YK_Assignment6

Yinan Kang

3/30/2019

Note: Hi TAs, I skipped a few sections, you will find 'n/a' next to them. Had a bit less time this week thanks to work (and a trip to see my parents). Still learned a lot from doing this assignment, though this will probably be my 'drop'. Thx -Yinan

```
# Packages

install.packages("car")
install.packages("corrplot")
require(car)
require(corrplot)
require(MASS)
```

Problem 5.26

(a)

```
# Import Data
colnames <- c("y","x")
df <- read.table(url("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets/KutnerDa
n <- nrow(df)

X = matrix(c(rep(1,16),df$x[1],df$x[2],df$x[3],df$x[4],df$x[5],df$x[6],df$x[7],df$x[8],df$x[9],df$x[10]
             ,df$x[12],df$x[13],df$x[14],df$x[15],df$x[16]),nrow=16,ncol=2)
Y = matrix(c(df$y[1],df$y[2],df$y[3],df$y[4],df$y[5],df$y[6],df$y[7],df$y[8],df$y[9],df$y[10],df$y[11]
             ,df$y[12],df$y[13],df$y[14],df$y[15],df$y[16]), nrow=16, ncol=1)

# Calculations
# (1) (X'X)^-1
ginv(t(X) %*% X)
```

```
##             [,1]        [,2]
## [1,]  0.675000 -0.02187500
## [2,] -0.021875  0.00078125
```

```
# (2) b
b<- (ginv(t(X) %*% X)) %*% (t(X)%*%Y)
b
```

```
##              [,1]
## [1,] 168.600000
## [2,]   2.034375
```

```r
# (3) Y-hat
h = X%*%(ginv(t(X) %*% X))%*%t(X)
h %*% Y # Y-hat answer
```

```
##          [,1]
##  [1,] 201.150
##  [2,] 201.150
##  [3,] 201.150
##  [4,] 201.150
##  [5,] 217.425
##  [6,] 217.425
##  [7,] 217.425
##  [8,] 217.425
##  [9,] 233.700
## [10,] 233.700
## [11,] 233.700
## [12,] 233.700
## [13,] 249.975
## [14,] 249.975
## [15,] 249.975
## [16,] 249.975
```

```r
# (4) H
h
```

```
##         [,1]   [,2]   [,3]   [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10]
##  [1,]  0.175  0.175  0.175  0.175 0.100 0.100 0.100 0.100 0.025 0.025
##  [2,]  0.175  0.175  0.175  0.175 0.100 0.100 0.100 0.100 0.025 0.025
##  [3,]  0.175  0.175  0.175  0.175 0.100 0.100 0.100 0.100 0.025 0.025
##  [4,]  0.175  0.175  0.175  0.175 0.100 0.100 0.100 0.100 0.025 0.025
##  [5,]  0.100  0.100  0.100  0.100 0.075 0.075 0.075 0.075 0.050 0.050
##  [6,]  0.100  0.100  0.100  0.100 0.075 0.075 0.075 0.075 0.050 0.050
##  [7,]  0.100  0.100  0.100  0.100 0.075 0.075 0.075 0.075 0.050 0.050
##  [8,]  0.100  0.100  0.100  0.100 0.075 0.075 0.075 0.075 0.050 0.050
##  [9,]  0.025  0.025  0.025  0.025 0.050 0.050 0.050 0.050 0.075 0.075
## [10,]  0.025  0.025  0.025  0.025 0.050 0.050 0.050 0.050 0.075 0.075
## [11,]  0.025  0.025  0.025  0.025 0.050 0.050 0.050 0.050 0.075 0.075
## [12,]  0.025  0.025  0.025  0.025 0.050 0.050 0.050 0.050 0.075 0.075
## [13,] -0.050 -0.050 -0.050 -0.050 0.025 0.025 0.025 0.025 0.100 0.100
## [14,] -0.050 -0.050 -0.050 -0.050 0.025 0.025 0.025 0.025 0.100 0.100
## [15,] -0.050 -0.050 -0.050 -0.050 0.025 0.025 0.025 0.025 0.100 0.100
## [16,] -0.050 -0.050 -0.050 -0.050 0.025 0.025 0.025 0.025 0.100 0.100
##       [,11] [,12]  [,13]  [,14]  [,15]  [,16]
##  [1,] 0.025 0.025 -0.050 -0.050 -0.050 -0.050
##  [2,] 0.025 0.025 -0.050 -0.050 -0.050 -0.050
##  [3,] 0.025 0.025 -0.050 -0.050 -0.050 -0.050
##  [4,] 0.025 0.025 -0.050 -0.050 -0.050 -0.050
##  [5,] 0.050 0.050  0.025  0.025  0.025  0.025
##  [6,] 0.050 0.050  0.025  0.025  0.025  0.025
##  [7,] 0.050 0.050  0.025  0.025  0.025  0.025
##  [8,] 0.050 0.050  0.025  0.025  0.025  0.025
##  [9,] 0.075 0.075  0.100  0.100  0.100  0.100
## [10,] 0.075 0.075  0.100  0.100  0.100  0.100
## [11,] 0.075 0.075  0.100  0.100  0.100  0.100
## [12,] 0.075 0.075  0.100  0.100  0.100  0.100
```

```
## [13,] 0.100 0.100  0.175  0.175  0.175  0.175
## [14,] 0.100 0.100  0.175  0.175  0.175  0.175
## [15,] 0.100 0.100  0.175  0.175  0.175  0.175
## [16,] 0.100 0.100  0.175  0.175  0.175  0.175
```

```r
# (5) SSE
t(Y-X%*%b)%*%(Y-X%*%b)
```

```
##         [,1]
## [1,] 146.425
```

```r
# (6) s2{b}
## calculate MSE
model <- lm(y~x, data=df)
mse <- mean(model$residuals^2)

mse*(ginv(t(X) %*% X)) # s2{b}
```

```
##            [,1]         [,2]
## [1,]  6.1773047 -0.200190430
## [2,] -0.2001904  0.007149658
```

```r
# (7) s2{pred} when Xh = 30
xh <- matrix(c(1,30),nrow=2,ncol=1)
mse*(1+t(xh)%*%(ginv(t(X) %*% X))%*%xh) #s2{pred} answer
```

```
##          [,1]
## [1,] 9.752134
```

