413_HW4

Yanzhuo Cao

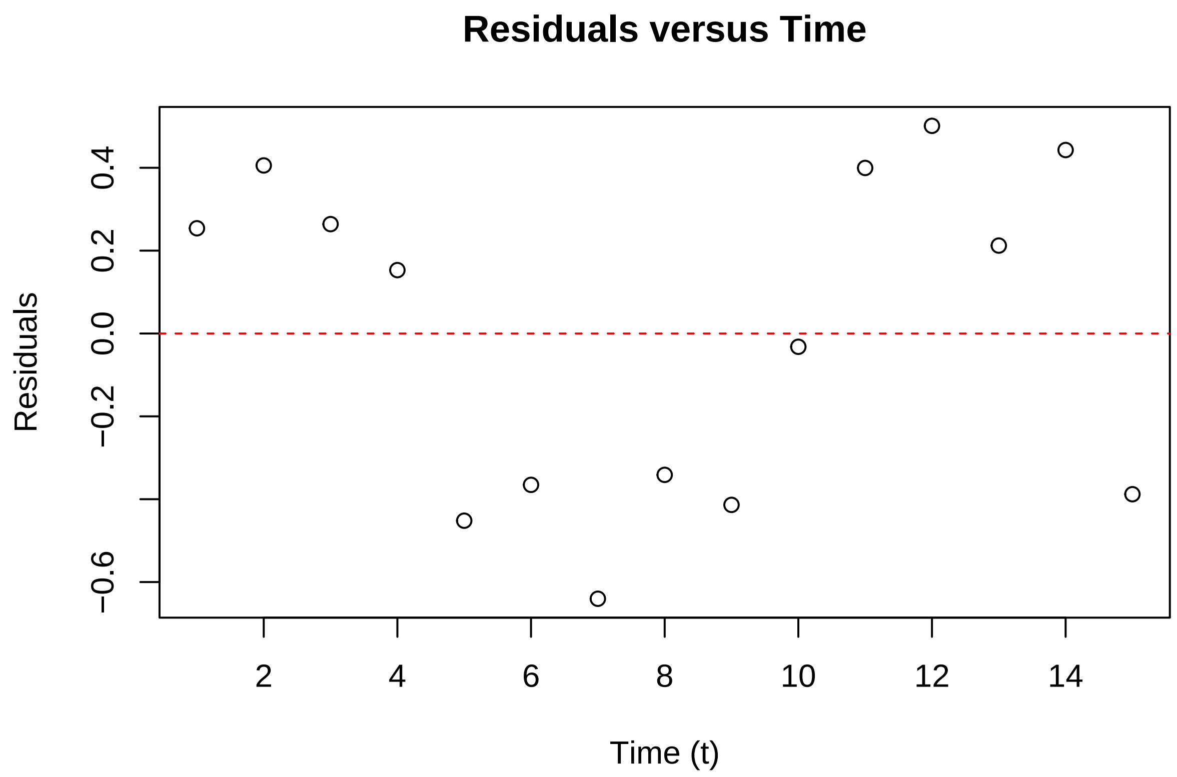
2023-07-19

Step 1: Fit the linear regression model and Plot the residuals versus time

```
model <- lm(yt ~ xt, data = data)
summary(model)
```

```
##
## Call:
## lm(formula = yt ~ xt, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6403 -0.3767  0.1530  0.3318  0.5012
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.59405    1.20560   20.40 2.96e-11 ***
## xt          -0.08918    0.01368   -6.52 1.94e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4077 on 13 degrees of freedom
## Multiple R-squared:  0.7658, Adjusted R-squared:  0.7478
## F-statistic: 42.51 on 1 and 13 DF,  p-value: 1.943e-05
```

```
residuals <- residuals(model)

plot(data$t, residuals, main = "Residuals versus Time",
     xlab = "Time (t)", ylab = "Residuals")
abline(h = 0, col = "red", lty = 2)
```



From the plot we can see that the residuals are not randomly distributed around 0, since it almost follows a period of 4. During [1,5] it is positive, [5,9] negative, and [9,14] positive, etc. Therefore, we can say that there is indication of autocorrelation.

