {
  test <- d[d$partition_index == i, ]
  train <- d[d$partition_index != i,]
  lm2 <- lm(Return_10_fwd ~ Inv_MAPE, data = train)
  pd2 <- predict(lm2, newdata = test, type = "response")
  MSE_i[i] <- mean((test$Return_10_fwd-pd2)^2)

  }

MSE = mean(MSE_i)
MSE
```

```
## [1] 0.001836749
```

1. As evident in the linear model, the coefficient of Inv_MAPE is 0.997627 and its standard error is 0.036567 . This means that for every unit increase in "Inv_MAPE" (the ratio of "Earnings+10MA_back," the ten year moving average of earnings to "Price," the index of U.S. stocks), the average rate of return over the next ten years from the current data, "Return_10_fwd," should increase by 0.997627 units. Moreovr, this rate of return can vary by 0.036567. Since P < 2e-16, the effect of Inv_MAPE is statistically significant.
2. The CV MSE for this model is 0.001836749 which is slightly smaller than the CV MSE of the previous model.

A simple model

```
k <- 5
MSE_i <- rep(NA, k)

for(i in 1:k) {
  test <- d[d$partition_index == i, ]
```

```r
  train <- d[d$partition_index != i,]
  MSE_i[i] <- mean((train$Return_10_fwd-train$Inv_MAPE)^2)
}

MSE = mean(MSE_i)
MSE
```

```
## [1] 0.001898626
```

```r
MSE_i <- rep(NA, k)

for(i in 1:k) {
  test <- d[d$partition_index == i, ]
  train <- d[d$partition_index != i,]
  MSE_i[i] <- mean((test$Return_10_fwd-test$Inv_MAPE)^2)

  }

MSE = mean(MSE_i)
MSE
```

```
## [1] 0.001898626
```

1. The training MSE for this model is 0.001898626. This is exactly the same as the CV test MSE for this model.

2. In most cases, we train our model on the training data and fit the derived model onto our test data afterwards, so that our training MSE may be lower than our test MSE at times. However, the training MSE and test MSE for this simple model are equivalent because we are not simply fitting a model derived from our training data onto the test data. Instead, in both the test and training data, we are evaluating the suggestion that 1/MAPE is equal to the average rate of return over the next ten years. Consequently, we would expect this MSE to be the same across the entire dataset.

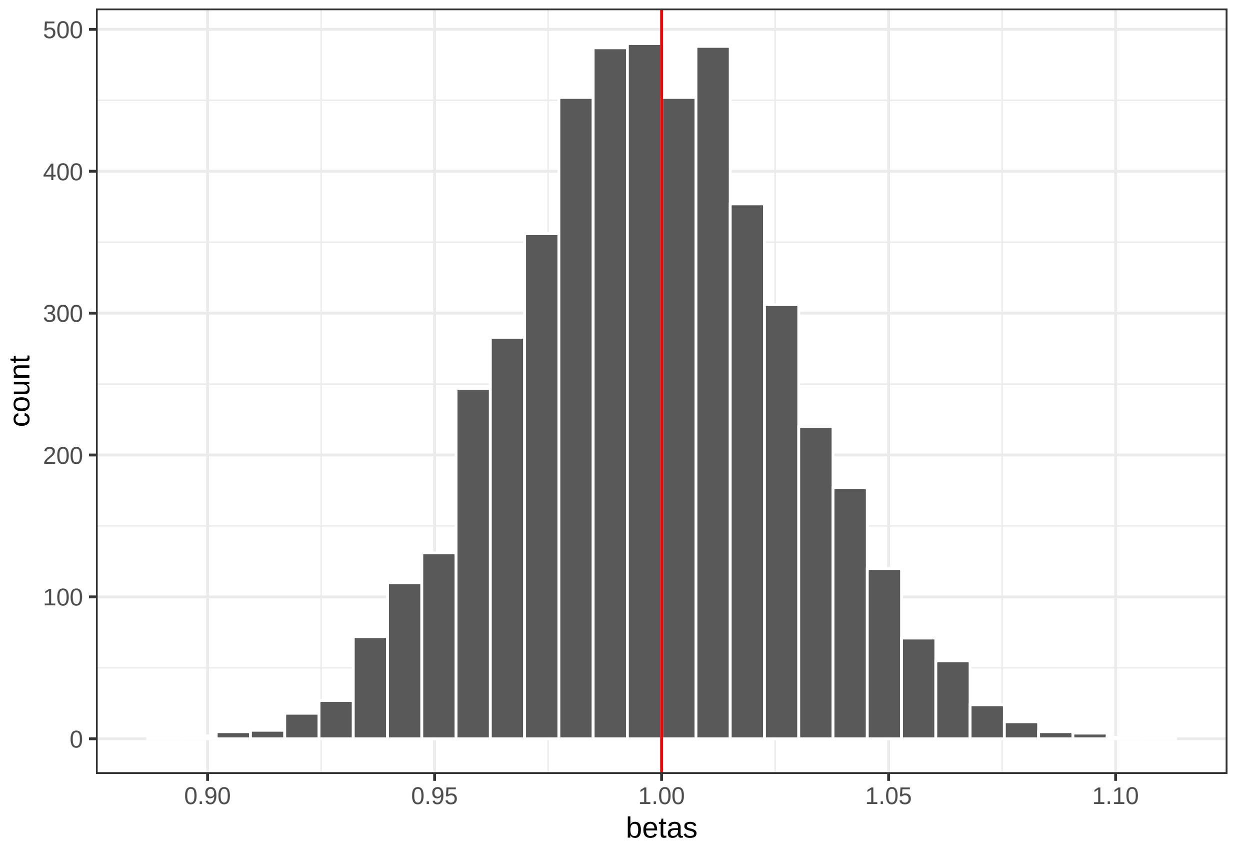
Is simple sufficient?

