Sharvi Tomar HW3

Due: Oct 22, 2021

Contents

Problem 1

For the salmonella data set fit a linear model with colonies as the response, and log(dose + 1) as predictor. Check for lack of fit.

```
library(faraway)
data1 = salmonella
head(data1)
```

```
##   colonies dose
## 1       15    0
## 2       21    0
## 3       29    0
## 4       16   10
## 5       18   10
## 6       21   10
```

Since σ^2 is not known, it is possible to check for lack-of-fit by comparing an estimate of σ^2 on a general model only when there is some replication. The data has replicates as known from above, hence it is possible to check for lack-of-fit using partial F-test.

Null Hypothesis: Current Model (No lack of fit)

Alternate Hypothesis: General Model (Lack of fit)

```
# Fitting current model
model_1 = lm(colonies ~ log(dose + 1), data = data1)
# Fitting generalised model
model_1factor = lm(colonies ~ factor(log(dose + 1)), data = data1)
```

```
# Comparing 2 models using ANOVA
anova(model_1, model_1factor)
```

```
## Analysis of Variance Table
##
## Model 1: colonies ~ log(dose + 1)
## Model 2: colonies ~ factor(log(dose + 1))
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     16 1881.1
## 2     12 1091.3  4    789.73 2.1709 0.1342
```

```
# Calculating p-value
1 - pf(2.1709, 4, 12)
```

```
## [1] 0.1341985
```

Since p-value>0.05, we fail to reject the null hypothesis and hence, the current model is sufficient to explain the variability in the response variable. Since the model proposed in the null hypothesis is statistically significant, we conclude that there is no lack of fit.

Problem 2

The gammaray data set shows the x-ray decay light curve of gamma ray burst. Note that the measurement errors on the response are provided. Build a model to predict the flux as a function of time using appropriate weights. Is any transformation suggested for the response and/or the predictors?

```
data2 = gammaray
head(data2)
```

```
##   time  flux error
## 1  133 122.7   5.7
## 2  143 109.5   5.4
## 3  153 101.4   5.2
## 4  163  92.0   4.9
## 5  173  86.8   4.8
## 6  183  83.7   4.7
```

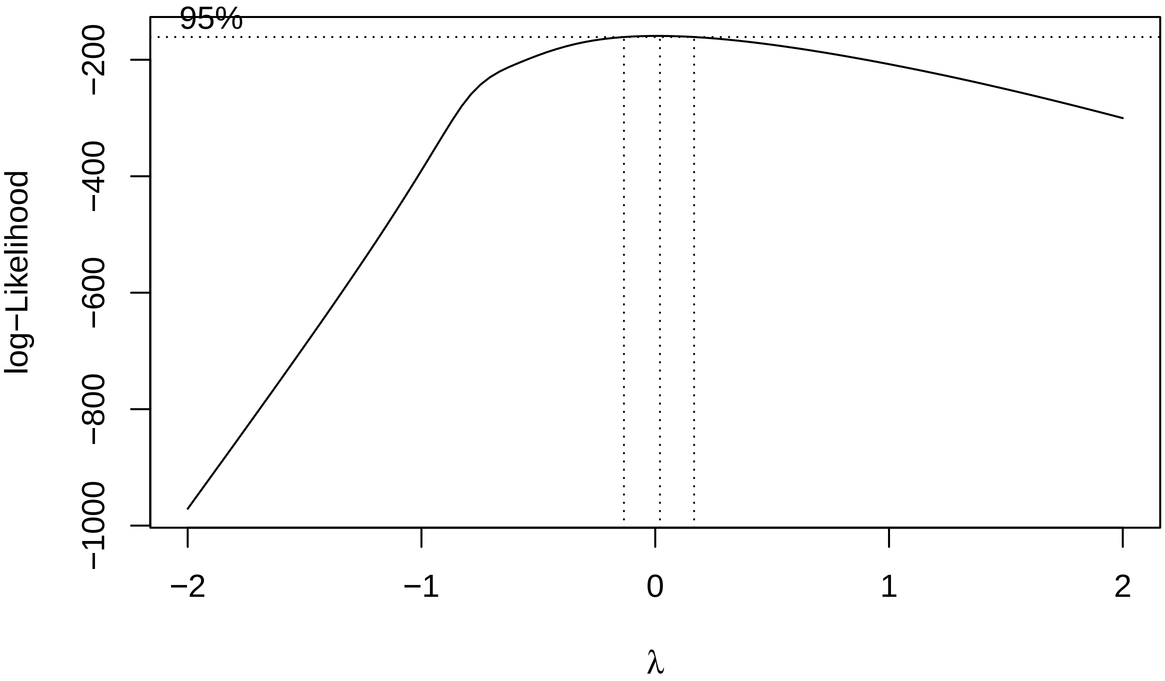
We will be starting with a simplistic weight of 1/error.

```
# Fitting model with weights=1/error
model_2 = lm(flux ~ time, data = data2, weights = 1 / error)
summary(model_2)
```

```
##
## Call:
## lm(formula = flux ~ time, data = data2, weights = 1/error)
##
## Weighted Residuals:
##    Min     1Q Median     3Q    Max
## -21.88  14.03  19.03  27.74  50.73
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.592e+00  1.032e+00   1.543    0.128
## time        -4.284e-06  3.109e-06  -1.378    0.173
##
## Residual standard error: 24.09 on 61 degrees of freedom
## Multiple R-squared:  0.03018,    Adjusted R-squared:  0.01428
## F-statistic: 1.898 on 1 and 61 DF,  p-value: 0.1733
```

