BDAR Coursework 2

JP Sainsbury

4 November 2017

JP Sainsbury ID Number - 13139347

## R Markdown

# 1

## (a)

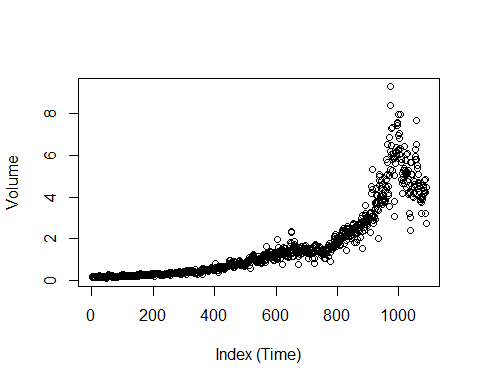
Logistic Regression Using weekly data set produce some numerical and graphical summaries, any patterns?

library(ISLR)  
cor(Weekly[,-9])

## Year Lag1 Lag2 Lag3 Lag4  
## Year 1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923  
## Lag1 -0.03228927 1.000000000 -0.07485305 0.05863568 -0.071273876  
## Lag2 -0.03339001 -0.074853051 1.00000000 -0.07572091 0.058381535  
## Lag3 -0.03000649 0.058635682 -0.07572091 1.00000000 -0.075395865  
## Lag4 -0.03112792 -0.071273876 0.05838153 -0.07539587 1.000000000  
## Lag5 -0.03051910 -0.008183096 -0.07249948 0.06065717 -0.075675027  
## Volume 0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617  
## Today -0.03245989 -0.075031842 0.05916672 -0.07124364 -0.007825873  
## Lag5 Volume Today  
## Year -0.030519101 0.84194162 -0.032459894  
## Lag1 -0.008183096 -0.06495131 -0.075031842  
## Lag2 -0.072499482 -0.08551314 0.059166717  
## Lag3 0.060657175 -0.06928771 -0.071243639  
## Lag4 -0.075675027 -0.06107462 -0.007825873  
## Lag5 1.000000000 -0.05851741 0.011012698  
## Volume -0.058517414 1.00000000 -0.033077783  
## Today 0.011012698 -0.03307778 1.000000000

Little correlation between this weeks return and the previous weeks return. The correlation between Year and Volume are substantial.

plot (Weekly$Volume,xlab = "Index (Time)", ylab = "Volume")



The above graphic shows a strong relationship, as time passes the volume increases exponentially.

## (b)

Perform logistic regression with Direction

glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data=Weekly,family = binomial)  
summary(glm.fit)

##   
## Call:  
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +   
## Volume, family = binomial, data = Weekly)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6949 -1.2565 0.9913 1.0849 1.4579   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.26686 0.08593 3.106 0.0019 \*\*  
## Lag1 -0.04127 0.02641 -1.563 0.1181   
## Lag2 0.05844 0.02686 2.175 0.0296 \*   
## Lag3 -0.01606 0.02666 -0.602 0.5469   
## Lag4 -0.02779 0.02646 -1.050 0.2937   
## Lag5 -0.01447 0.02638 -0.549 0.5833   
## Volume -0.02274 0.03690 -0.616 0.5377   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1496.2 on 1088 degrees of freedom  
## Residual deviance: 1486.4 on 1082 degrees of freedom  
## AIC: 1500.4  
##   
## Number of Fisher Scoring iterations: 4

The smallest p-value:Lag2, its positive coefficient suggests if the market had a positive return last week its likely to go up this week.

## (c)

Compute the confusion matrix and overall fraction of correct predictions, what is the confusion matrix telling me about the type of mistakes made by logistic regression.

glm.probs=predict(glm.fit,type="response")  
glm.pred=rep("Down",1089)  
Direction = glm.pred[glm.probs>.5]="Up"  
table(glm.pred,Weekly$Direction)

##   
## glm.pred Down Up  
## Down 54 48  
## Up 430 557

(8+538)/1089

## [1] 0.5013774

mean(glm.pred==Weekly$Direction)

## [1] 0.5610652

LR correctly predicted the movement of the market 50.14% of the time

What is the confusion matrix telling me about the types of mistakes made by logistic regression. It incorrectly predicted the False negative 476 times which is more than the False positive 67.