(b) n/a

(c) n/a

Problem 6.18

(a)

```r
rm(list=ls())
# Import Data
colnames <- c("y","x1","x2","x3","x4")
df <- read.table(url("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets/KutnerDa
n <- nrow(df)

# Stem and Leaf for each X:
stem(df$x1)
```

```
##
##   The decimal point is at the |
##
##    0 | 0000000000000000
##    2 | 00000000000000000000000
##    4 | 00000
##    6 | 0
##    8 | 0
```

```
##    10 | 00
##    12 | 00000
##    14 | 0000000000000
##    16 | 0000000000
##    18 | 000
##    20 | 00
```

```r
stem(df$x2)
```

```
##
##    The decimal point is at the |
##
##     2 | 0
##     4 | 080003358
##     6 | 012613
##     8 | 00001223456001555689
##    10 | 01334456667777812334466668
##    12 | 000111157778890002
##    14 | 6
```

```r
stem(df$x3)
```

```
##
##    The decimal point is 1 digit(s) to the left of the |
##
##    0 | 00000000000000000000000000000002333333333334444445555556678889
##    1 | 023444469
##    2 | 1223477
##    3 | 3
##    4 |
##    5 | 7
##    6 | 0
##    7 | 3
```

```r
stem(df$x4)
```

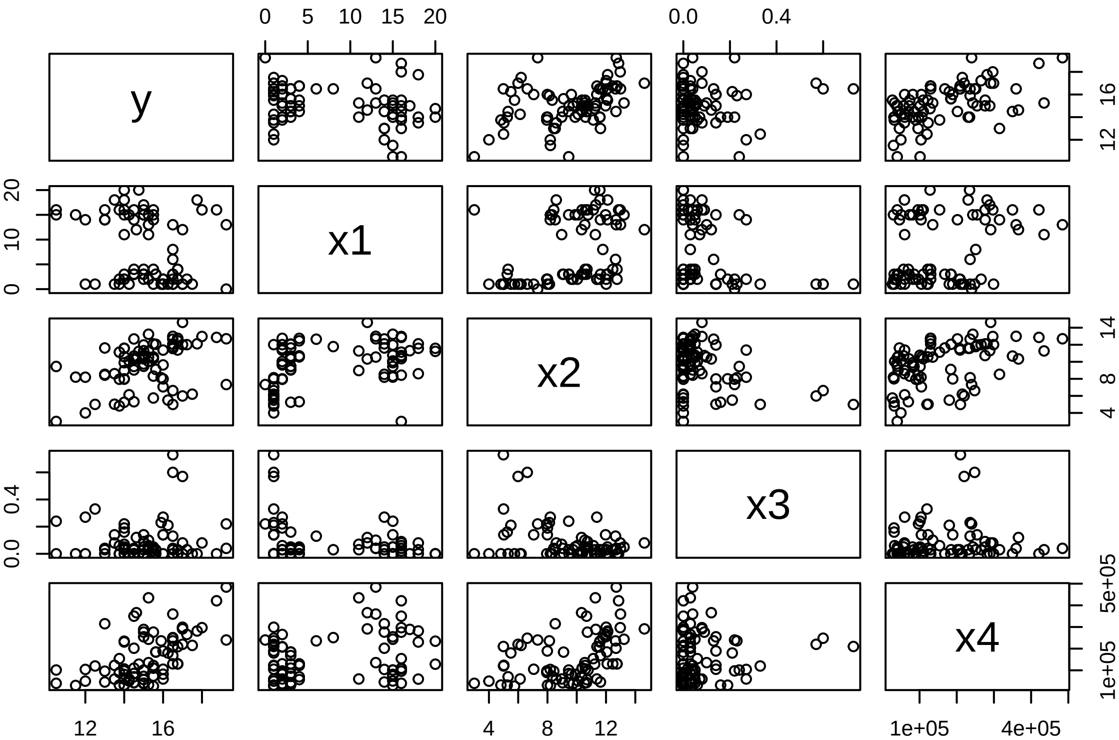
```
##
##    The decimal point is 5 digit(s) to the right of the |
##
##    0 | 333333444444
##    0 | 555666667778899
##    1 | 000001111222333334
##    1 | 578889
##    2 | 011122334444
##    2 | 555788899
##    3 | 002
##    3 | 567
##    4 | 23
##    4 | 8
```

What can be learned from these stem and leaf plots?

Response: We see the distributions of the different X values. X2 and X4 are largely normally distributed, with slight skews (right for X2, left for X4). X1 looks to have a bimodal distribution and X3's observations lie mostly in the 0-10 range, with only a few greater than 20s.

(b)

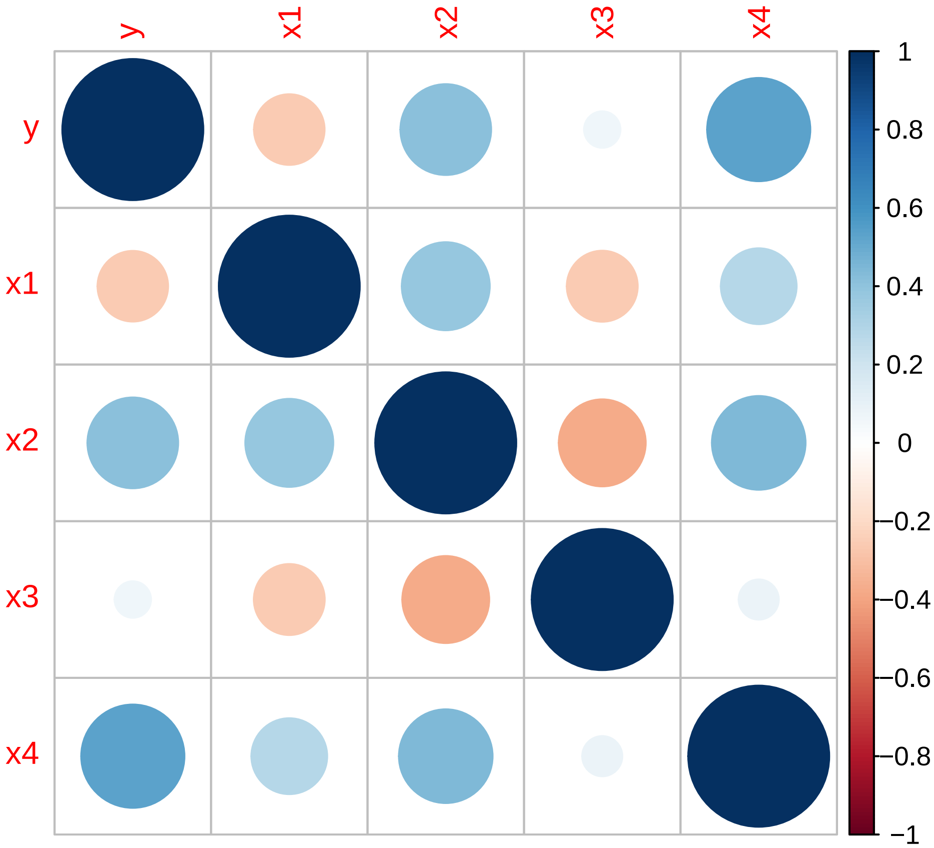
```r
# Scatterplot
pairs(y~x1+x2+x3+x4, data=df)
```




```r
# Correlation Plot
m <- cor(df)
m
```

```
##               y         x1         x2          x3         x4
## y    1.00000000 -0.2502846  0.4137872  0.06652647 0.53526237
## x1  -0.25028456  1.0000000  0.3888264 -0.25266347 0.28858350
## x2   0.41378716  0.3888264  1.0000000 -0.37976174 0.44069713
## x3   0.06652647 -0.2526635 -0.3797617  1.00000000 0.08061073
## x4   0.53526237  0.2885835  0.4406971  0.08061073 1.00000000
```

```r
corrplot(m)
```