```r
betas <- rep(NA, 5000)
for(i in 1:5000) {
  boot_ind <- sample(1:nrow(d),
                     size = nrow(d),
                     replace = TRUE)
  d_boot <- d[boot_ind, ]
  betas[i] <- coef(lm(Return_10_fwd ~ Inv_MAPE, data = d_boot))[2]
}


#Graoh results
df <- data.frame(betas)

ggplot(df, aes(betas)) +
  geom_histogram(col = "white") +  theme_bw() + geom_vline(xintercept= 1, color="red")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```r
#Obtain 95% bootstrap confidence interval
m_boot = lm(Return_10_fwd ~ Inv_MAPE, data = d_boot)
summary(m_boot)
```

```
##
## Call:
## lm(formula = Return_10_fwd ~ Inv_MAPE, data = d_boot)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.106281 -0.031423  0.003522  0.030028  0.100153
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.005818   0.002854  -2.039   0.0417 *
## Inv_MAPE     0.970366   0.035948  26.993   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04278 on 1478 degrees of freedom
## Multiple R-squared:  0.3302, Adjusted R-squared:  0.3298
## F-statistic: 728.6 on 1 and 1478 DF,  p-value: < 2.2e-16
```

```r
confint(m_boot, level = 0.95)
```

```
##                  2.5 %        97.5 %
## (Intercept) -0.0114152 -0.0002200273
## Inv_MAPE     0.8998514  1.0408814627
```

```r
confint(m2, level = 0.95)
```

```
##                   2.5 %       97.5 %
## (Intercept) -0.01347076 -0.002160041
## Inv_MAPE     0.92589823  1.069356445
```

The 95% bootstrap confidence interval for the slope of 1/MAPE is between 0.96444846 and 1.106315186. In comparision, through directly running confint() on the function in questino 2, we are 95% certain that the slope for 1/MAPE falls between 0.92589823 and 1.069356445. The lower limit of the 95% bootstrap confidence interval is only slightly higher than the interval obtained through the second method. However, the upper limit of the 95% bootstrap confidence interval is very slightly lower than the interval obtained through the second method. Consequently the confidence intervals for the slope of 1/MAPE are extremely similar.

