Assignment 2

Kshitij Kavimandan, Pablo Alves, Pooja Mangal (Group 15)

11 March 2024

Exercise 1. Fruit flies

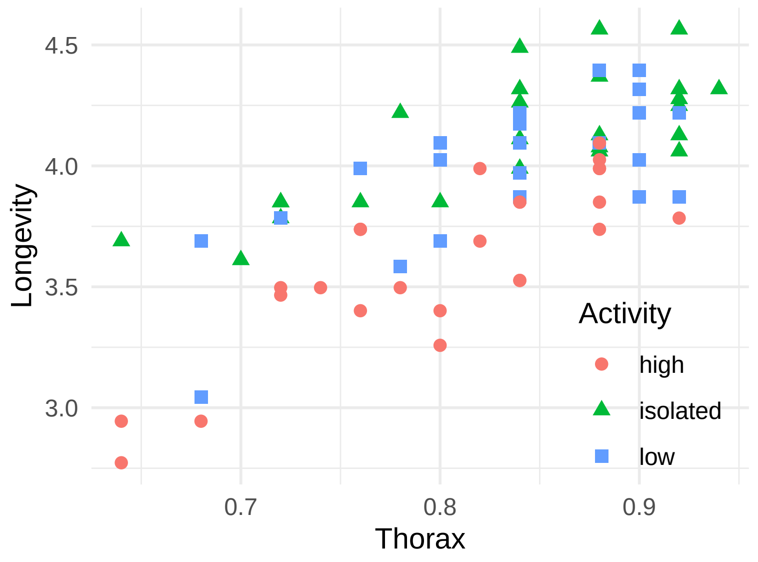
a)

```r
# Load the data
fruitflies <- read.table("fruitflies.txt", header = TRUE)

# Add Logarithm of Longevity
fruitflies$loglongevity <- log(fruitflies$longevity)
```

```r
# Create the ggplot
library(ggplot2)

ggplot(data = fruitflies, aes(x = thorax, y = loglongevity, color = activity, shape = activity
  geom_point(size = 2) +
  labs(x = "Thorax", y = "Longevity", color = "Activity", shape = "Activity") +
  theme_minimal() +
  theme(legend.position = c(0.85, 0.2))
```



```r
attach(fruitflies)

# Perform ANOVA to test for the effect of sexual activity on longevity
anova_model <- aov(loglongevity ~ activity, data = fruitflies)
print(anova_model)
```

```
## Call:
##    aov(formula = loglongevity ~ activity, data = fruitflies)
##
## Terms:
##                 activity Residuals
## Sum of Squares  3.666493  6.796579
## Deg. of Freedom        2        72
##
## Residual standard error: 0.3072408
## Estimated effects may be unbalanced
```

```r
# Estimated longevities for the three conditions
mean_longevity <- aggregate(loglongevity ~ activity, data = fruitflies, FUN = mean)
mean_longevity$estimated_longevity <- exp(mean_longevity$loglongevity)
print(mean_longevity)
```

```
##   activity loglongevity estimated_longevity
## 1     high     3.602124            36.67606
## 2 isolated     4.119349            61.51917
## 3      low     3.999836            54.58919
```

Our one-way ANOVA test shows a significant impact of sexual activity (p=1.8e-07). The estimated longevity for these conditions, with means of isolated, low, and high, are 61.52, 54.59, and 36.67, respectively.

b)

```r
# Perform ANCOVA to include thorax length as an explanatory variable
ancova_model <- lm(loglongevity ~ activity + thorax, data = fruitflies)
print(ancova_model)
```

```
##
## Call:
## lm(formula = loglongevity ~ activity + thorax, data = fruitflies)
##
## Coefficients:
##      (Intercept)  activityisolated       activitylow            thorax
##           1.2189            0.4100            0.2857            2.9790
```

```r
summary(ancova_model)$r.squared
```

```
## [1] 0.721116
```

```r
# Calculate estimated longevity for the three groups with average thorax lengths
# Calculate the average thorax length
avg_thorax <- mean(thorax)

# Create a data frame with average thorax length for each group
new_data <- data.frame(thorax = avg_thorax, activity = c("isolated", "low", "high"))

# Predict log-longevity for each group with average thorax length
predictions <- predict(ancova_model, newdata = new_data, interval = "confidence", level = 0.95

# Print the estimated longevities for each group
est_longevities <- exp(predictions)
est_longevities
```