Interpretation: We see that the predictors are not too correlated, with the highest correlation being ~0.5 between X1 and X4.

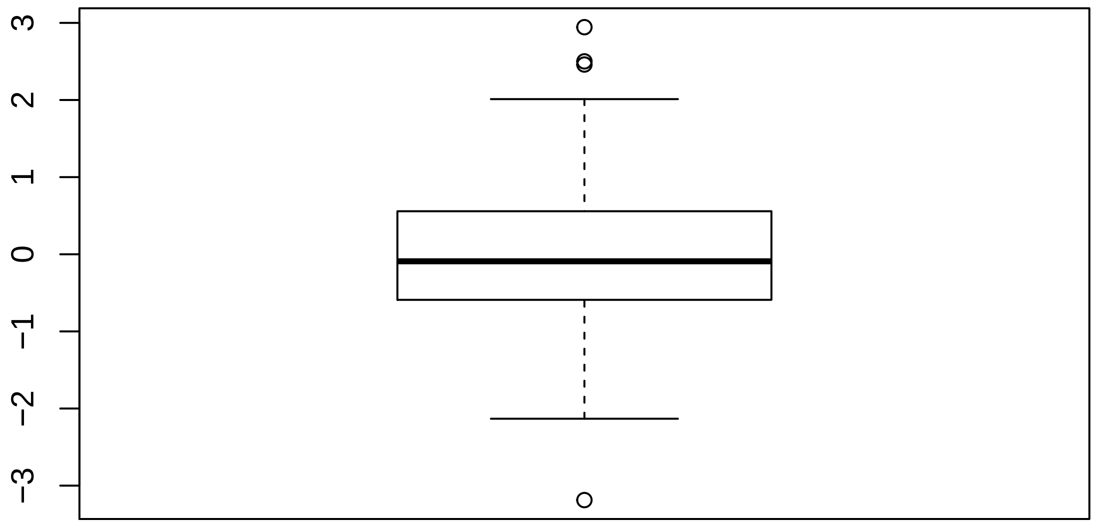
(c)

```r
# Fit regression model
model <- lm(y~x1+x2+x3+x4, data=df)
```

Regression model: y = 1.22e+01 - 1.42e-01(x1) + 2.82e-01(x2) + 6.19e-01(x3) + 7.92e-06(x4)

(d)

```r
# Box-plot of residuals
boxplot(model$residuals)
```

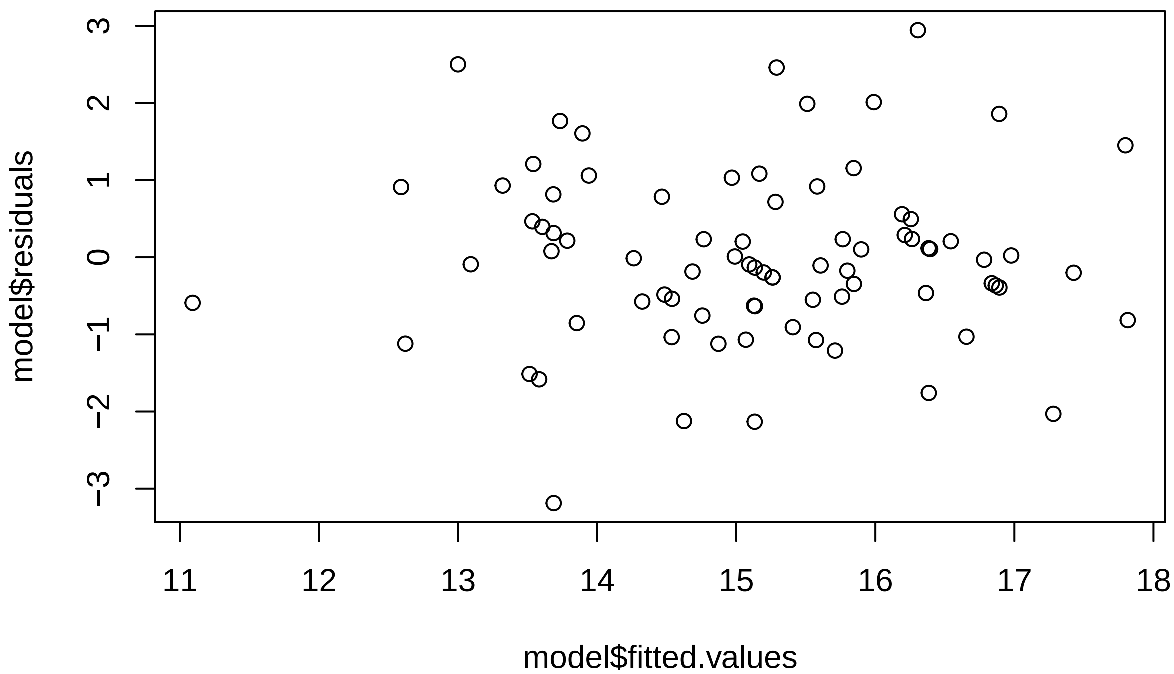


Does the distribution of residuals seem fairly symmetrical?