We can see that the R-squared value for the model 0.03018 is very low. Hence, we need to make transformations. We can check for tranformations required in the response variable using the Box-Cox method.

```
#Box-Cox Transformation
library(MASS)
boxcox(model_2)
```



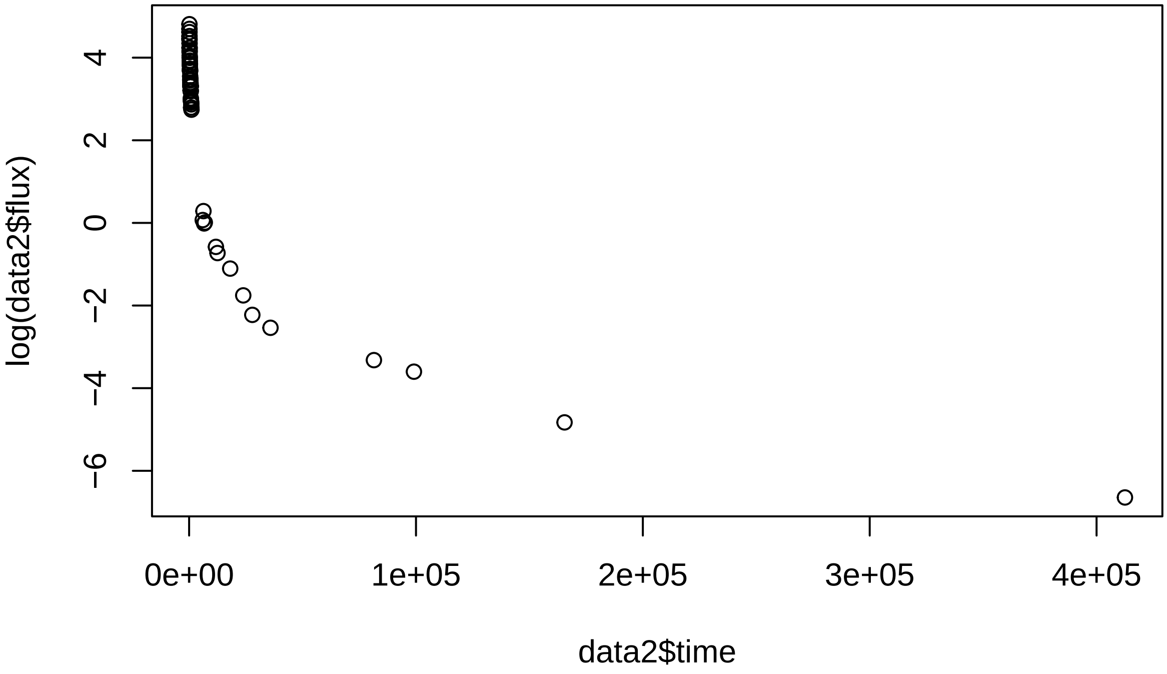
λ value of zero suggests that a log transformation of the response variable is required.

```
# Fitting the previous model with log transformation of response
model_2log = lm(log(flux) ~ time, data = data2, weights = 1 / error)
summary(model_2log)
```

```
##
## Call:
## lm(formula = log(flux) ~ time, data = data2, weights = 1/error)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -18.917   3.192   3.959   4.503   7.683
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.269e+00  2.111e-01  -10.75 1.03e-15 ***
## time        -1.089e-05  6.360e-07  -17.12  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.929 on 61 degrees of freedom
## Multiple R-squared:  0.8278, Adjusted R-squared:  0.825
## F-statistic: 293.2 on 1 and 61 DF,  p-value: < 2.2e-16
```

