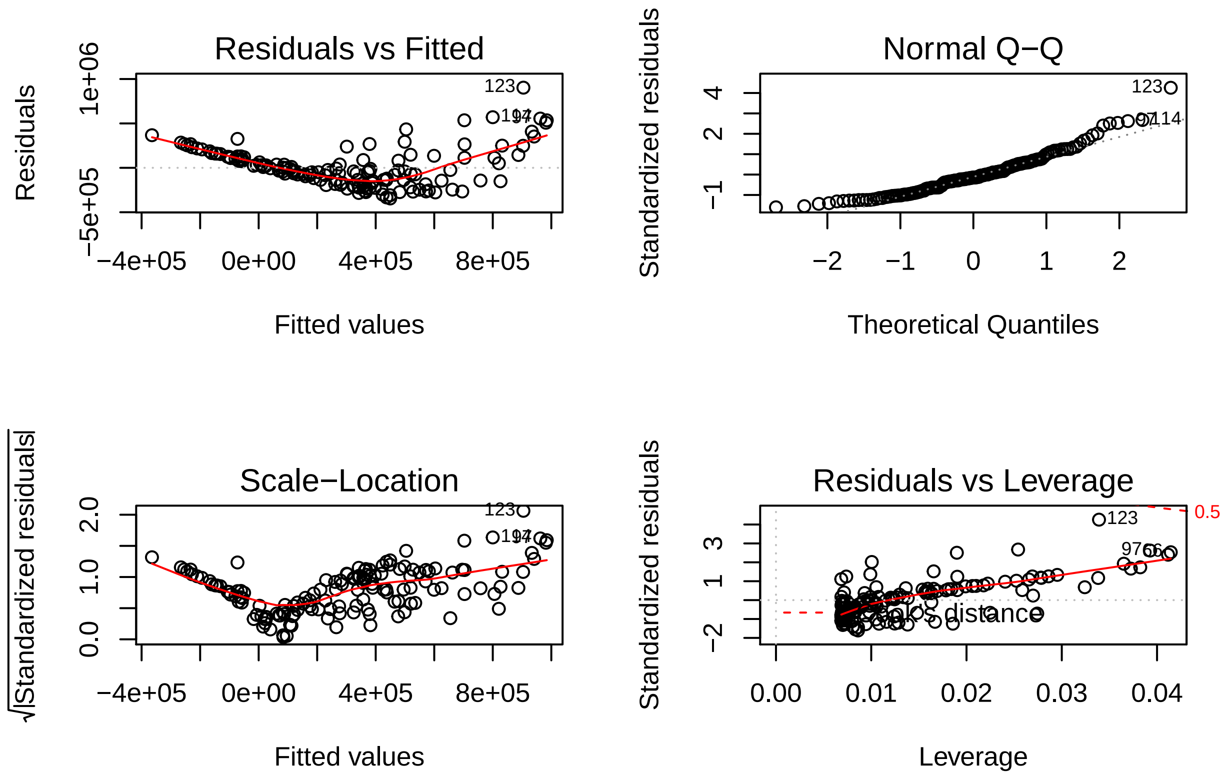
Home Work 4

Shashi Roshan (NetID : sroshan2)