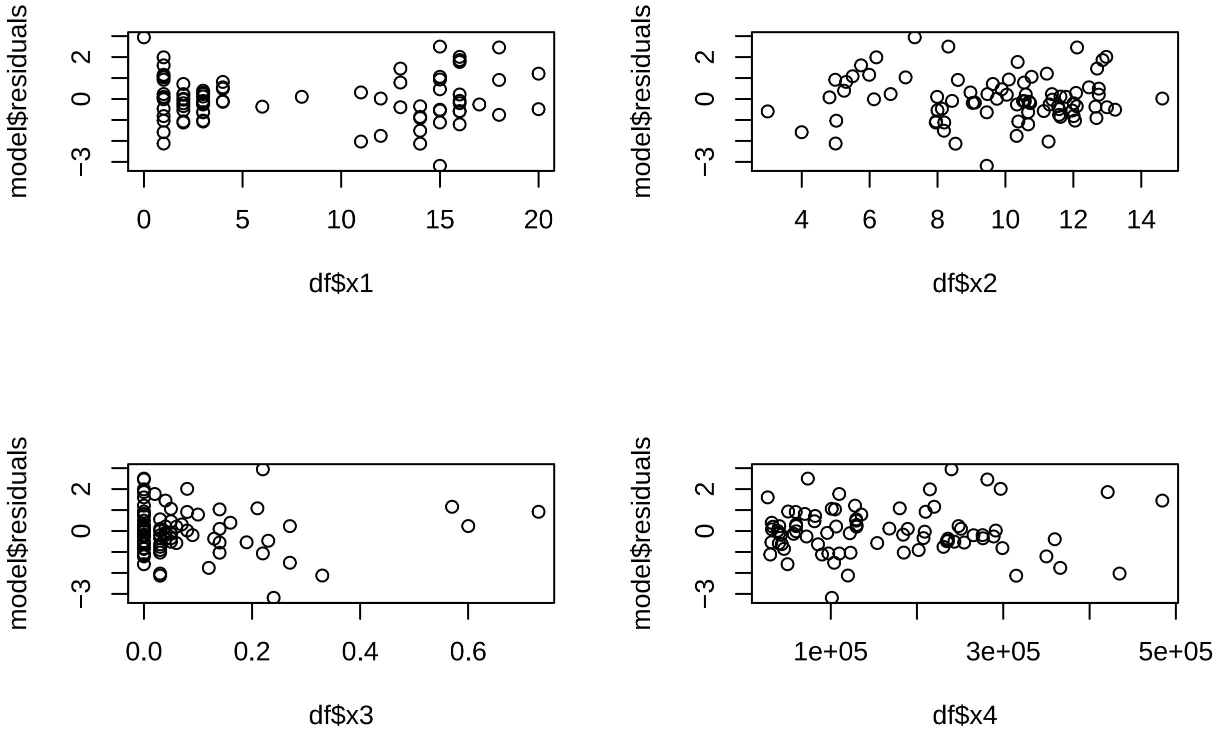
Response: Yes, it does based on boxplot.

(e)

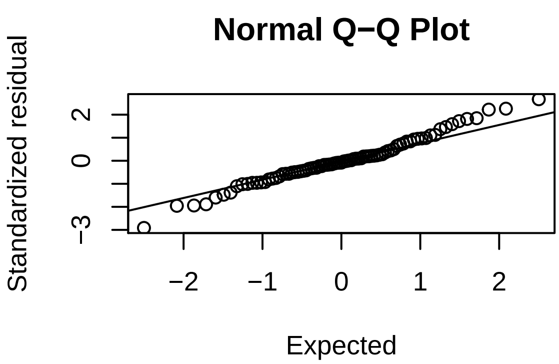
```r
# Plot Residuals vs. Fitted Values
plot(model$residuals ~ model$fitted.values)
```




```r
# Residuals vs. X1,X2,X3,X4
par(mfrow=c(2,2))
plot(model$residuals ~ df$x1)
plot(model$residuals ~ df$x2)
plot(model$residuals ~ df$x3)
plot(model$residuals ~ df$x4)
```




```r
## Normal Probability Plot
model.stan <- rstandard(model)
qqnorm(model.stan, main="Normal Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.stan)
```



Interpretation: The residuals look approximately normal and the distribution does not look to have a pattern.

(f)

Can a formal lack of fit test be conducted here?

Response: Yes, because there are multiple samples for each X, F-test can be performed here.

(g) n/a

Problem 6.19

(a)

Hypothesis Decision:
H0: B1 = B2 = ... = 0
Ha: not all B values (k=1,...,p-1) equal 0

If f.crit <= f ... conclude H0
If f.crit > f ... conclude Ha

```r
# Calculate SSR
anova(model)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value    Pr(>F)
## x1         1 14.819  14.819 11.4649  0.001125 **
## x2         1 72.802  72.802 56.3262 9.699e-11 ***
## x3         1  8.381   8.381  6.4846  0.012904 *
## x4         1 42.325  42.325 32.7464 1.976e-07 ***
## Residuals 76 98.231   1.293
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
SSR <- 14.819+72.802+8.381+42.325
msr <- SSR
mse <- mean(model$residuals^2)

# Calculate F values
f.crit <- msr/mse
p <- 4
n <- n
f <- qf(.95,p-1,n-p)

print(f.crit)
```

```
## [1] 114.0631
```

```r
print(f)
```

```
## [1] 2.723343
```

Hypothesis Decision:
H0: B1 = B2 = ... = 0
Ha: not all B values (k=1,...,p-1) equal 0

If f.crit <= f ... conclude H0
If f.crit > f ... conclude Ha

Decision: Since f.crit > f, we conclude Ha, that there IS a regression relation among the 4 predictors. Also, p value is 7.272e-14.

(b) n/a

(c)

$R^2 = 0.5847$, from summary(model). The multivariate linear regression model we've built only accounts for roughly just greater than half the variance existing in the data, which shows it is likely not a good-fitting model.