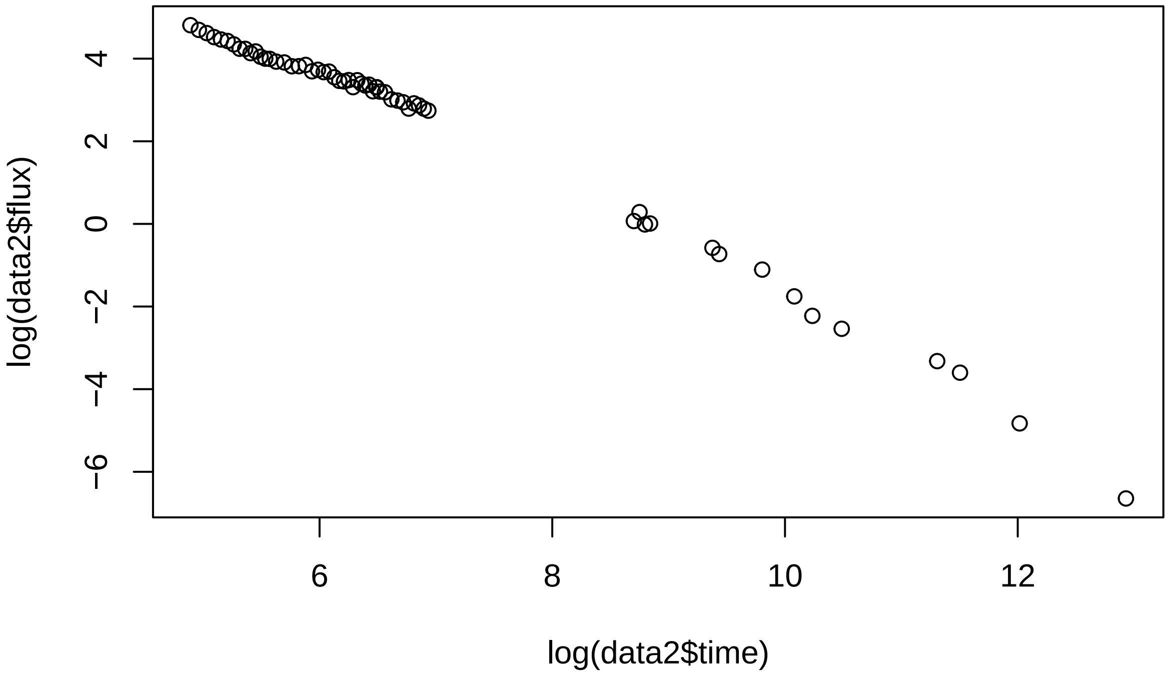
We can see a great improvement in the value of R-squared. The R-squared value has increased from 0.03018 to 0.8278 with the log transformation of the response. Now going ahead to check for the transformations required for predictor.

```
# Plotting predictor(time) with the log transformation of response
plot(data2$time, log(data2$flux))
```



From the plot above plot of Predictor 'time' vs log transformation of response, we can see that it might be good idea to take log of the predictor variable for a getting a better linear relationship between the two.

```
# Plotting the log transformation of the predictor with the log transformation of the response
plot(log(data2$time), log(data2$flux))
```



Since we can see a strong linear relationship between the log transformation of the predictor (time) with the log transformation of the response(flux), we go ahead with fitting a linear model between these two.

```
# Fitting a model with log transformation of predictor with log transformation of response
model_2logpred = lm(log(flux) ~ log(time),
                    data = data2 ,
                    weights = 1 / error)
summary(model_2logpred)
```

```
##
## Call:
## lm(formula = log(flux) ~ log(time), data = data2, weights = 1/error)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2948 -0.9795 -0.9323 -0.7641  6.0373
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 15.27506    0.35397   43.15   <2e-16 ***
## log(time)   -1.68718    0.02864  -58.91   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.561 on 61 degrees of freedom
## Multiple R-squared:  0.9827, Adjusted R-squared:  0.9824
## F-statistic:  3471 on 1 and 61 DF,  p-value: < 2.2e-16
```

We can see that the R-squared value for the model has now increased to 0.9827 after taking log transformation of the predictor as well.

