M6 applied

5(a)

In [28]:
```
library(ISLR)
```

In [29]:
```
data(Default)
```

In [30]:
```
set.seed(1)
```

In [58]:
```
fit1 = glm(default~income + balance, data=Default,
           family=binomial)
```

In [32]:
```
summary(fit1)
```

```
Call:
glm(formula = default ~ income + balance, family = binomial,
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

#5(b)

In [33]:
```
set.seed(1)
train = sample(nrow(Default),nrow(Default)*0.5)
```

In [34]:
```
fit2 = glm(default~income+balance, data=Default, family=binomial,
```

In [35]:
```
prob2 = predict(fit2,Default[-train,], type="response")
pred2 = ifelse(prob2 > 0.5, "Yes", "No")
table(pred2, Default[-train,]$default)
```

```
pred2   No  Yes
  No  4824  108
  Yes   19   49
```

In [36]:
```
mean(Default[-train,]$default!= pred2)
```

0.0254

In [37]:
```
#test error is 0.0254
```

5(c)

In [59]:
```
#repeat 1
set.seed(2)
train = sample(nrow(Default), nrow(Default)*0.5)
fit2 = glm(default~income+balance, data=Default, family=binomial,
           subset=train)
prob2 = predict(fit2,Default[-train,],type="response")
pred2 = ifelse(prob2 > 0.5, "Yes", "No")
mean(Default[-train,]$default != pred2)
```

0.0238

In [60]:
```
#repeat 2
train=sample(nrow(Default), nrow(Default)*0.5)
fit2=glm(default~income+balance, data=Default, family=binomial,
         subset=train)
prob2=predict(fit2, Default[-train,], type="response")
pred2=ifelse(prob2 > 0.5, "Yes", "No")
mean(Default[-train,]$default != pred2)
```

0.0288

In [61]:
```
train=sample(nrow(Default), nrow(Default)*0.5)
fit2=glm(default~income+balance, data=Default, family=binomial,
         subset=train)
prob2=predict(fit2, Default[-train,], type="response")
pred2=ifelse(prob2 > 0.5, "Yes", "No")
mean(Default[-train,]$default != pred2)
```

0.0254

In [41]:
```
# from three repetitions, the test error is 2.5% consistance
#(0.02 appx)very less variance
```

#5(d)

In [62]:
```
set.seed(1)
train=sample(nrow(Default), nrow(Default)*0.5)
fit3=glm(default~income + balance + student, data=Default,
         family=binomial, subset=train)
prob3=predict(fit3, Default[-train,], type="response")
pred3=ifelse(prob3 > 0.5, "Yes", "No")
mean(Default[-train,]$default != pred3)
```

0.026

In [43]:
```
#the test error is almost similar with adding "student".
#there is no significance reduction in the error.
```

#6(a)

In [63]:

```
require(ISLR)
data(Default)
set.seed(1)
fit1=glm(default~income+balance, data=Default,
         family=binomial)
summary(fit1)
```

```
Call:
glm(formula = default ~ income + balance, family = binomial,
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

In [45]:

```
#estimated std error for income is 0.000004985 and for
#balance is 0.0002274
```

6(b)

In [46]:
```
set.seed(1)
boot.fn=function(df, trainid) {
return(coef(glm(default~income + balance, data=df,
                family=binomial, subset=trainid)))
}
boot.fn(Default, 1:nrow(Default))
```

(Intercept)	-11.5404684366776
income	2.08089755005988e-05
balance	0.00564710294333903

6(c)

In [47]:
```
require(boot)
boot(Default,boot.fn,R=100)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Default, statistic = boot.fn, R = 100)


Bootstrap Statistics :
         original        bias     std. error
t1* -1.154047e+01  8.556378e-03 4.122015e-01
t2*  2.080898e-05 -3.993598e-07 4.186088e-06
t3*  5.647103e-03 -4.116657e-06 2.226242e-04
```

In [48]:
```
#std error estimates are very close using glm and bootstrap
#using R=100
#glm-(4.985e-06 for income,2.274e-04 for balance)
#bootstrap-(4.186088e-06 for income,2.226242e-04 for balance)
```