Step 2: Perform the Durbin-Watson test

```
dw_test <- dwtest(model)

cat("Durbin-Watson Statistic:", dw_test$statistic, "\n")
```

```
## Durbin-Watson Statistic: 0.8182972
```

```
cat("Durbin-Watson p-value:", dw_test$p.value, "\n")
```

```
## Durbin-Watson p-value: 0.00156337
```

```
print(dw_test$p.value)
```

```
## [1] 0.00156337
```

```
if (dw_test$p.value < 0.05) {
  cat("Exists autocorrelation.\n")
} else {
  cat("No autocorrelation.\n")
}
```

```
## Exists autocorrelation.
```

Since the result shows that the D-W Statistic value is much less than 2 and p-value is much less that 0.05, we can say that there is positive autocorrelation in the errors.

Step 3: Fit an autoregressive model to the residuals (one iteration of Cochrane-Orcutt)

```r
cochrane_orcutt_model <- cochrane.orcutt(model)
summary(cochrane_orcutt_model)
```

```
## Call:
## lm(formula = yt ~ xt, data = data)
##
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 26.611277   1.113022  23.909 1.719e-11 ***
## xt          -0.115793   0.012955  -8.938 1.188e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2999 on 12 degrees of freedom
## Multiple R-squared:  0.8694 ,  Adjusted R-squared:  0.8585
## F-statistic: 79.9 on 1 and 12 DF,  p-value: < 1.188e-06
##
## Durbin-Watson statistic
## (original):    0.81830 , p-value: 1.563e-03
## (transformed): 0.85205 , p-value: 1.243e-02
```

From the table we can see that the standard error of xt is 0.012955 and the standard error of intercept is 1.113022.

Step 4: Conclusion

```r
dw_test_iteration1 <- dwtest(cochrane_orcutt_model)

cat("Durbin-Watson Statistic after Iteration 1:", dw_test_iteration1$statistic, "\n")
```

```
## Durbin-Watson Statistic after Iteration 1: 0.8520489
```

```r
cat("Durbin-Watson p-value after Iteration 1:", dw_test_iteration1$p.value, "\n")
```

```
## Durbin-Watson p-value after Iteration 1: 0.01242933
```

```r
if (dw_test_iteration1$p.value < 0.05) {
  cat("Still autocorrelation.\n")
} else {
  cat("No autocorrelation after one iteration.\n")
}
```

```
## Still autocorrelation.
```

```r
cat("If the iteration is successful: ")
```

```
## If the iteration is successful:
```

```r
if (dw_test_iteration1$p.value < 0.05) {
  cat("Unsuccessful\n")
} else {
  cat("Successful\n")
}
```

```
## Unsuccessful
```

413_HW4_Q2

Yanzhuo Cao

2023-07-19

```r
# a. Fit a logistic regression model
logit_model <- glm(cbind(redeem, size - redeem) ~ discount, data = data, family = binomial)
summary(logit_model)
```