```r
summary(m2)
```

```
##
## Call:
## lm(formula = Return_10_fwd ~ Inv_MAPE, data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.106266 -0.030816  0.002843  0.028174  0.103910
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.007815   0.002883  -2.711  0.00679 **
## Inv_MAPE     0.997627   0.036567  27.282  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04286 on 1478 degrees of freedom
## Multiple R-squared:  0.3349, Adjusted R-squared:  0.3345
## F-statistic: 744.3 on 1 and 1478 DF,  p-value: < 2.2e-16
```

```r
summary(m1)
```

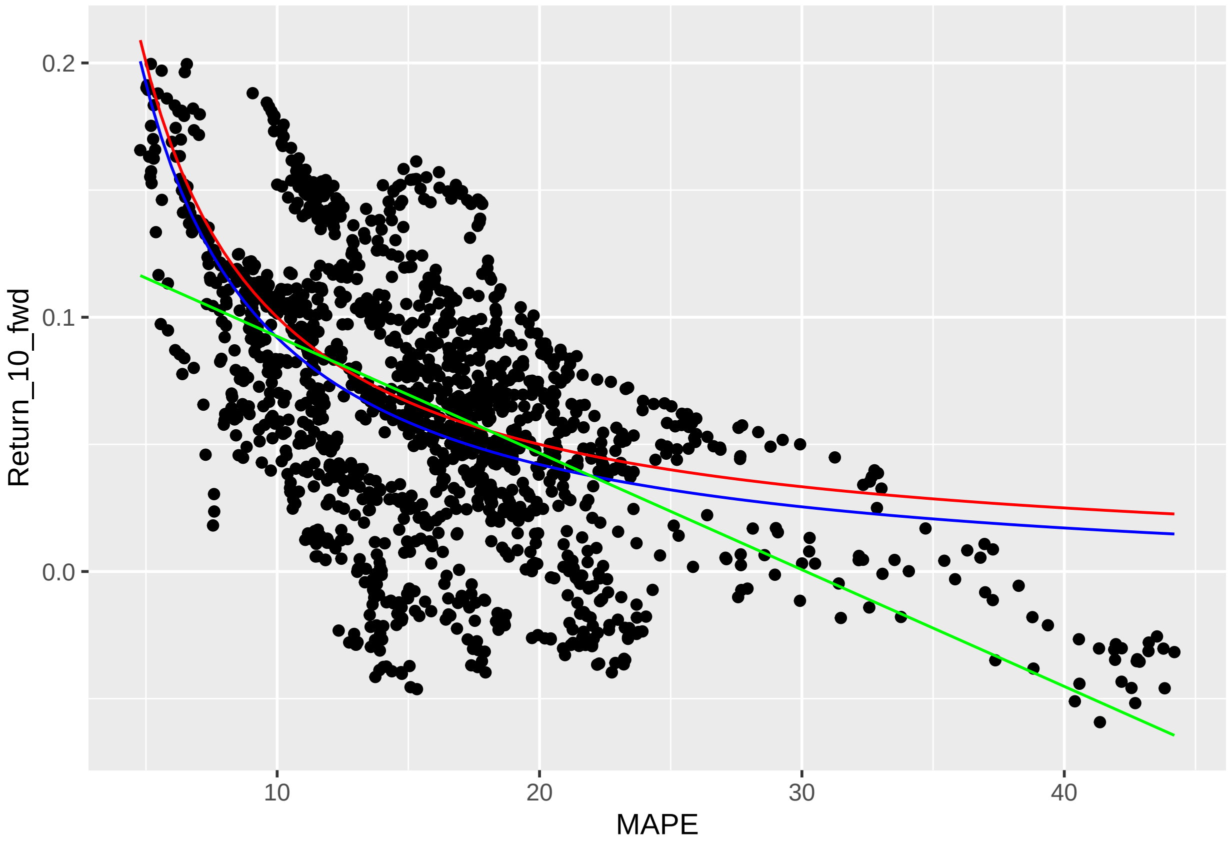
```
##
## Call:
## lm(formula = Return_10_fwd ~ MAPE, data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.116777 -0.029650  0.004347  0.028478  0.093157
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.1383475  0.0029889   46.29   <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
simple_mod <- function(MAPE) {
  1/MAPE
}

second_mod <- function(MAPE) {
  -0.007815 +  (0.997627 * 1/MAPE)
}

first_mod <- function(MAPE) {
    0.1383475 + (-0.0045885 * MAPE)
  }

ggplot(d, aes(x = MAPE, y = Return_10_fwd)) + geom_point() + stat_function(fun = simple_mod, color = "re
```



All three models capture the mostly negatively correlated relationship between MAPE and the average rate of return over the next 10 yeras of the original dataset. However, Model 2 and the simple model capture a slighltly curvilinear relationship betwee MAPE and the rate of return. Comparatively, Model 1 represents a linear relationship between MAPE and the rate of return. As MAPE increases, the average rate of return decrease consistently.

The big picture