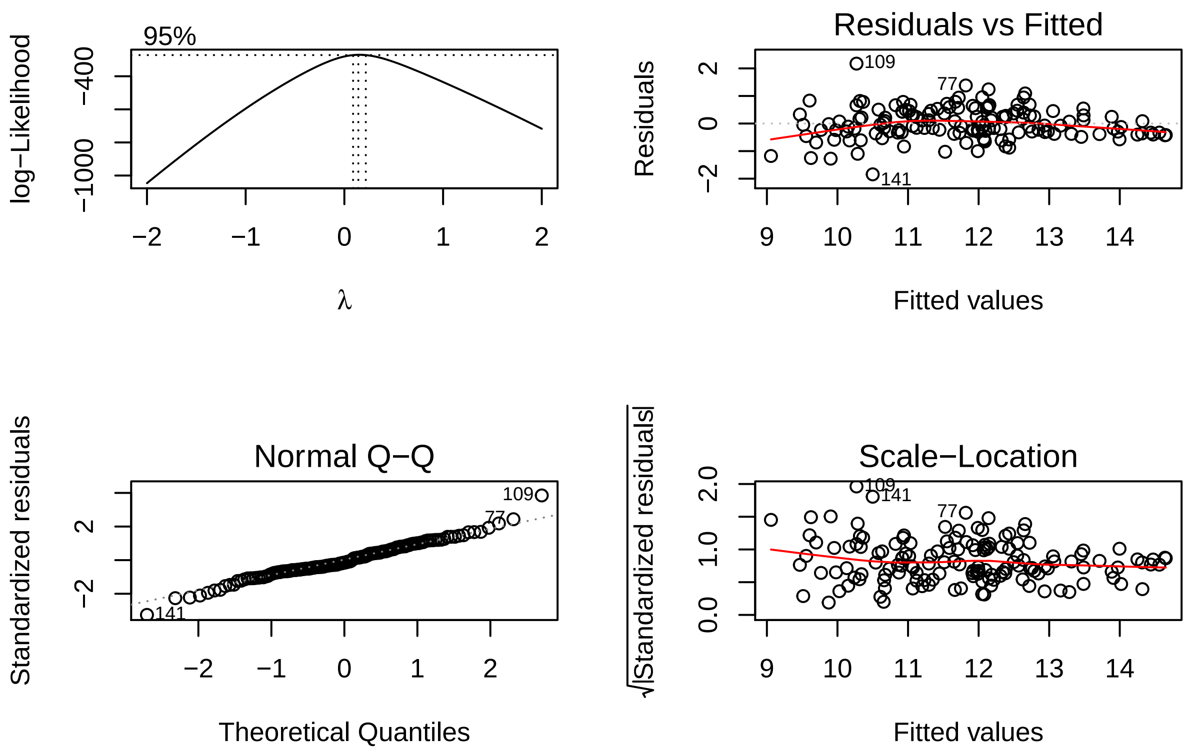
```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

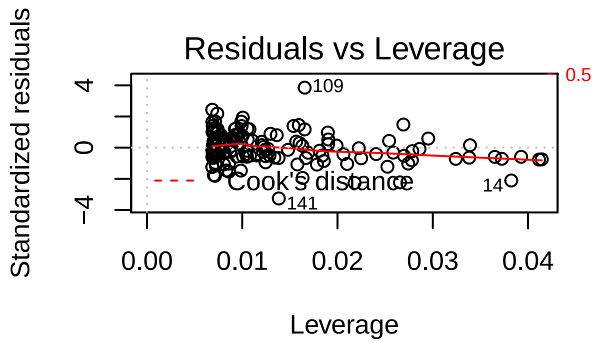
Question 1 (a)

```r
data = read.csv("lpga2009.csv", header = T)
data$X = NULL
data$Golfer = NULL

m1 = lm(prize ~ percentile, data = data)
par(mfrow = c(2, 2))

plot(m1)
```




```r
library(MASS)
boxcox(prize ~ percentile, data = data)

m2 = lm(log(prize) ~ percentile, data = data)

plot(m2)
```


95%
log−Likelihood
−400
−1000
−2
−1
0
1
2
λ
Residuals vs Fitted
Residuals
109
77
141
9
10
11
12
13
14
Fitted values
Normal Q−Q
Standardized residuals
109
77
141
Theoretical Quantiles
Scale−Location
√|Standardized residuals|
2.0
1.0
0.0
Fitted values



From boxcox, optimal lambda is around 0.2 (which can be rounded to 0). Hence taking log transformation on response variable.

Question 1 (b)

The transformed residuals are not normally distributed.

Question 1 (c)

```r
# high leverage points
hat_matrix_val = hatvalues(m2)
model_matrix = model.matrix(m2)
cutoff = (2 * ncol(model_matrix)) / (nrow(model_matrix))
leverage_points = which(hat_matrix_val >= cutoff)
cat('Leverage_points : ')
```

```
## Leverage_points :
```

```r
print(unique(leverage_points))
```

```
##  [1]  14  17  20  37  66  74  78  90  97 104 123 127 129 134
```

```r
# outliers
student.res <- rstandard(m2)
cat('Outliers : \n')
```

```
## Outliers :
```

```r
print(student.res[abs(student.res) > 2])
```

```
##         9        14        77        85       109       140       141
## -2.228619 -2.107811  2.433927  2.185379  3.849720 -2.262971 -3.259379
```

```r
# influential points
cutoff = 0.5
cooks_distance = cooks.distance(m2)
outliers = which(cooks_distance >= cutoff)
cat('Influential points : None\n')
```

```
## Influential points : None
```

```r
print(outliers)
```

```
## named integer(0)
```

If we set cutoff value of studentized residual as 2, then observations at these positions which are outliers : (9, 14, 77, 85, 109, 140, 141).

Question 1 (d)

```r
m3 = lm(ln.prize. ~ . - prize, data=data)
print(vif(m3))
```

```
##                drive              fairways             pct_greens
##             2.718513              1.975722              15.705538
##            ave_putts         per_sandsaves           ntournaments
##             9.465912              1.325023              26.743236
##             regputts completed_tournaments             percentile
##             6.425568             26.563694              30.612646
##     rounds_completed               strokes
##            77.948704             38.080500
```

Number of predictors with (VIF > 5) : 8

Since many predictors have VIF > 5, collinearity is an issue.

Question 1 (e)