# 2

LOGISTIC REGRESSION - Develop a model to predict if a car gets high or low milage based on the Auto data set.

## (a)

mpg01 = rep(0,392)  
mpg01[Auto$mpg>median(Auto$mpg)]=1  
table(mpg01)

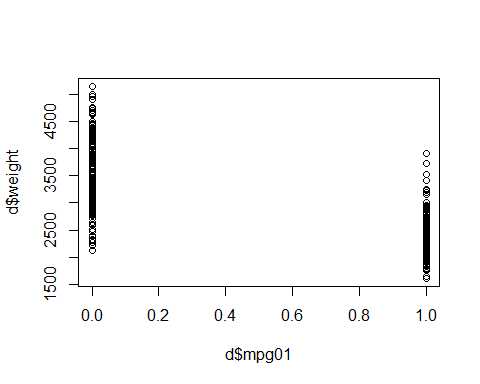
## mpg01  
## 0 1   
## 196 196

d =data.frame(Auto, mpg01)

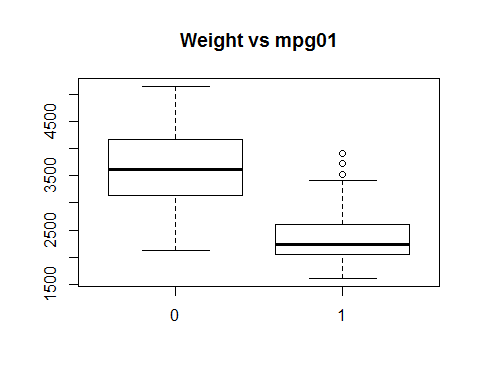
## (b)

Explore the data graphically to investigate the association between mpg01 and the other features

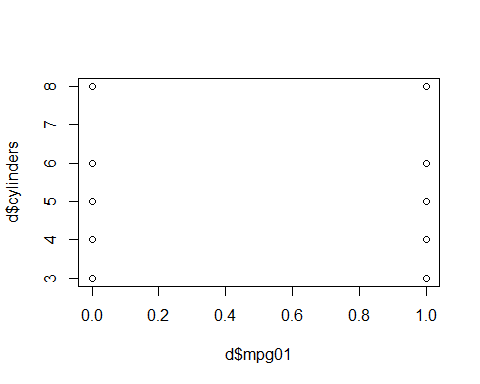
plot(d$mpg01,d$weight) #Yes useful to predict mpg01



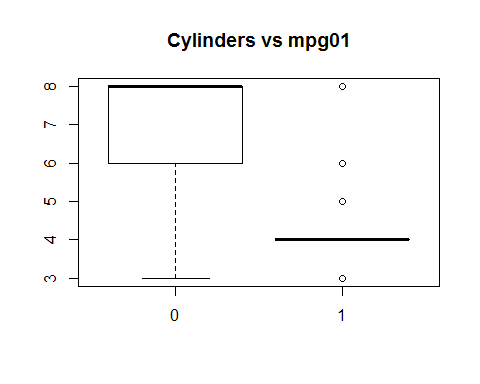
boxplot(d$weight ~ d$mpg01, main = "Weight vs mpg01")



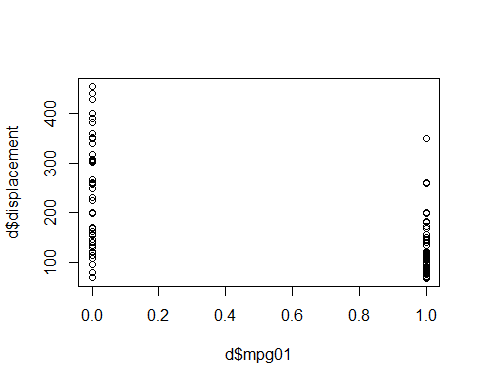
##The above shows the heavier the car the more miles per gallon, possibiliy bigger fuel tank  
plot(d$mpg01,d$cylinders) #Not useful



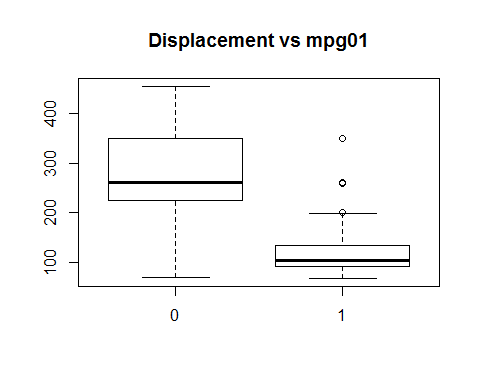
boxplot(d$cylinders ~ d$mpg01, main = "Cylinders vs mpg01")