```r
mean(betas)
```

```
## [1] 0.997454
```

1. Since Model 2 has the lowest CV MSE of 0.0018769, I would select the model to make predictions on returns. Looking at the plot from question 5, Model 2 is able to capture the slighty curvilinear

relationship between MAPE and the average rate of returns, subsequently seeming to be a pretty good model. Furthermore, in contrast to Model 1, Model 2 better estimates the higher average rate of returns when MAPE (the ration between "Price" and "Earnings") is small. However, Model 2 does not model the slight curvature of the data accurately, subsequently tending to overestimate the average rate of returns when MAPE is larger.

2.Based off my bootstrapping procedure for the slope of the linear model using 1/MAPE as a predictor, the simple-minded model is very plausible given the data. It follows from the suggestion that the average rate of return over the next 10 years is equal to 1/MAPE that the slope of the simple-minded model is 1. This is very close to the mean of the slope estimates determined from the bootstrap distribution of slope for 1/MAPE in Model 2 (0.9980164).

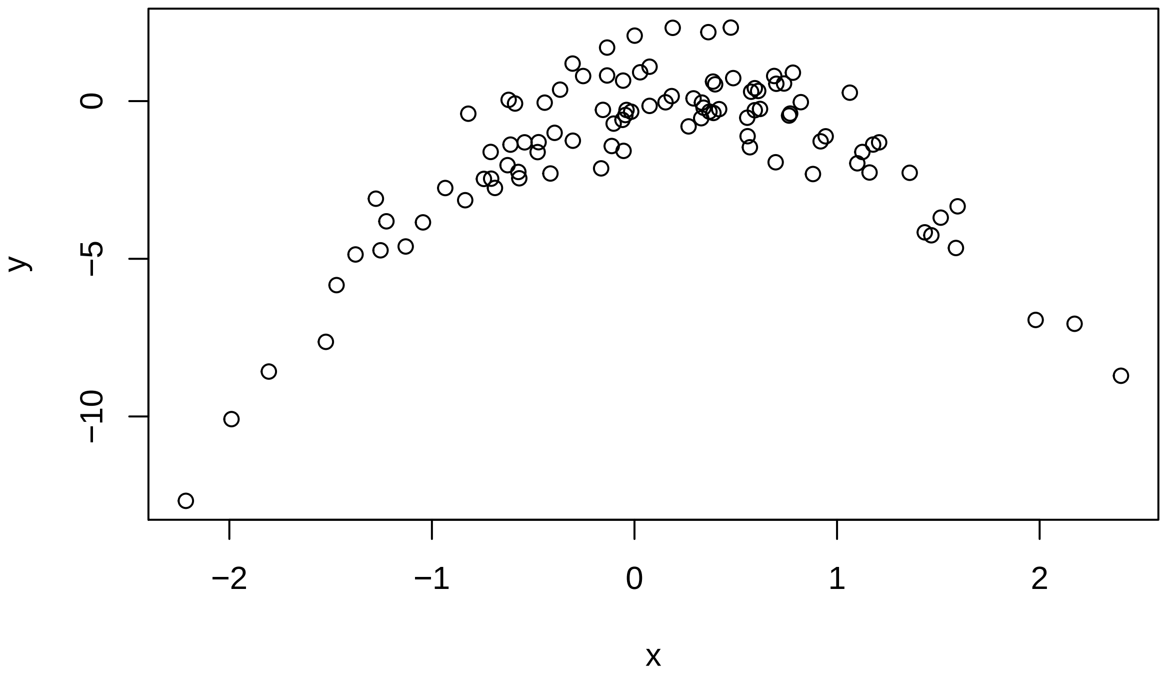
Chapter 5 exercises

4. The standard deviation of the predicted observation can be estimatted through the bootstrap approach. Correspondingly, we would generate many random samples from our original dataset, sampling for replacement each time, and record the estimates for the standard deviation each time. Afterwards, we would find the mean of the estimates from each time and the standard deviation across all estimates.

5.

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

plot(x,y)
```



```
#c
library(boot)
set.seed(2)
Data <- data.frame(x, y)
fit1 <- glm(y ~ x)
cv.glm(Data, fit1)$delta[1]
```

```
## [1] 7.288162
```

```r
fit2 <- glm(y ~ poly(x, 2))
cv.glm(Data, fit2)$delta[1]
```

```
## [1] 0.9374236
```

```r
fit3 <- glm(y ~ poly(x, 3))
cv.glm(Data, fit3)$delta[1]
```

```
## [1] 0.9566218
```

```r
fit4 <- glm(y ~ poly(x, 4))
cv.glm(Data, fit4)$delta[1]
```

```
## [1] 0.9539049
```

```r
set.seed(19)
fit5 <- glm(y ~ x)
cv.glm(Data, fit5)$delta[1]
```

```
## [1] 7.288162
```

```r
fit6 <- glm(y ~ poly(x, 2))
cv.glm(Data, fit6)$delta[1]
```

```
## [1] 0.9374236
```

```r
fit7 <- glm(y ~ poly(x, 3))
cv.glm(Data, fit7)$delta[1]
```

```
## [1] 0.9566218
```

```r
fit8 <- glm(y ~ poly(x, 4))
cv.glm(Data, fit8)$delta[1]
```

```
## [1] 0.9539049
```

```r
summary(fit1)
```

```
##
## Call:
## glm(formula = y ~ x)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -9.5161  -0.6800   0.6812   1.5491   3.8183
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.6254     0.2619  -6.205 1.31e-08 ***
## x             0.6925     0.2909   2.380   0.0192 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 6.760719)
##
##     Null deviance: 700.85  on 99  degrees of freedom
## Residual deviance: 662.55  on 98  degrees of freedom
## AIC: 478.88
##
```