Now we go fit the model weights by a stronger factor of 1/error^2.

```r
# Fitting the previous model by changing weights to 1/error^2
model_2error2 = lm(log(flux) ~ log(time),
                   data = data2 ,
                   weights = 1 / error ^ 2)
summary(model_2error2)
```

```
##
## Call:
## lm(formula = log(flux) ~ log(time), data = data2, weights = 1/error^2)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -33.867  -2.253  -1.447  -1.028  29.548
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 18.84987    0.18776   100.4   <2e-16 ***
## log(time)   -1.97161    0.01466  -134.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.081 on 61 degrees of freedom
## Multiple R-squared:  0.9966, Adjusted R-squared:  0.9966
## F-statistic: 1.808e+04 on 1 and 61 DF,  p-value: < 2.2e-16
```

The R-squared value for the model has now increased even more to 0.9966 after taking weights= 1/ error^2 following log transformations of the predictor and response variable.

Since with the above model, we have got the highest value of R-square we conclude that it is the most suitable model in comparison with the other fitted models for modeling the relationship in the data provided.

Problem 3

For the prostate data, fit a model with lpsa as the response and the other variables as predictors.

```r
# Fitting a model with lpsa as response and all other variables as predictors
data3 = prostate
model3 = lm(lpsa ~ ., data = data3)
summary(model3)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = data3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7331 -0.3713 -0.0170  0.4141  1.6381
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.669337   1.296387   0.516  0.60693
## lcavol       0.587022   0.087920   6.677 2.11e-09 ***
## lweight      0.454467   0.170012   2.673  0.00896 **
## age         -0.019637   0.011173  -1.758  0.08229 .
## lbph         0.107054   0.058449   1.832  0.07040 .
## svi          0.766157   0.244309   3.136  0.00233 **
## lcp         -0.105474   0.091013  -1.159  0.24964
## gleason      0.045142   0.157465   0.287  0.77503
## pgg45        0.004525   0.004421   1.024  0.30886
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7084 on 88 degrees of freedom
## Multiple R-squared:  0.6548, Adjusted R-squared:  0.6234
## F-statistic: 20.86 on 8 and 88 DF,  p-value: < 2.2e-16
```

(a) Compute and comment on the correlation between predictors

```
# checking pairwise correlation
round(cor(data3[1:8]), dig = 2)
```

```
##         lcavol lweight  age  lbph   svi   lcp gleason pgg45
## lcavol    1.00    0.19 0.22  0.03  0.54  0.68    0.43  0.43
## lweight   0.19    1.00 0.31  0.43  0.11  0.10    0.00  0.05
## age       0.22    0.31 1.00  0.35  0.12  0.13    0.27  0.28
## lbph      0.03    0.43 0.35  1.00 -0.09 -0.01    0.08  0.08
## svi       0.54    0.11 0.12 -0.09  1.00  0.67    0.32  0.46
## lcp       0.68    0.10 0.13 -0.01  0.67  1.00    0.51  0.63
## gleason   0.43    0.00 0.27  0.08  0.32  0.51    1.00  0.75
## pgg45     0.43    0.05 0.28  0.08  0.46  0.63    0.75  1.00
```

The highest correlation values are for:

1) pgg45 and gleason = 0.75. This means 'pgg45' and 'gleason' have strong positive linear relationship between them.
2) lcp and lcavol = 0.68, lcp and svi = 0.67, lcp and pgg445 = 0.63. This means 'lcp' has a moderate-to-strong positive relationship with 'lcavol', 'svi' and 'pgg445'.

Other variates do not seem to have any significant positive or negative relationship with each other. To understand and verify claims about collinearity between the predictor, let us go ahead to perform statistical tests.

(b) Compute and comment on the Condition number

```
# Standardize matrix
x = model.matrix(model3)[, -1]
x = x - matrix(apply(x, 2, mean), 97, 8, byrow = TRUE)
x = x / matrix(apply(x, 2, sd), 97, 8, byrow = TRUE)
```

Since the Condition Number is scale-dependent, hence the columns of x have been standardized.

```
e = eigen(t(x) %*% x)

# As eigen values are sorted in descending order, we take the max and min from the list
largest_eigen = e$val[1]
smallest_eigen = tail(e$val, n = 1)

k = sqrt(largest_eigen / smallest_eigen)
k
```

```
## [1] 4.11621
```

As per the empirical rule for declaring collinearity, κ>=30. In the case above, κ value is not greater than 30, hence there in no collinearity amongst the predictors.