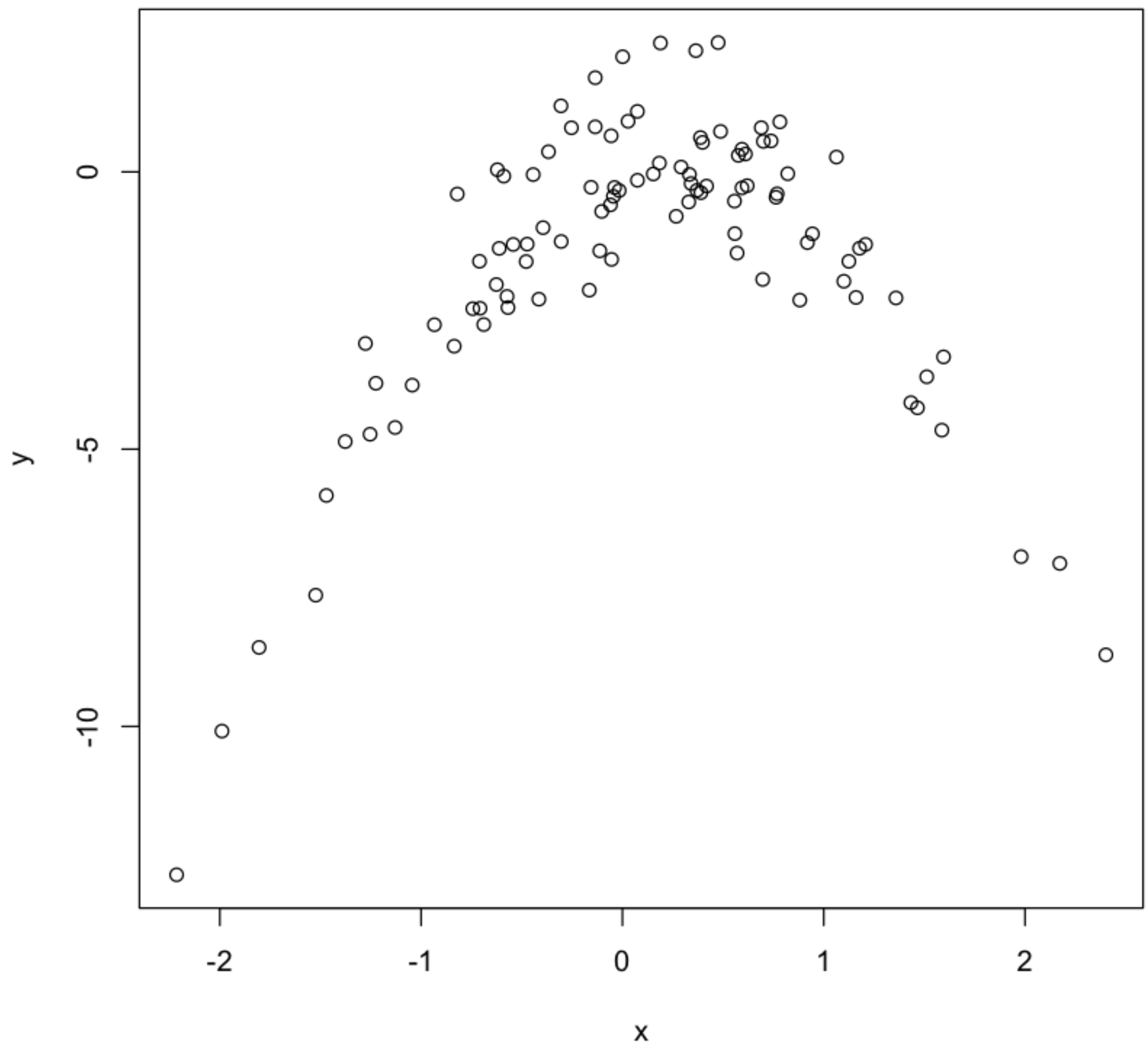
8(a)

In [49]:
```
#n=100 (obvervations)
#p=2 (features)
#equation is--
# Y = X - 2X^2 + \epsilon

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

8(b)

In [50]:
```
plot(x,y)
```



In [51]:
```
#relation b/w x and y is quadratic
```

#8(c)

In [52]:
```
set.seed(1)
df = data.frame(y, x, x2=x^2, x3=x^3, x4=x^4) #df dataframe
fit1 = glm(y~x, data=df)
cv.err1 = cv.glm(df, fit1)
cv.err1$delta
fit2 = glm(y~x + x2, data=df)
cv.err2 = cv.glm(df, fit2)
cv.err2$delta
fit3 = glm(y~x + x2 + x3, data=df)
cv.err3 = cv.glm(df, fit3)
cv.err3$delta
fit4 = glm(y~x + x2 + x3 + x4, data=df)
cv.err4 = cv.glm(df, fit4)
cv.err4$delta
```

7.28816160667281 7.28474411546929

0.937423637615552 0.937178917181123

0.956621830108939 0.956253813731321

0.953904892744804 0.953445283156601

#8(d)

In [53]:
```
set.seed(2)
df = data.frame(y, x, x2=x^2, x3=x^3, x4=x^4)
fit1 = glm(y~x, data=df)
cv.err1 = cv.glm(df, fit1)
cv.err1$delta
fit2 = glm(y~x + x2, data=df)
cv.err2 = cv.glm(df, fit2)
cv.err2$delta
fit3 = glm(y~x + x2 + x3, data=df)
cv.err3 = cv.glm(df, fit3)
cv.err3$delta
fit4 = glm(y~x + x2 + x3 + x4, data=df)
cv.err4 = cv.glm(df, fit4)
cv.err4$delta
```

7.28816160667281 7.2847441154693

0.937423637615552 0.937178917181122

0.95662183010894 0.956253813731321

0.953904892744804 0.953445283156601

In [54]:
```
# no diff in results b/w (c) and (d)
```

8(e)

In [55]:
```
#using quad model,x  x^2 x^3  having the least error.
#it was expected as true model was generated by quadratic model
```

In [56]:
```
fitn = lm(y~poly(x,4))
summary(fitn)
```

```
Call:
lm(formula = y ~ poly(x, 4))

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

Residual standard error: 0.9591 on 95 degrees of freedom
Multiple R-squared:  0.8753,	Adjusted R-squared:  0.8701 
F-statistic: 166.7 on 4 and 95 DF,  p-value: < 2.2e-16
```

In [57]:
```
#from summary, x and x^2 are the significant predictors.
#this agrees with cross-validation results  which indicates using
#x and x^2 gives the best outcome.
```