```
## Number of Fisher Scoring iterations: 2
```

```r
summary(fit2)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 2))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9650  -0.6254  -0.1288   0.5803   2.2700
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5500     0.0958  -16.18  < 2e-16 ***
## poly(x, 2)1   6.1888     0.9580    6.46 4.18e-09 ***
## poly(x, 2)2 -23.9483     0.9580  -25.00  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9178258)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  89.029  on 97  degrees of freedom
## AIC: 280.17
##
## Number of Fisher Scoring iterations: 2
```

```r
summary(fit3)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 3))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9765  -0.6302  -0.1227   0.5545   2.2843
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.55002    0.09626 -16.102  < 2e-16 ***
## poly(x, 3)1   6.18883    0.96263   6.429 4.97e-09 ***
## poly(x, 3)2 -23.94830    0.96263 -24.878  < 2e-16 ***
## poly(x, 3)3   0.26411    0.96263   0.274    0.784
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9266599)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  88.959  on 96  degrees of freedom
## AIC: 282.09
##
## Number of Fisher Scoring iterations: 2
```

```r
summary(fit4)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## 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, 4)1   6.18883    0.95905   6.453 4.59e-09 ***
## poly(x, 4)2 -23.94830    0.95905 -24.971  < 2e-16 ***
## poly(x, 4)3   0.26411    0.95905   0.275    0.784
## poly(x, 4)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
```

n = 100 and p = 2 Y = X - $2X^2$ + ϵ

b) There is a curved relationship between x and y. Initially, there is a positively correlated relationship between x and y as y increases as x increases. However, once x is greater than 0, there is a negatively correlated relationship between x and y as an increase in x is assocaited with a decrease in y.

c) Compute LOOCV errors:

i. 7.288162
ii. 0.9374236 iii.0.9566218
iii. 0.9539049

d) Compute LOOCV errors:

i. 7.288162
ii. 0.9374236 iii.0.9566218
iii. 0.9539049

The results are exactly the same as the results from (c). This is because LOOCV uses a single observation as a validation set.

e) "Fit2" had the smallest LOOCV error. This is not surprising because the scatterplot from b demonstrates that the relationship between x and y is quadratic.

f) After fitting each of the models, it is clear that the linear and quadratic terms are consistently statistically significants. However, the cubic and 4th degree terms are not. Consequently, the results agree with our cross-validation results in which the quadratic ("fit2") model had the least LOOCV error.