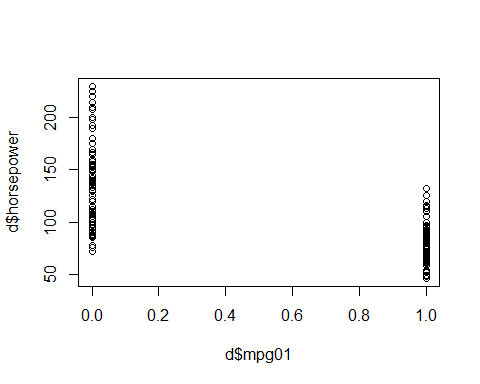
plot(d$mpg01,d$displacement) #Yes useful to predict mpg01



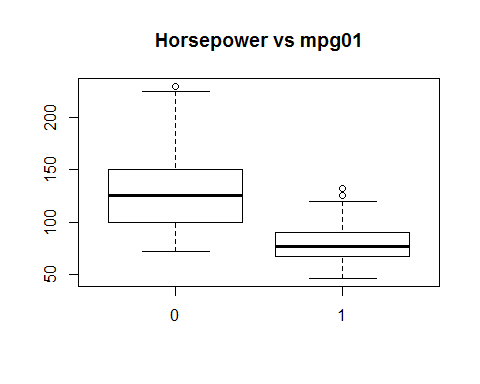
boxplot(d$displacement ~ d$mpg01, main = "Displacement vs mpg01")



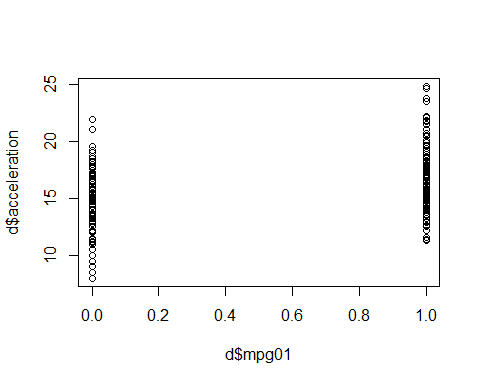
plot(d$mpg01,d$horsepower) #Yes useful to predict mpg01



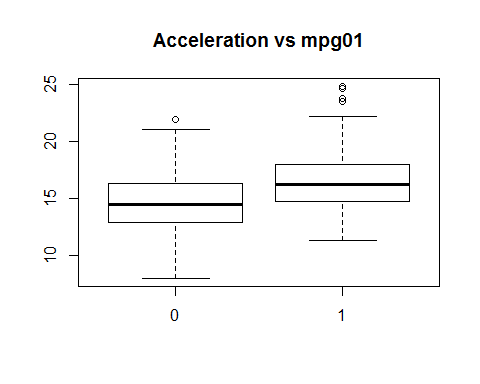
boxplot(d$horsepower ~ d$mpg01, main = "Horsepower vs mpg01")



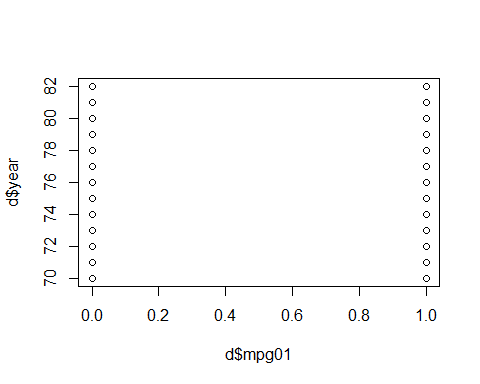
plot(d$mpg01,d$acceleration) #Yes useful to predict mpg01