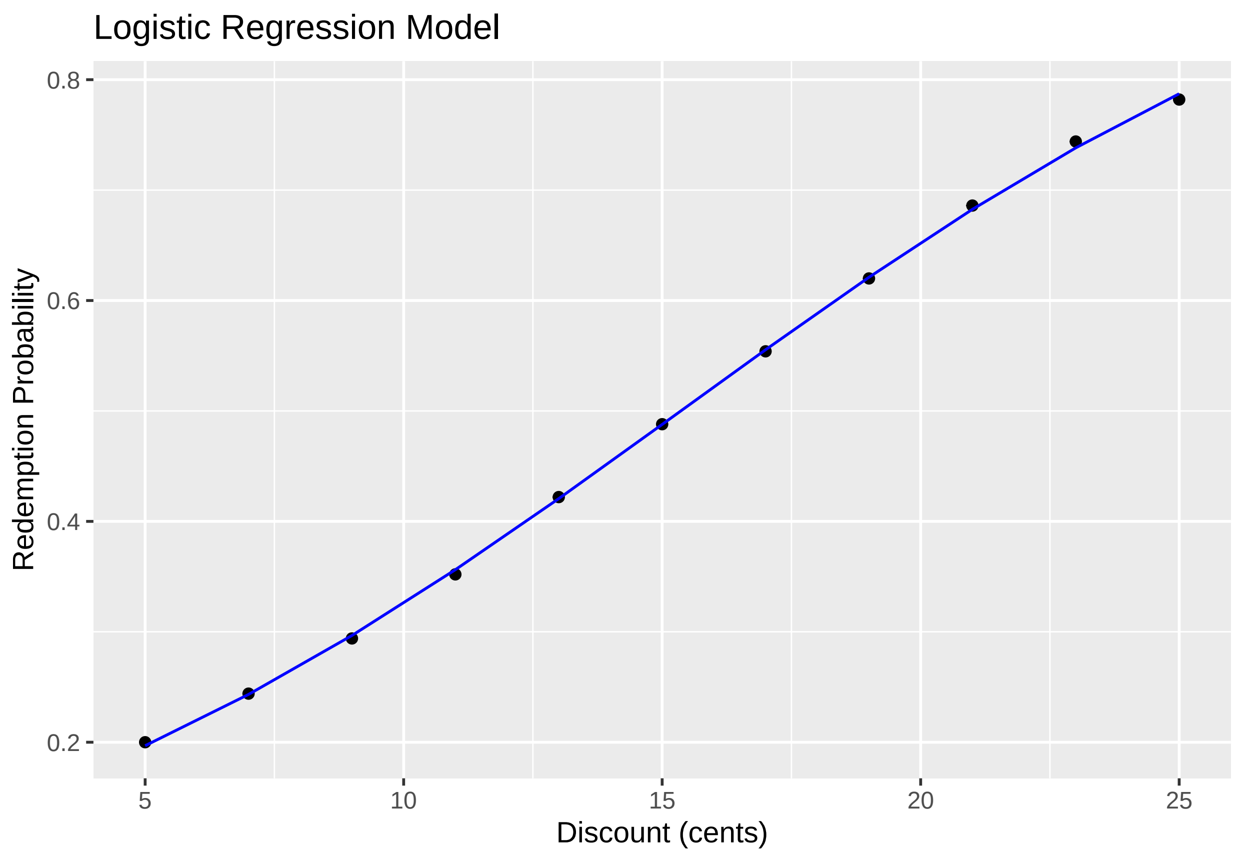
```
##
## Call:
## glm(formula = cbind(redeem, size - redeem) ~ discount, family = binomial,
##     data = data)
##
## Deviance Residuals:
##       Min         1Q     Median         3Q        Max
## -0.286011  -0.097431   0.009331   0.114557   0.292382
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.084754   0.080398  -25.93   <2e-16 ***
## discount     0.135727   0.004957   27.38   <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: 871.21962  on 10  degrees of freedom
## Residual deviance:   0.29426  on  9  degrees of freedom
## AIC: 75.708
##
## Number of Fisher Scoring iterations: 3
```

From the result we can get that Number_Redeemed = 0.135727*discount - 2.084754

```r
# b. Test model adequacy with deviance
model_summary <- tidy(logit_model)
deviance <- logit_model$deviance
df_resid <- logit_model$df.residual

# Deviance test
chi2_p_value <- pchisq(deviance, df_resid, lower.tail = FALSE)

# Perform deviance test
chi2_p_value <- pchisq(deviance, df_resid, lower.tail = FALSE)

# Check if the model is adequate
alpha = 0.05 # Significance level (e.g., 0.05 for 95% confidence)
if (chi2_p_value < alpha) {
  cat("The logistic regression model is adequate at the", (1 - alpha) * 100, "% confidence level.\n")
} else {
  cat("The logistic regression model is not adequate at the", (1 - alpha) * 100, "% confidence level.\n
}
```

```
## The logistic regression model is not adequate at the 95 % confidence level.
```

```r
# c. Draw a graph of the data and the fitted logistic regression model
plot_data <- data %>%
  mutate(prob = redeem / size)

ggplot(plot_data, aes(discount, prob)) +
  geom_point() +
  geom_line(aes(y = fitted(logit_model)), color = "blue") +
  labs(title = "Logistic Regression Model",
       x = "Discount (cents)",
       y = "Redemption Probability")
```



```r
# d. Expand the linear predictor to include a quadratic term
logit_model_quad <- glm(cbind(redeem, size - redeem) ~ discount + I(discount^2), data = data, family = 

summary_linear <- summary(logit_model)
summary_quad <- summary(logit_model_quad)

cat("Linear Model AIC:", summary_linear$aic, "\n")
```