(c) Compute and comment on the variance inflation factors.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following objects are masked from 'package:faraway':
##
##     logit, vif
```

```
car::vif(model3)
```

```
##   lcavol  lweight      age     lbph      svi      lcp  gleason    pgg45
## 2.054115 1.363704 1.323599 1.375534 1.956881 3.097954 2.473411 2.974361
```

Since the vif is not greater than 4 for any variable, we conclude that the variables do not have collinearity.

Problem 4

Use the cheddar data for this question

(a) Fit an additive model for taste as a response, with the other three variables as predictors. Is any transformation of the predictors suggested? Justify your answer.

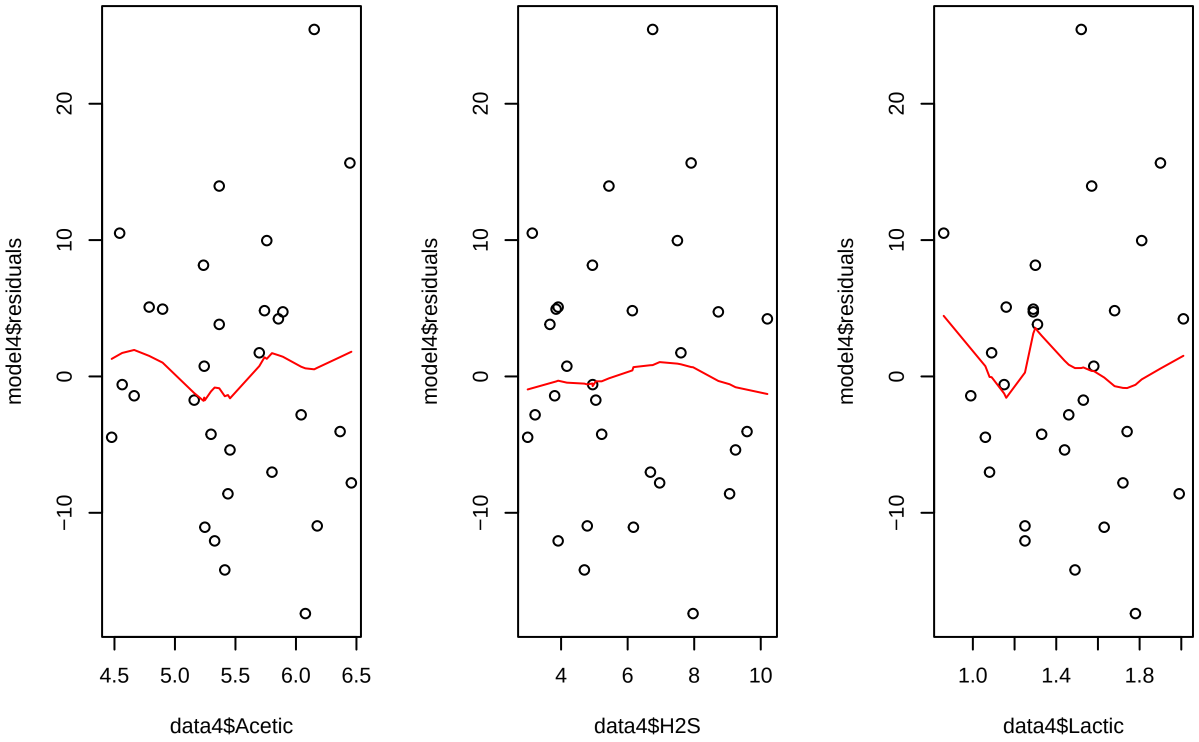
```
data4 = cheddar
# Fitting model for taste as response and other variables as predictors
model4 = lm(taste ~ ., data4)
summary(model4)
```