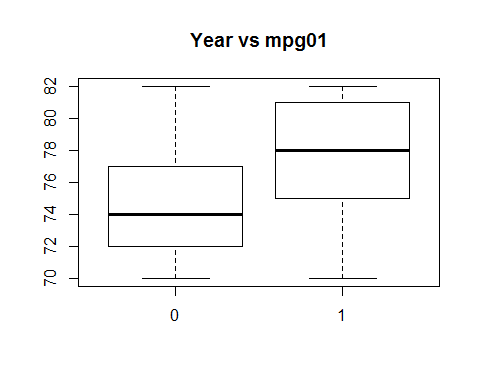
boxplot(d$acceleration ~ d$mpg01, main = "Acceleration vs mpg01")



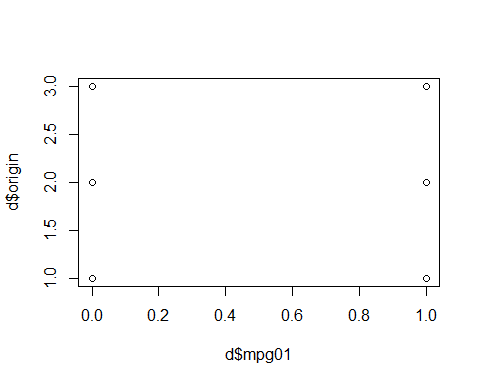
plot(d$mpg01,d$year) #Not useful



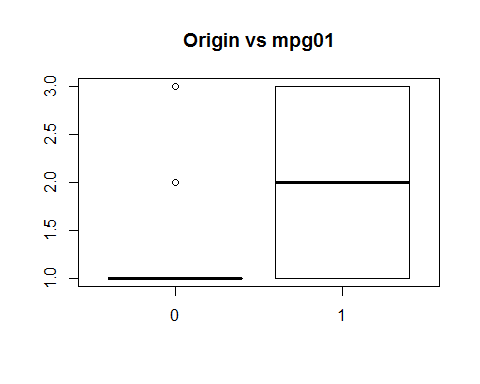
boxplot(d$year ~ d$mpg01, main = "Year vs mpg01")



plot(d$mpg01,d$origin) #Not useful



boxplot(d$origin ~ d$mpg01, main = "Origin vs mpg01")



# 3

## (a)

Fit a logistic regression model

set.seed(1)  
  
glm.fit=glm(default~income+balance, data = Default, family = binomial)  
  
glm.fit

##   
## Call: glm(formula = default ~ income + balance, family = binomial,   
## data = Default)  
##   
## Coefficients:  
## (Intercept) income balance   
## -1.154e+01 2.081e-05 5.647e-03   
##   
## Degrees of Freedom: 9999 Total (i.e. Null); 9997 Residual  
## Null Deviance: 2921   
## Residual Deviance: 1579 AIC: 1585

## (b)

Split the data set

set.seed(1)  
train = sample(1:nrow(Default), nrow(Default)/2)  
default.validation = Default[-train,]  
  
##(ii)  
  
glm.fit=glm(default~income+balance, data = Default, family = binomial, subset=train)  
  
#(iii)  
glm.probs=predict(glm.fit, default.validation, type="response")  
  
glm.pred=rep("No",5000)  
glm.pred[glm.probs>.5]="Yes"  
  
##(iv)  
  
table(glm.pred, default.validation$default)

##   
## glm.pred No Yes  
## No 4805 115  
## Yes 28 52

mean(glm.pred==default.validation)

## [1] 0.41845

mean(glm.pred!=default.validation)

## [1] 0.58155

58.16% are misclassified.

## (c)

Repeat with different splits - Repeat 1

set.seed(1)  
train = sample(1:nrow(Default), nrow(Default)/1.5)  
default.validation = Default[-train,]  
  
##(ii)  
  
glm.fit=glm(default~income+balance, data = Default, family = binomial, subset=train)  
  
#(iii)  
glm.probs=predict(glm.fit, default.validation, type="response")  
  
glm.pred=rep("No",3334)  
glm.pred[glm.probs>.5]="Yes"  
  
##(iv)  
  
table(glm.pred, default.validation$default)

##   
## glm.pred No Yes  
## No 3211 79  
## Yes 14 30

mean(glm.pred==default.validation)

## [1] 0.4183413

mean(glm.pred!=default.validation)

## [1] 0.5816587

### Repeat 2

set.seed(1)  
train = sample(1:nrow(Default), nrow(Default)/1.01)  
default.validation = Default[-train,]  
  
##(ii)  
  
glm.fit=glm(default~income+balance, data = Default, family = binomial, subset=train)  
  