Problm 6.28

```r
# Importing Data
rm(list=ls())
cdi.df <- read.csv("CDI.csv")
n <- nrow(cdi.df)
attach(cdi.df)

# Model 1 predictors
x1 <- Total.Population
x2 <- Land.Area
x3 <- Total.personal.income

# Model 2 predictors
x4 <- Total.Population / Land.Area
x5 <- X.Pop.aged.65.and.over/Total.Population
x6 <- Total.personal.income

Y <- Number.Active.Physicians
```

(a)

```r
# Stem and leaf plots
par(mfrow=c(3,2))
stem(x1)
```

```
##
##   The decimal point is 6 digit(s) to the right of the |
##
##   0 | 11111111111111111111111111111111111111111111111111111111111111111111+254
##   0 | 55555555555555555555555666666666666666777777777777777777778888888888
##   1 | 000000122233333444
##   1 | 55699
##   2 | 1134
##   2 | 58
##   3 |
##   3 |
##   4 |
##   4 |
##   5 | 1
##   5 |
##   6 |
```

```
##    6 |
##    7 |
##    7 |
##    8 |
##    8 | 9
```

```r
stem(x2)
```

```
##
##    The decimal point is 3 digit(s) to the right of the |
##
##     0 | 00001111111111112222222222222222222233333333333333333333333333444444+252
##     1 | 00000000000000011111111111111222222222223333334444555666677778889999
##     2 | 0001111466778
##     3 | 3344688
##     4 | 00122368
##     5 | 45
##     6 | 023
##     7 | 29
##     8 | 11
##     9 | 22
##    10 |
##    11 |
##    12 |
##    13 |
##    14 |
##    15 |
##    16 |
##    17 |
##    18 |
##    19 |
##    20 | 1
```

```r
stem(x3)
```

```
##
##    The decimal point is 4 digit(s) to the right of the |
##
##     0 | 11111111111112222222222222222222222222222222222222222222222222222222+263
##     1 | 000000000000111111111222223333344444445555555567788888888999
##     2 | 00111123333444777888899
##     3 | 0255678899
##     4 | 19
##     5 | 59
##     6 |
##     7 |
##     8 |
##     9 |
##    10 |
##    11 | 1
##    12 |
##    13 |
##    14 |
##    15 |
##    16 |
##    17 |
```

```
##   18 | 4
```

```r
stem(x4)
```

```
##
##   The decimal point is 3 digit(s) to the right of the |
##
##    0 | 00000000000000001111111111111111111111111111111111111111111111111111+321
##    2 | 00001112233456700111145
##    4 | 05884
##    6 | 2464
##    8 | 19
##   10 | 378
##   12 |
##   14 | 4
##   16 |
##   18 |
##   20 |
##   22 |
##   24 |
##   26 |
##   28 |
##   30 |
##   32 | 4
```

```r
stem(x5)
```

```
##
##   The decimal point is 5 digit(s) to the left of the |
##
##    0 | 123444566667778888990000111112333333344444445555556666666666778888888
##    2 | 0000000011111122222333333333334444455566666677777788888888899999001111+18
##    4 | 0001112233444445555566666666777777888900001111122233333444555566666778
##    6 | 000112223334455555556688888999999001112222333455666677788888999999
##    8 | 001111122234455566888890000111112222333444444445555678888899
##   10 | 000112233455677789901112334555667889
##   12 | 01222444777990123778
##   14 | 012290
##   16 | 1
##   18 | 1
##   20 |
##   22 | 1
##   24 |
##   26 | 3
##   28 |
##   30 | 45
```

```r
stem(x6)
```

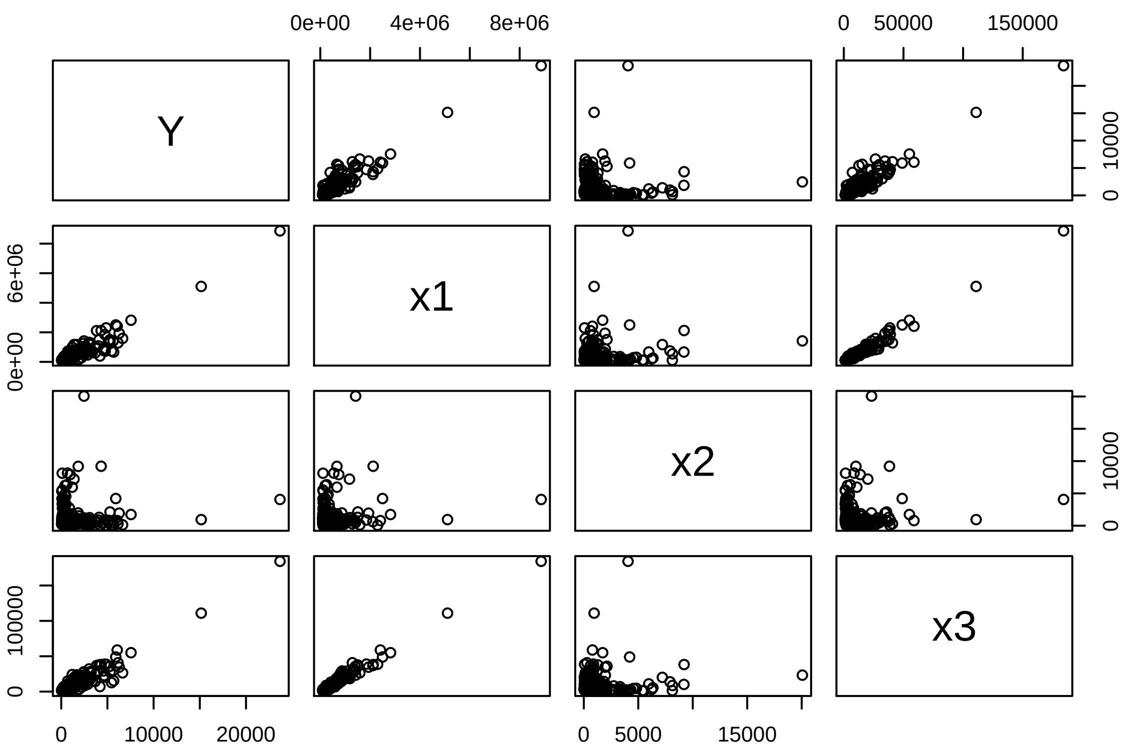
```
##
##   The decimal point is 4 digit(s) to the right of the |
##
##    0 | 11111111111112222222222222222222222222222222222222222222222222222222+263
##    1 | 0000000000001111111112222233333444444455555555677888888888999
##    2 | 0011112333444477788899
##    3 | 0255678899
```

```
##     4 | 19
##     5 | 59
##     6 |
##     7 |
##     8 |
##     9 |
##    10 |
##    11 | 1
##    12 |
##    13 |
##    14 |
##    15 |
##    16 |
##    17 |
##    18 | 4
```