```r
# using AIC
m3_step_1 = step(m3, direction='backward', k=2, trace=F)
cat('Final model : \n')
```

```
## Final model :
```

```r
print(m3_step_1)
```

```
##
## Call:
## lm(formula = ln.prize. ~ drive + fairways + ave_putts + regputts +
##     rounds_completed + strokes, data = data)
##
## Coefficients:
##      (Intercept)             drive          fairways         ave_putts
##         71.80194          -0.02014          -0.02935           0.32958
##         regputts  rounds_completed           strokes
##         -6.54330           0.03165          -0.72546
```

Question 1 (f)

```r
# using BIC
m3_step_2 = step(m3, direction='backward', k=log(nrow(data)), trace=F)
cat('Final model : \n')
```

```
## Final model :
```

```r
print(m3_step_2)
```

```
##
## Call:
## lm(formula = ln.prize. ~ drive + fairways + ave_putts + regputts +
##     rounds_completed + strokes, data = data)
##
## Coefficients:
##      (Intercept)             drive          fairways         ave_putts
##         71.80194          -0.02014          -0.02935           0.32958
##         regputts  rounds_completed           strokes
##         -6.54330           0.03165          -0.72546
```

Question 1 (g)

```r
# using AIC
m3_step_3 = step(lm(ln.prize. ~ 1, data=data), direction='forward', list(upper = m3, lower = ~1), k=2,
cat('Final model : \n')
```

```
## Final model :
```

```r
print(m3_step_3)
```

```
##
## Call:
## lm(formula = ln.prize. ~ percentile + rounds_completed + strokes +
##     fairways + drive, data = data)
##
## Coefficients:
##      (Intercept)        percentile  rounds_completed           strokes
##        64.614229          0.008019          0.031091         -0.689694
##         fairways             drive
##        -0.027596         -0.012065
```

Question 1 (h)

```r
# using BIC
m3_step_4 = step(lm(ln.prize. ~ 1, data=data), direction='forward', list(upper = m3, lower = ~1), k=log
cat('Final model : \n')
```

```
## Final model :
```

```r
print(m3_step_4)
```

```
##
## Call:
## lm(formula = ln.prize. ~ percentile + rounds_completed + strokes +
##     fairways, data = data)
##
## Coefficients:
##      (Intercept)        percentile  rounds_completed           strokes
##        57.113063          0.007762          0.031157         -0.637315
##         fairways
##        -0.017826
```

Question 1 (i)

```r
n.folds = 10
SSE.predict = numeric(n.folds)
# choosing folds
folds = rep_len(1:n.folds, nrow(data))

# Using 1.e model (Starting with the full model, Use Backwards Elimination to select a model based on A
for(k in 1:n.folds) {
     # actual split of the data
     test.index <- which(folds == k)
     data.train <- data[-test.index,]
     data.test <- data[test.index,]
     lm.temp <- m3_step_1
```

```r
    SSE.predict[k] <- crossprod(predict(lm.temp, data.test)-data.test$ln.prize.)
}
print(sum(SSE.predict))
```

```
## [1] 24.8493
```

```r
# Using 1.f model (Starting with the full model, Use Backwards Elimination to select a model based on B
SSE.predict = numeric(n.folds)

for(k in 1:n.folds) {
    # actual split of the data
    test.index <- which(folds == k)
    data.train <- data[-test.index,]
    data.test <- data[test.index,]
    lm.temp <- m3_step_2
    SSE.predict[k] <- crossprod(predict(lm.temp, data.test)-data.test$ln.prize.)
}
print(sum(SSE.predict))
```

```
## [1] 24.8493
```

```r
# Using 1.g model (Starting with the intercept only model, Use Forward Selection to select a model base
SSE.predict = numeric(n.folds)

for(k in 1:n.folds) {
    # actual split of the data
    test.index <- which(folds == k)
    data.train <- data[-test.index,]
    data.test <- data[test.index,]
    lm.temp <- m3_step_3
    SSE.predict[k] <- crossprod(predict(lm.temp, data.test)-data.test$ln.prize.)
}
print(sum(SSE.predict))
```

```
## [1] 26.64009
```

```r
# Using 1.h model (Starting with the intercept only model, Use Forward Selection to select a model base
SSE.predict = numeric(n.folds)

for(k in 1:n.folds) {
    # actual split of the data
    test.index <- which(folds == k)
    data.train <- data[-test.index,]
    data.test <- data[test.index,]
    lm.temp <- m3_step_4
    SSE.predict[k] <- crossprod(predict(lm.temp, data.test)-data.test$ln.prize.)
}
print(sum(SSE.predict))
```

```
## [1] 27.50501
```

Cross validation prefers the model obtained by performing backward elimination on Full model (using AIC and BIC).

Question 1 (j)