#(iii)  
glm.probs=predict(glm.fit, default.validation, type="response")  
  
glm.pred=rep("No",100)  
glm.pred[glm.probs>.5]="Yes"  
  
##(iv)  
  
table(glm.pred, default.validation$default)

##   
## glm.pred No Yes  
## No 99 1

mean(glm.pred==default.validation)

## [1] 0.4375

mean(glm.pred!=default.validation)

## [1] 0.5625

### Repeat 3

set.seed(1)  
train = sample(1:nrow(Default), nrow(Default)/4)  
default.validation = Default[-train,]  
  
##(ii)  
  
glm.fit=glm(default~income+balance, data = Default, family = binomial, subset=train)  
  
#(iii)  
glm.probs=predict(glm.fit, default.validation, type="response")  
  
glm.pred=rep("No",7500)  
glm.pred[glm.probs>.5]="Yes"  
  
##(iv)  
  
table(glm.pred, default.validation$default)

##   
## glm.pred No Yes  
## No 7198 155  
## Yes 53 94

mean(glm.pred==default.validation)

## [1] 0.4188

mean(glm.pred!=default.validation)

## [1] 0.5812

The result show that the sample size changes do not affect the ability to correctly predict the validation/ test set results.

## (d)

set.seed(1)  
train = sample(1:nrow(Default), nrow(Default)/2)  
default.validation = Default[-train,]  
  
##(ii)  
  
glm.fit=glm(default~income+balance+student, data = Default, family = binomial, subset=train)  
  
#(iii)  
glm.probs=predict(glm.fit, default.validation, type="response")  
  
glm.pred=rep("No",5000)  
glm.pred[glm.probs>.5]="Yes"  
  
##(iv)  
  
table(glm.pred, default.validation$default)

##   
## glm.pred No Yes  
## No 4803 114  
## Yes 30 53

mean(glm.pred==default.validation)

## [1] 0.41815

mean(glm.pred!=default.validation)

## [1] 0.58185

Including the dummy variable student does not help reduce the error rate.

# 4

## (a)

LOOCV and Loop On the weekly data set fit a logistic regression mode that predicts Direction using Lag1 and Lag2

library(ISLR)  
glm.fit=glm(Direction~Lag1+Lag2, data=Weekly, family=binomial)  
summary(glm.fit)

##   
## Call:  
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.623 -1.261 1.001 1.083 1.506   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.22122 0.06147 3.599 0.000319 \*\*\*  
## Lag1 -0.03872 0.02622 -1.477 0.139672   
## Lag2 0.06025 0.02655 2.270 0.023232 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1496.2 on 1088 degrees of freedom  
## Residual deviance: 1488.2 on 1086 degrees of freedom  
## AIC: 1494.2  
##   
## Number of Fisher Scoring iterations: 4

On the Weekly data set fit logistic regression model that predicts Direction using Lag1 and Lag2 using all but the first observation.

## (b)

Weekly2 <- Weekly[-c(1),]  
glm.fit=glm(Direction~Lag1+Lag2, data=Weekly2, family=binomial)  
summary(glm.fit)

##   
## Call:  
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly2)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6258 -1.2617 0.9999 1.0819 1.5071   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.22324 0.06150 3.630 0.000283 \*\*\*  
## Lag1 -0.03843 0.02622 -1.466 0.142683   
## Lag2 0.06085 0.02656 2.291 0.021971 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1494.6 on 1087 degrees of freedom  
## Residual deviance: 1486.5 on 1085 degrees of freedom  
## AIC: 1492.5  
##   
## Number of Fisher Scoring iterations: 4

Predict the direction of the first observation

## (c)

glm.probs = predict(glm.fit, Weekly[1,], type = "response")  
glm.probs

## 1   
## 0.5713923

Weekly[1,]

## Year Lag1 Lag2 Lag3 Lag4 Lag5 Volume Today Direction  
## 1 1990 0.816 1.572 -3.936 -0.229 -3.484 0.154976 -0.27 Down

Our model predicts the direct as up but the actual direction was down, incorrect classification.