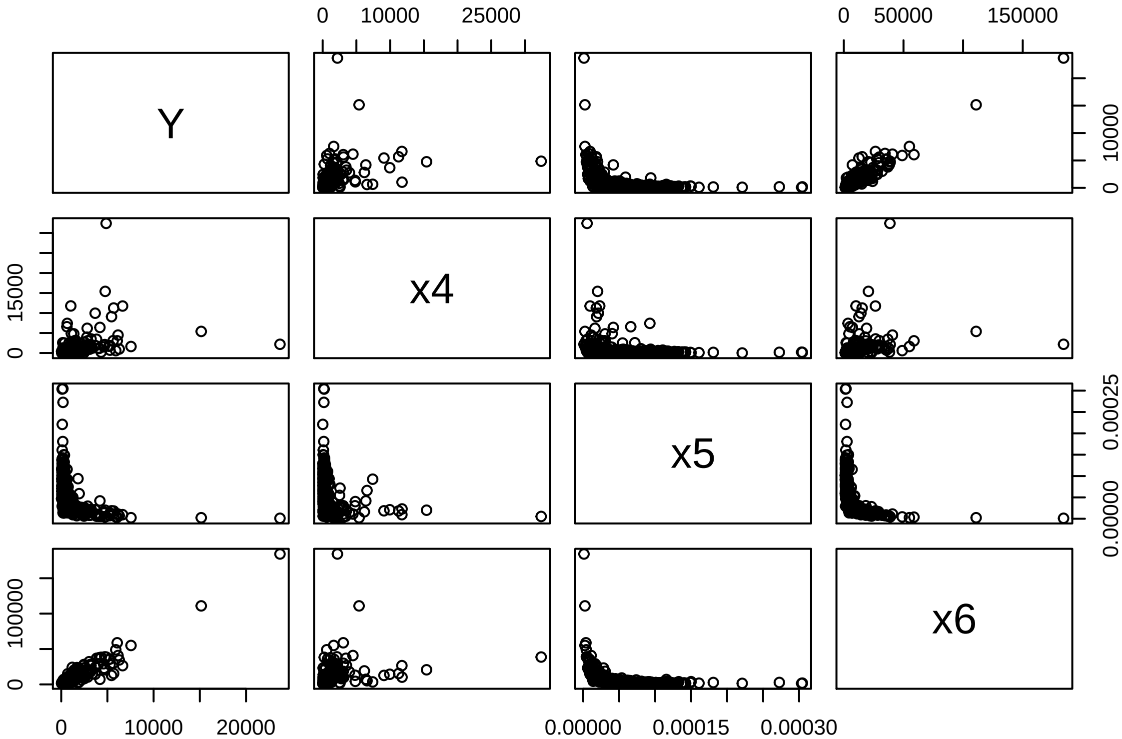
Noteworthy takeaway here is that for each predictor, values skew to the low end.

(b)

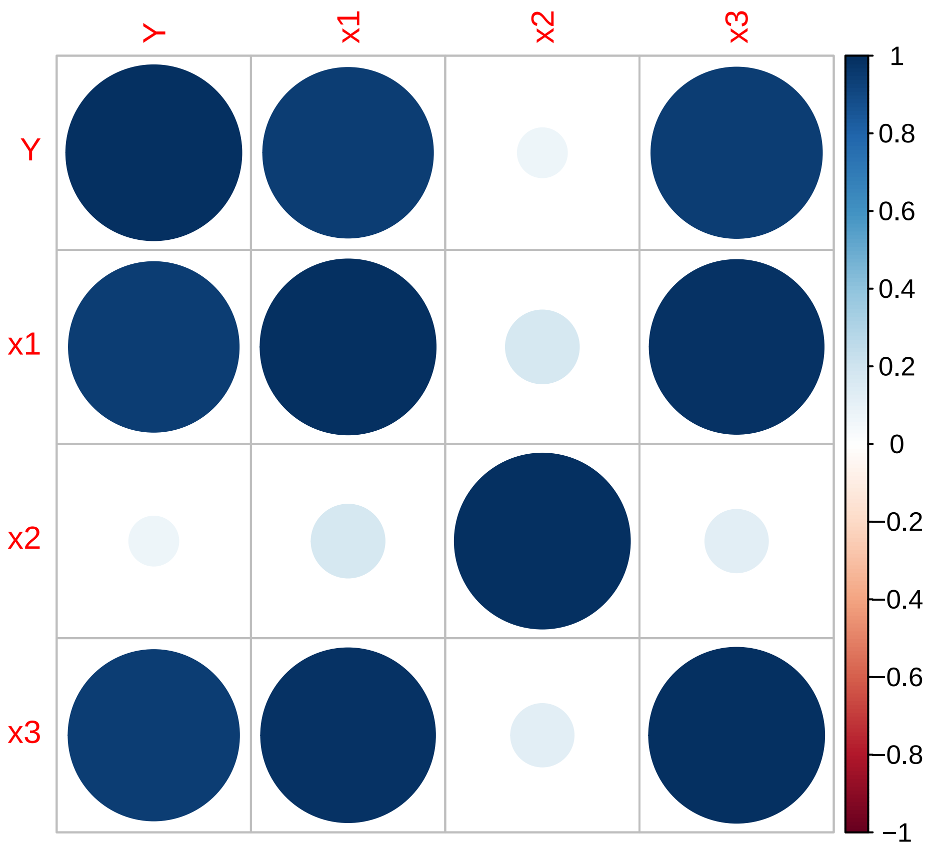
```r
# Generating models
model1 <- lm(Y~x1+x2+x3)
model2 <- lm(Y~x4+x5+x6)

# Scatterplot s
pairs(Y~x1+x2+x3)
```