```
print(summary(m3_step_1)$adj.r.squared)
```

```
## [1] 0.9086872
```

```
print(summary(m3_step_2)$adj.r.squared)
```

```
## [1] 0.9086872
```

```
print(summary(m3_step_3)$adj.r.squared)
```

```
## [1] 0.9028059
```

```
print(summary(m3_step_4)$adj.r.squared)
```

```
## [1] 0.9003621
```

Adjusted R2 prefers the model obtained by performing backward elimination on Full model (using AIC and BIC).

Question 3 (a)

```
data3 = read.csv("reaction.csv", header = T)

mean_1 = mean(data3$regular)
mean_2 = mean(data3$upside.down)

t.test(data3$regular, data3$upside.down, paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  data3$regular and data3$upside.down
## t = -0.52819, df = 19, p-value = 0.6035
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -8.972377  5.356377
## sample estimates:
## mean of the differences
##                  -1.808
```

Since the p-value is 0.6035, we cannot reject the null hypothesis. Hence, we conclude that there exists no difference between the mean of the two samples.

Question 3 (b)

```r
sum(data3$regular > data3$upside.down)
```

```
## [1] 15
```

Number of observations who have better (lower) reaction times in the Upside Down : 15

Question 3 (c)

```r
set.seed(1)

res = c()
for(i in 1:1000){
  x = sample(data3$regular)
  res = c(res, sum(x >= data3$upside.down))
}

cat('p-value : ')
```

```
## p-value :
```

```r
cat(sum(res>=15) / 1000)
```

```
## 0.011
```

We can reject the null hypothesis since the p-value is less than threshold. Hence, there is significant evidence at alpha = 0.05 to conclude that most people have faster reaction times in the upside down.

Question 4 (a)

```r
data4 = read.csv("polydata.csv", header = T)

m5_ = lm(y ~ poly(x, 5), data=data4)
summary(m5_)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 5), data = data4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -180.026  -47.079   -5.587   47.631  178.020
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    217.99       5.24  41.605   <2e-16 ***
## poly(x, 5)1  1929.21      74.28  25.971   <2e-16 ***
```

```
## poly(x, 5)2    33.34      74.28   0.449    0.654    
## poly(x, 5)3   933.91      74.28  12.572   <2e-16 ***
## poly(x, 5)4   -18.98      74.28  -0.255    0.799    
## poly(x, 5)5  -120.58      74.28  -1.623    0.106    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 74.28 on 195 degrees of freedom
## Multiple R-squared:  0.8108, Adjusted R-squared:  0.8059 
## F-statistic: 167.1 on 5 and 195 DF,  p-value: < 2.2e-16
```

```
m4_ = lm(y ~ poly(x, 4), data=data4)
summary(m4_)
```

```
## 
## Call:
## lm(formula = y ~ poly(x, 4), data = data4)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -186.361  -47.407   -6.424   44.403  186.674 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  217.992      5.261  41.433   <2e-16 ***
## poly(x, 4)1 1929.206     74.593  25.863   <2e-16 ***
## poly(x, 4)2   33.336     74.593   0.447    0.655    
## poly(x, 4)3  933.912     74.593  12.520   <2e-16 ***
## poly(x, 4)4  -18.979     74.593  -0.254    0.799    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 74.59 on 196 degrees of freedom
## Multiple R-squared:  0.8082, Adjusted R-squared:  0.8043 
## F-statistic: 206.5 on 4 and 196 DF,  p-value: < 2.2e-16
```

```
m3_ = lm(y ~ poly(x, 3), data=data4)
summary(m3_)
```

```
## 
## Call:
## lm(formula = y ~ poly(x, 3), data = data4)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -186.449  -48.766   -6.227   43.362  188.339 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  217.992      5.249  41.531   <2e-16 ***
## poly(x, 3)1 1929.206     74.415  25.925   <2e-16 ***
## poly(x, 3)2   33.336     74.415   0.448    0.655    
## poly(x, 3)3  933.912     74.415  12.550   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 74.42 on 197 degrees of freedom
## Multiple R-squared:  0.8081, Adjusted R-squared:  0.8052
## F-statistic: 276.6 on 3 and 197 DF,  p-value: < 2.2e-16
```

3rd order polynomial best suits this data.

Question 4 (b)

```
{plot(data4$x, data4$y)
lines(data4$x, predict(m3_, newdata=data4))
lines(data4$x, predict(m3_, newdata=data4, interval='prediction', level=0.95)[,3], col='red')
lines(data4$x, predict(m3_, newdata=data4, interval='prediction', level=0.95)[,2], col='green')
legend("topleft", legend = c("Upper Bound", "Prediction", "Lower Bound"), col = c("red", "black", "green
}
```