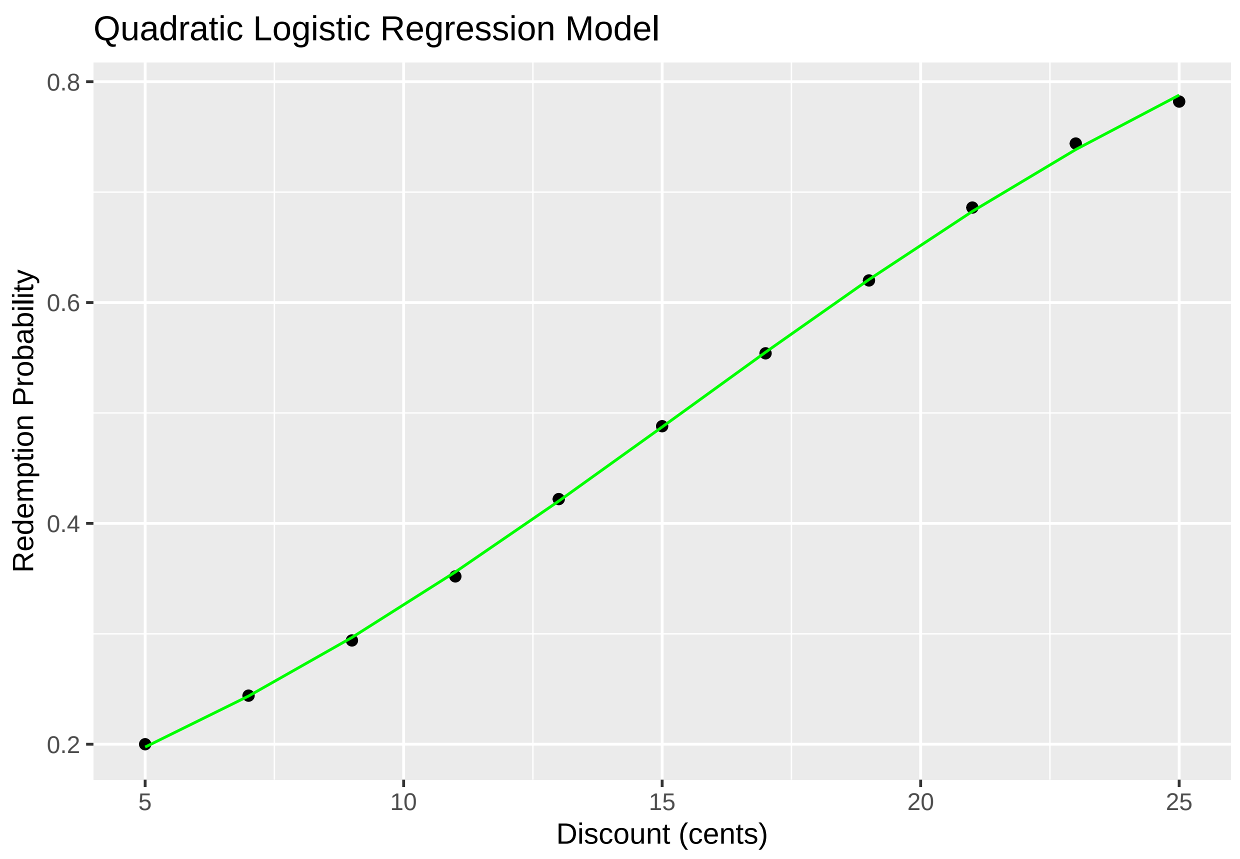
```
## Linear Model AIC: 75.70828
```

```r
cat("Quadratic Model AIC:", summary_quad$aic, "\n")
```

```
## Quadratic Model AIC: 77.70429
```

From the result we can see that the AIC of Quadratic Model is not far more than that of Linear Model, therefore we cannot say that it is evidence to include quadratic terms.

```r
# e. Draw a graph of the expanded model
plot_data$fit_quad <- fitted(logit_model_quad)

ggplot(plot_data, aes(discount, prob)) +
  geom_point() +
  geom_line(aes(y = fit_quad), color = "green") +
  labs(title = "Quadratic Logistic Regression Model",
       x = "Discount (cents)",
       y = "Redemption Probability")
```