```
##
## Call:
## lm(formula = taste ~ ., data = data4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.390  -6.612  -1.009   4.908  25.449
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -28.8768    19.7354  -1.463  0.15540
## Acetic        0.3277     4.4598   0.073  0.94198
## H2S           3.9118     1.2484   3.133  0.00425 **
## Lactic       19.6705     8.6291   2.280  0.03108 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.13 on 26 degrees of freedom
## Multiple R-squared:  0.6518, Adjusted R-squared:  0.6116
## F-statistic: 16.22 on 3 and 26 DF,  p-value: 3.81e-06
```

The R-squared value of the model is 0.6518. We might want to try doing transformations on the predictors to create a better model.

We plot each predictor variable with model residuals in order to see if there any pattern in order to apply tranformations.

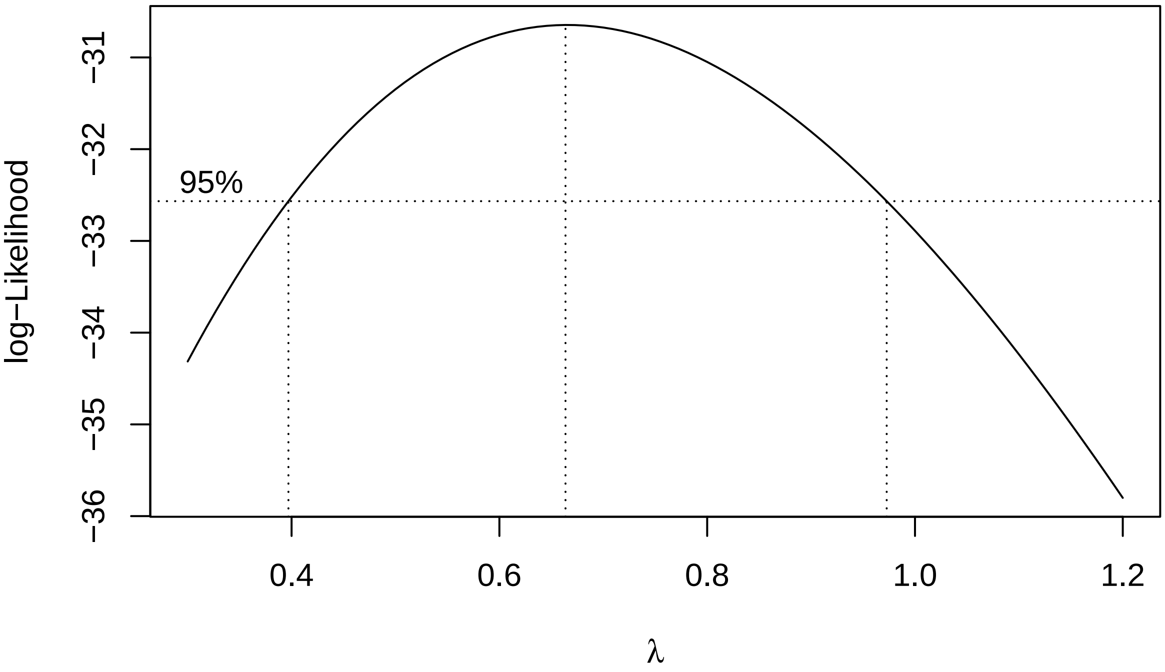
```r
# Plotting each predictor variable with model residuals
par(mfrow = c(1, 3))

plot(data4$Acetic, model4$residuals)
lines(supsmu(data4$Acetic, model4$residuals), col = "red")

plot(data4$H2S, model4$residuals)
lines(supsmu(data4$H2S, model4$residuals), col = "red")

plot(data4$Lactic, model4$residuals)
lines(supsmu(data4$Lactic, model4$residuals), col = "red")
```



Since the there is no significant trend/pattern in the predictor vs model residuals plot for each of the 3 predictors, we can conclude that there is no requirement of performing transformations on the predictors.

(b) Use the Box-Cox method to determine an optimal transformation of the response. Would it be reasonable to leave the response untransformed?