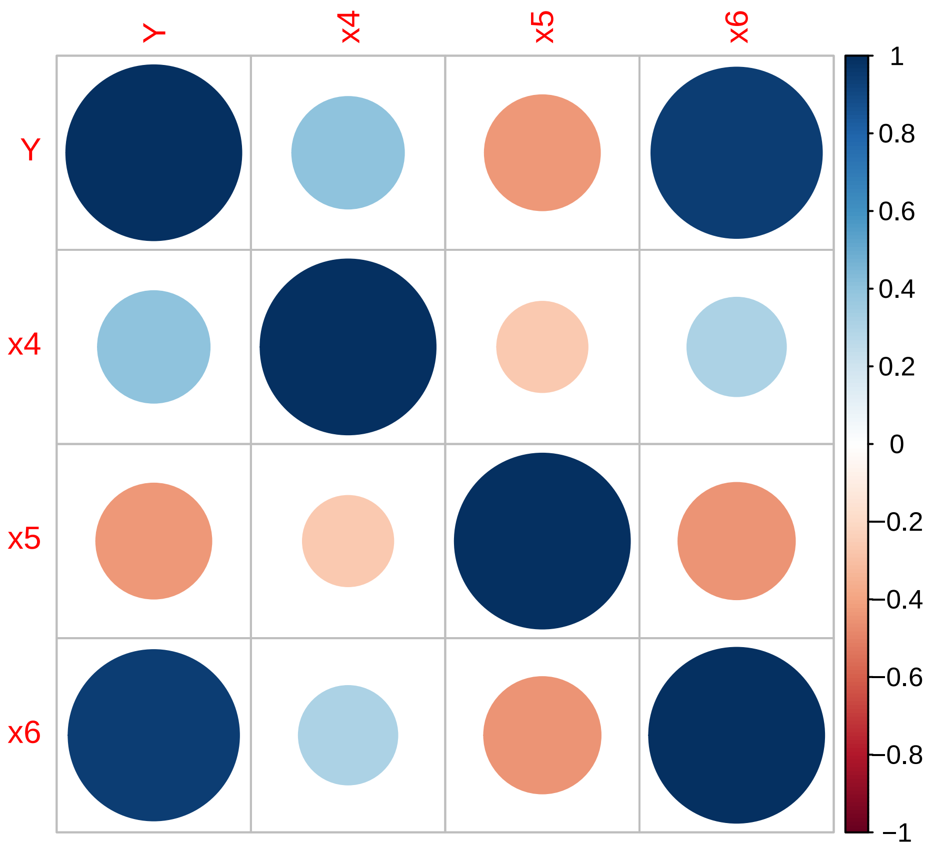

```r
pairs(Y~x4+x5+x6)
```




```
# Correlation Plot
cor1 <- cor(data.frame(Y,x1,x2,x3))
cor2 <- cor(data.frame(Y,x4,x5,x6))
corrplot(cor1)
```




```
corrplot(cor2)
```



Findings:

From scatterplot, we see close to linear relationship between personal income (x3,x6) and Y and also total population (x1) and Y.

From correlation matrix, for model 1, we see that total population and personal income (x1 and x3) are heavily correlated together and with Y. Form model 2, personal income (x6) heavily correlates with Y.

(c) Fit model, already done in (b)

(d)

```r
summary(model1)
```

```
##
## Call:
## lm(formula = Y ~ x1 + x2 + x3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1855.6  -215.2   -74.6    79.0  3689.0
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.332e+01  3.537e+01  -0.377 0.706719
## x1           8.366e-04  2.867e-04   2.918 0.003701 **
## x2          -6.552e-02  1.821e-02  -3.597 0.000358 ***
## x3           9.413e-02  1.330e-02   7.078 5.89e-12 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 560.4 on 436 degrees of freedom
## Multiple R-squared:  0.9026, Adjusted R-squared:  0.902
## F-statistic:  1347 on 3 and 436 DF,  p-value: < 2.2e-16
```

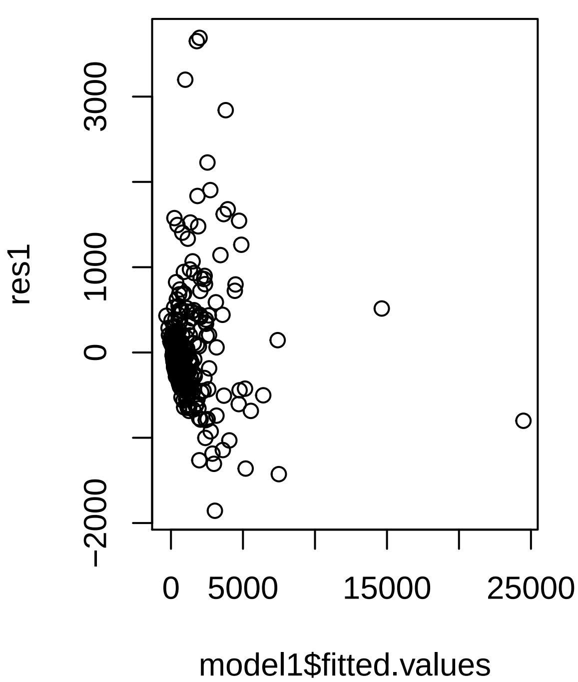
```r
summary(model2)
```

```
##
## Call:
## lm(formula = Y ~ x4 + x5 + x6)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3075.25  -171.71   -36.99    67.50  3053.46
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.017e+02  5.751e+01  -1.769   0.0776 .
## x4           9.694e-02  1.238e-02   7.831 3.72e-14 ***
## x5           1.174e+05  6.776e+05   0.173   0.8625
## x6           1.267e-01  2.266e-03  55.888  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 534.1 on 436 degrees of freedom
## Multiple R-squared:  0.9116, Adjusted R-squared:  0.9109
## F-statistic:  1498 on 3 and 436 DF,  p-value: < 2.2e-16
```