From the graph we can also conclude that the quadratic model isn't obviously better than the original model.

```r
# f. Find 95% confidence intervals on the model parameters for the quadratic logistic regression model
ci_quad_model <- confint(logit_model_quad, level = 0.95)
```

```
## Waiting for profiling to be done...
```

```r
print(ci_quad_model)
```

```
##                      2.5 %       97.5 %
## (Intercept)   -2.439842485 -1.714298381
## discount       0.081986267  0.186519548
## I(discount^2) -0.001652178  0.001761554
```

413_HW4_Q3_cyz

Yanzhuo Cao

2023-07-19

Fit a Poisson regression model with a log link

```
poisson_model <- glm(frac ~ inb + extrp + seamh + time, data = mine, family = poisson(link = "log"))
# Output model summary
summary(poisson_model)
```

```
##
## Call:
## glm(formula = frac ~ inb + extrp + seamh + time, family = poisson(link = "log"),
##     data = mine)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.78962  -0.85988  -0.04893   0.37313   2.16201
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.5930896  1.0256803  -3.503  0.00046 ***
## inb         -0.0014066  0.0008358  -1.683  0.09240 .
## extrp        0.0623458  0.0122862   5.074 3.89e-07 ***
## seamh       -0.0020803  0.0050661  -0.411  0.68134
## time        -0.0308135  0.0162648  -1.894  0.05816 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 74.984  on 43  degrees of freedom
## Residual deviance: 37.856  on 39  degrees of freedom
## AIC: 144.13
##
## Number of Fisher Scoring iterations: 5
```

```
# Perform Chi-squared test
deviance_test <- anova(poisson_model, test = "Chisq")
deviance_test
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: frac
##
## Terms added sequentially (first to last)
```

```
##
##
##       Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                     43     74.984
## inb    1   3.1440        42     71.840   0.07621 .
## extrp  1  29.7460        41     42.094 4.925e-08 ***
## seamh  1   0.3433        40     41.750   0.55795
## time   1   3.8944        39     37.856   0.04845 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

According to the table's results:

The p-value for the "inb" variable is 0.07621, which is greater than 0.05, indicating that the coefficient for "inb" may not be statistically significant. The p-value for the "seamh" variable is 0.55795, which is greater than 0.05, suggesting that the coefficient for "seamh" may not be statistically significant. The p-value for the "time" variable is 0.04845, which is very close to 0.05, indicating that the coefficient for "time" may not be statistically significant. In conclusion, based on the model's fitting results and the p-values of the predictor variables, we can say that the model is not entirely satisfactory from a statistical perspective.

Find approximate 95% Wald confidence intervals on the model parameters

```r
# Extract the estimated values of model parameters
model_parameters <- coef(poisson_model)
# Calculate the standard errors of the parameters
parameter_standard_errors <- sqrt(diag(vcov(poisson_model)))
# Calculate Wald confidence intervals
lower_ci <- model_parameters - 1.96 * parameter_standard_errors
upper_ci <- model_parameters + 1.96 * parameter_standard_errors
# Combine the results into a data frame
conf_interval <- data.frame(Lower_CI = lower_ci, Upper_CI = upper_ci)
rownames(conf_interval) <- names(model_parameters)
conf_interval
```

```
##                 Lower_CI     Upper_CI
## (Intercept) -5.603422873 -1.582756283
## inb         -0.003044787  0.000231611
## extrp        0.038264806  0.086426715
## seamh       -0.012009984  0.007849300
## time        -0.062692541  0.001065556
```

From part (a) we can find that whose value of Pr(>Chi) is acceptable are inb, extrp and time. Therefore, we do the regression again based on these variables.

```r
# Suppose we think "seamh" variable might be unimportant, so we refit the model
reduced_poisson_model <- glm(frac ~ inb + extrp + time, data = mine, family = poisson(link = "log"))
# Output model summary and compare AIC and BIC
summary(reduced_poisson_model)
```

```
##
## Call:
## glm(formula = frac ~ inb + extrp + time, family = poisson(link = "log"),
##     data = mine)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7727  -0.9073  -0.0107   0.2716   2.1783
##
```