```
#boxcox(model4, se)
boxcox(model4, plotit = TRUE, lambda = seq(0.3, 1.2, by = 0.1))
```



Since 1 lies outside the confidence interval, hence Box-Cox suggests a transformation on the response variable.

(c) Use the optimal transformation of the response and refit the additive model. Do these new results make any difference to the transformations suggested for the predictors in part a)?

From the above box-cox transformation, we can take the value of λ=0.667 for transforming the reponse variable.

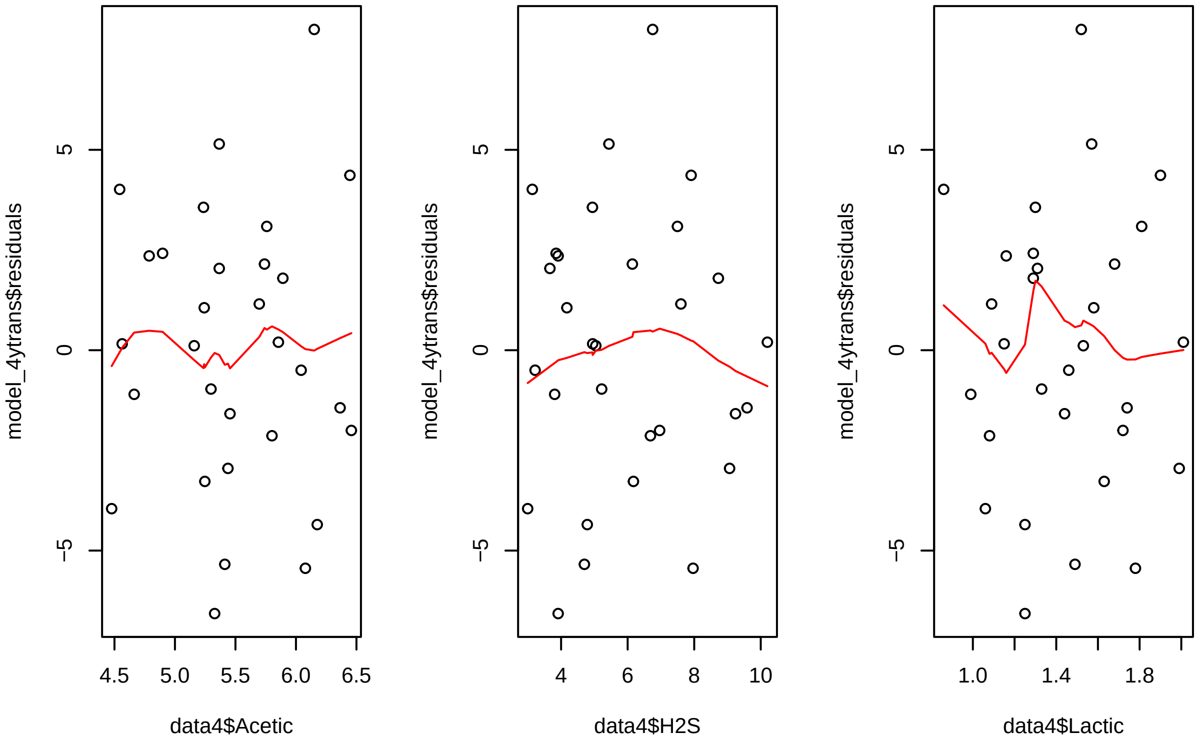
```
# Fitting the model with transformed response
model_4ytrans = lm((taste ^ 0.667 - 1) / 0.667 ~ ., data = data4)
summary(model_4ytrans)
```

```
##
## Call:
## lm(formula = (taste^0.667 - 1)/0.667 ~ ., data = data4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5716 -2.1018  0.1353  2.3017  8.0036
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.92001    7.09331  -1.117  0.27440
## Acetic       0.01459    1.60293   0.009  0.99281
## H2S          1.43940    0.44871   3.208  0.00353 **
## Lactic       6.77246    3.10146   2.184  0.03820 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.641 on 26 degrees of freedom
## Multiple R-squared:  0.6468, Adjusted R-squared:  0.606
## F-statistic: 15.87 on 3 and 26 DF,  p-value: 4.572e-06
```

The R-squared for the model is 0.6468.

We plot each predictor variable with model residuals in order to see if there any pattern in order to apply tranformations.

```
# Plotting each predictor variable with model residuals
par(mfrow = c(1, 3))

plot(data4$Acetic, model_4ytrans$residuals)
lines(supsmu(data4$Acetic, model_4ytrans$residuals), col = "red")

plot(data4$H2S, model_4ytrans$residuals)
lines(supsmu(data4$H2S, model_4ytrans$residuals), col = "red")

plot(data4$Lactic, model_4ytrans$residuals)
lines(supsmu(data4$Lactic, model_4ytrans$residuals), col = "red")
```



Since the there is no significant trend/pattern in the predictor vs model residuals plot for each of the 3 predictors, we can conclude that there is no requirement of performing transformations on the predictors.

Hence, the new results also do not make any difference to the transformations suggested for the predictors in part a).