## (d)

Write a for loop from i = 1 to i = n where n is the number of observations in the data set.

n = nrow(Weekly)  
  
count = rep(0,n)  
  
for (i in 1:n) {  
 glm.fit = glm(Direction ~ Lag1 + Lag2, data = Weekly[-i,], family = binomial)  
 prediction = predict.glm(glm.fit, Weekly[i,], type = "response") > 0.5  
 actual = Weekly[i, ]$Direction == "Up"  
 if (prediction != actual)  
 count[i] = 1}  
  
sum(count)

## [1] 490

## (e)

mean(count)

## [1] 0.4499541

The LOOCV estimate for the test error is 45%.

# 5

## (a)

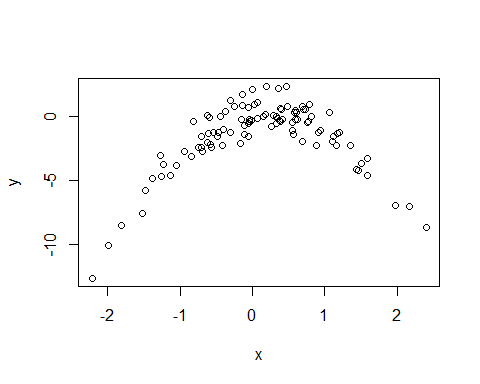
set.seed(1)  
x = rnorm(100)  
y=x-2\*x^2+rnorm(100)

n is eqaul to 100 and p is eqaul to 2.

## (b)

Create a scatter plot

plot(y~x)



As x approaches zero y also approaches zero, the plot shows a parabola shap plot open at the bottom

## (c)

Set a random seed, and then compute the LOOCV errors that result from fitting the following four models using least squares.

library(boot)  
set.seed(1)  
randomdata = data.frame(x, y)  
glm.fit=glm(y~x ,data=randomdata)  
cv.err =cv.glm(randomdata ,glm.fit)  
cv.err$delta

## [1] 7.288162 7.284744

glm.fit=glm(y~poly(x,2), data=randomdata)  
cv.err=cv.glm (randomdata ,glm.fit)  
cv.err$delta

## [1] 0.9374236 0.9371789

glm.fit=glm(y~poly(x,3), data=randomdata)  
cv.err=cv.glm (randomdata ,glm.fit)  
cv.err$delta

## [1] 0.9566218 0.9562538

glm.fit=glm(y~poly(x,4), data=randomdata)  
cv.err=cv.glm (randomdata ,glm.fit)  
cv.err$delta

## [1] 0.9539049 0.9534453

## (d)

set.seed(2)  
XY = data.frame(x, y)  
glm.fit=glm(y~x ,data=randomdata)  
cv.err =cv.glm(randomdata ,glm.fit)  
cv.err$delta

## [1] 7.288162 7.284744

glm.fit=glm(y~poly(x,2), data=randomdata)  
cv.err=cv.glm (randomdata ,glm.fit)  
cv.err$delta

## [1] 0.9374236 0.9371789

glm.fit=glm(y~poly(x,3), data=randomdata)  
cv.err=cv.glm (randomdata ,glm.fit)  
cv.err$delta

## [1] 0.9566218 0.9562538

glm.fit=glm(y~poly(x,4), data=randomdata)  
cv.err=cv.glm (randomdata ,glm.fit)  
cv.err$delta

## [1] 0.9539049 0.9534453

My results are the same, I was expecting different two numbers in the delta vector contain the cross-validation results as the data set is different. There is a drop in the estimated test MSE between the linear and quadratic plots, no improvement with using higher-order polynomials.

## (e)

A model that predicts Y using a quadratic function of X performs better than a model that involves only a linear function of x, and there is little evidence in favor of the model that uses a cubic function of x or quartic.

## (f)

summary(glm(y~x ,data=randomdata))

##   
## Call:  
## glm(formula = y ~ x, data = randomdata)  
##   
## 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

summary(glm(y~poly(x,2), data=randomdata))

##   
## Call:  
## glm(formula = y ~ poly(x, 2), data = randomdata)  
##   
## 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

summary(glm(y~poly(x,3), data=randomdata))

##   
## Call:  
## glm(formula = y ~ poly(x, 3), data = randomdata)  
##   
## 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

summary(glm(y~poly(x,4), data=randomdata))

##   
## Call:  
## glm(formula = y ~ poly(x, 4), data = randomdata)  
##   
## 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

The coefficient estimates support the cross validation conculsions.