```
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -3.7206821  0.9788770  -3.801 0.000144 ***
## inb         -0.0014793  0.0008244  -1.794 0.072757 .  
## extrp        0.0627011  0.0122711   5.110 3.23e-07 ***
## time        -0.0316514  0.0163095  -1.941 0.052298 .  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 74.984  on 43  degrees of freedom
## Residual deviance: 38.031  on 40  degrees of freedom
## AIC: 142.3
##
## Number of Fisher Scoring iterations: 5
```

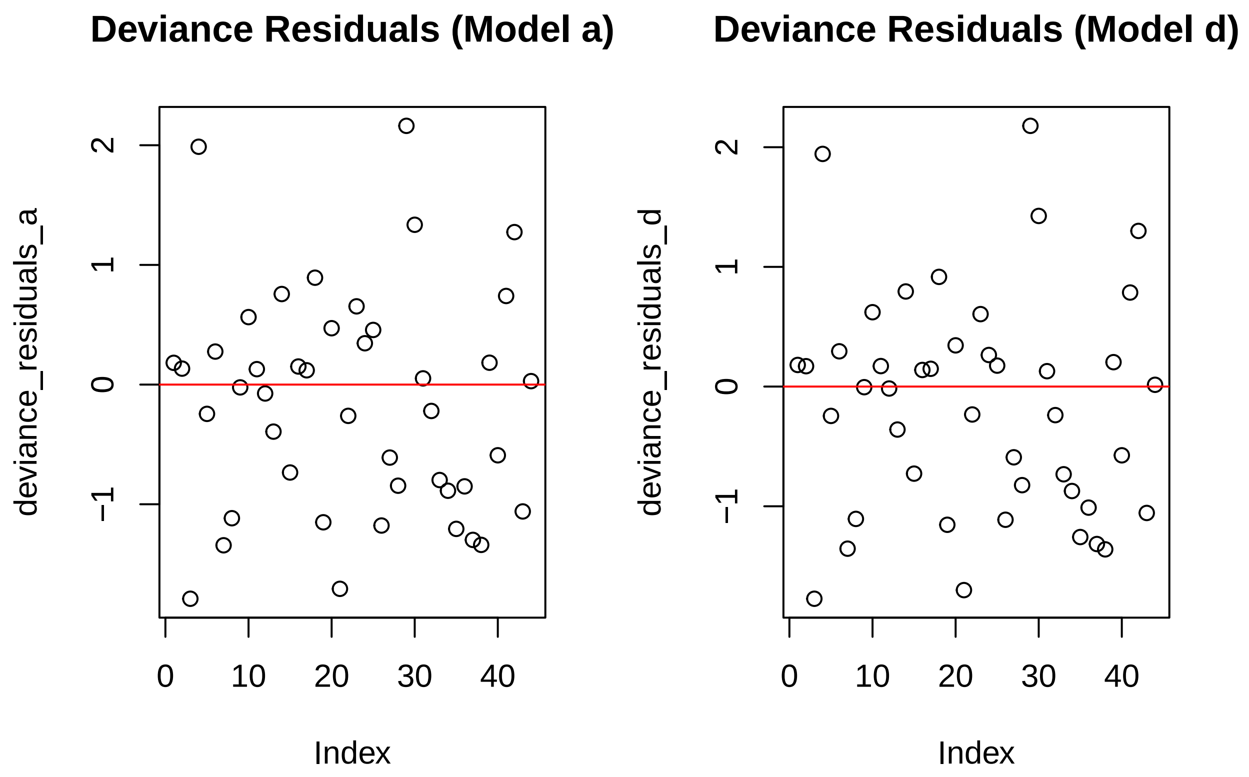
```r
AIC(reduced_poisson_model)
```

```
## [1] 142.3023
```

```r
BIC(reduced_poisson_model)
```

```
## [1] 149.4391
```

```r
# Extract deviance residuals from models
deviance_residuals_a <- residuals(poisson_model, type = "deviance")
deviance_residuals_d <- residuals(reduced_poisson_model, type = "deviance")
# Plot deviance residuals for model a
par(mfrow = c(1, 2)) # Arrange plots in 1 row and 2 columns
# Deviance residuals plot for model a
plot(deviance_residuals_a, main = "Deviance Residuals (Model a)")
abline(h = 0, col = "red") # Add horizontal reference line
# Deviance residuals plot for model d
plot(deviance_residuals_d, main = "Deviance Residuals (Model d)")
abline(h = 0, col = "red") # Add horizontal reference line
```



```r
par(mfrow = c(1, 1)) # Restore default plot layout
```

From the residual plot we can see that there is no significant difference between the results we get from part (a) and part (d).