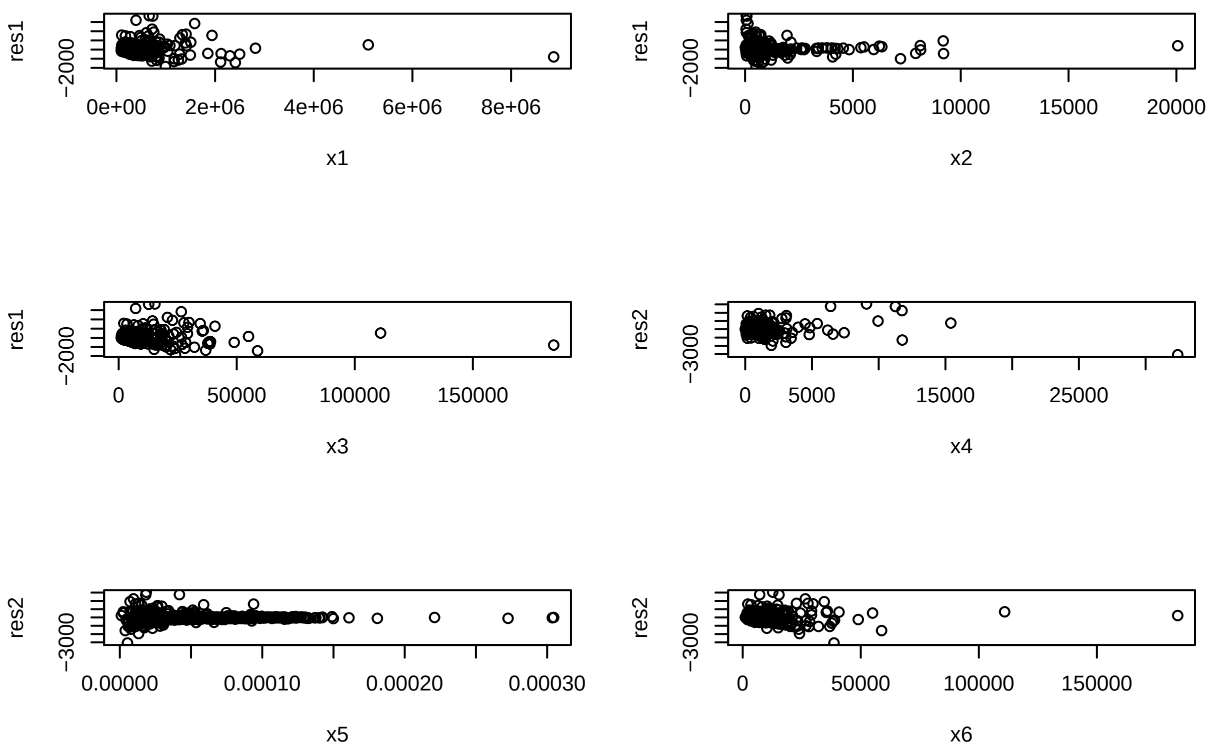
Conclusion:
Model 2 produces the higher R^2 value, thus explaining more variation than Model 1.

(e)

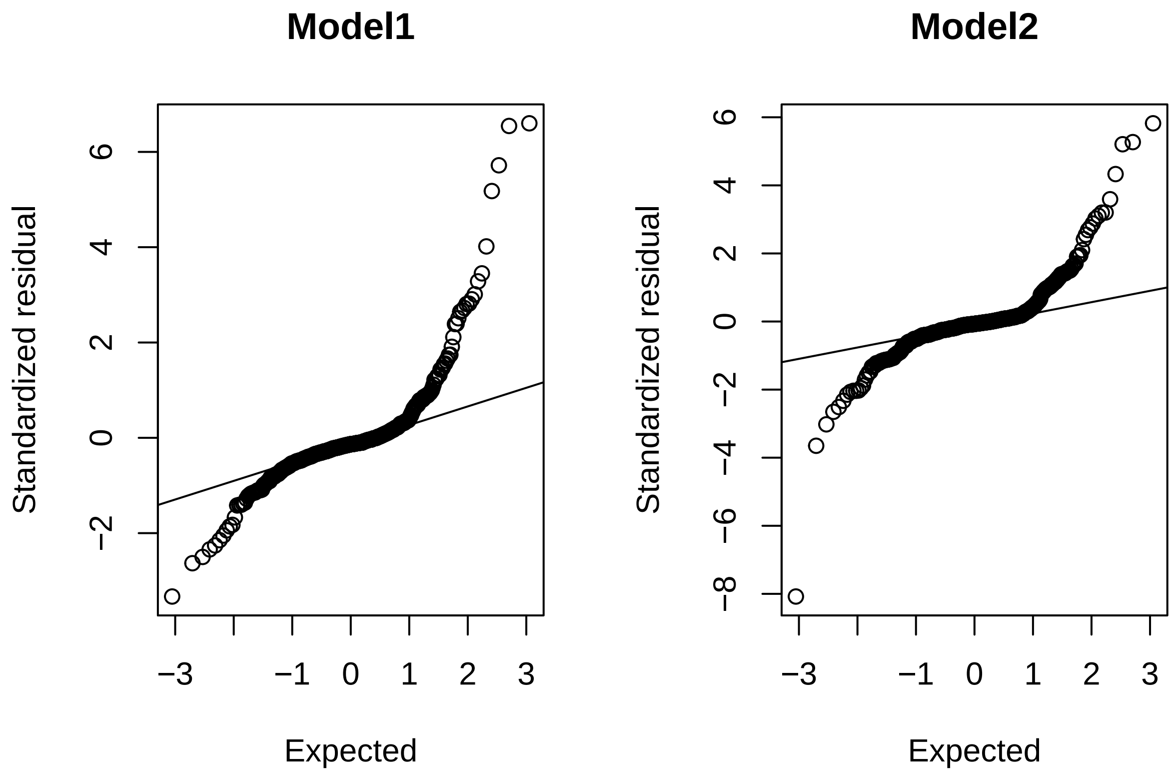
```r
# Obtaining residuals:
res1 <- model1$residuals
res2 <- model2$residuals

# Residual plots:
par(mfrow=c(1,2))
plot(res1~model1$fitted.values)
plot(res2~model2$fitted.values)
```




```
par(mfrow=c(3,2))
plot(res1~x1)
plot(res1~x2)
plot(res1~x3)
plot(res2~x4)
plot(res2~x5)
plot(res2~x6)
```



```r
# Normal Probability Plots

par(mfrow=c(1,2))
model1.stan <- rstandard(model1)
model2.stan <- rstandard(model2)
qqnorm(model1.stan, main="Model1", xlab="Expected", ylab="Standardized residual")
qqline(model1.stan)
qqnorm(model2.stan, main="Model2", xlab="Expected", ylab="Standardized residual")
qqline(model2.stan)
```



Analysis: We can see that the residuals for Model 2 are more well-behaved than Model 1. Residuals are closer to 0 for more of its predictors, and the Q-Q plot shows more normality than for Model 1.

7.7 n/a