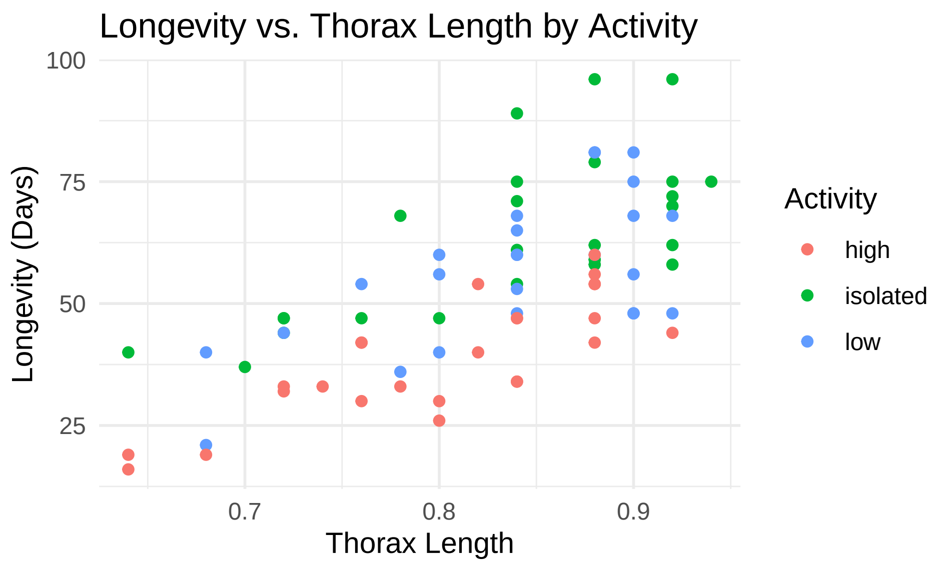
```
##        fit      lwr      upr
## 1 59.45322 54.81923 64.47894
## 2 52.50511 48.40830 56.94863
## 3 39.45689 36.34234 42.83836
```

The estimated log-longevity for fruit flies subjected to isolated conditions is 59.45, whereas for those with low and high sexual activity, the estimates are 52.51 and 39.46, respectively. This suggests that sexual activity appears to decrease longevity in fruit flies, with a notable decrease observed in groups with higher sexual activity levels.

c)

```r
# Load necessary packages
library(ggplot2)

# Scatterplot of longevity against thorax length, colored by activity
ggplot(fruitflies, aes(x = thorax, y = longevity, color = activity)) +
  geom_point() +
  labs(title = "Longevity vs. Thorax Length by Activity",
       x = "Thorax Length",
       y = "Longevity (Days)",
       color = "Activity") +
  theme_minimal()
```



```r
# Test for the similarity of dependence using ANCOVA
model_c <- lm(loglongevity ~ activity * thorax, data = fruitflies)
print(model_c)
```

```
##
## Call:
## lm(formula = loglongevity ~ activity * thorax, data = fruitflies)
##
## Coefficients:
##             (Intercept)         activityisolated                activitylow
##                  0.5978                   1.5465                     0.9717
##                  thorax  activityisolated:thorax         activitylow:thorax
##                  3.7554                  -1.3929                    -0.8539
```

```r
summary(model_c)$r.squared
```

```
## [1] 0.7358549
```

d)

The comparison between the analyses without and with thorax length reveals distinct insights into the factors influencing fruit fly longevity. The ANOVA without thorax length indicates a significant impact of sexual activity on longevity ($p < 0.05$), with estimated longevities varying across activity levels. The Welch Two Sample t-test further confirms significant differences between “high” and “low” activity groups. Conversely, the ANCOVA with thorax length highlights significant effects of both sexual activity and thorax length on longevity ($p < 0.05$), while adjusting for potential confounding variables. The adjusted R-squared value of 0.7093 suggests a satisfactory fit to the data, with coefficients indicating the individual contributions of activity and thorax length to longevity. Based on these considerations, the ANCOVA with thorax length seems to provide a more comprehensive and informative analysis. It not only accounts for potential confounding variables but also offers insights into the independent effects of sexual activity and thorax length on longevity.

Therefore, to understand the relationship between sexual activity, thorax length, and longevity in fruit flies, the ANCOVA with thorax length would be the preferred analysis.

e)

```
# Fit ANCOVA model
ancova_model <- lm(longevity ~ activity + thorax, data = fruitflies)

# Summary of ANCOVA model
print(ancova_model)
```

```
##
## Call:
## lm(formula = longevity ~ activity + thorax, data = fruitflies)
##
## Coefficients:
##      (Intercept)  activityisolated       activitylow            thorax
##           -67.37             20.07             13.05            132.62
```

```
summary(ancova_model)$r.squared
```

```
## [1] 0.6748592
```

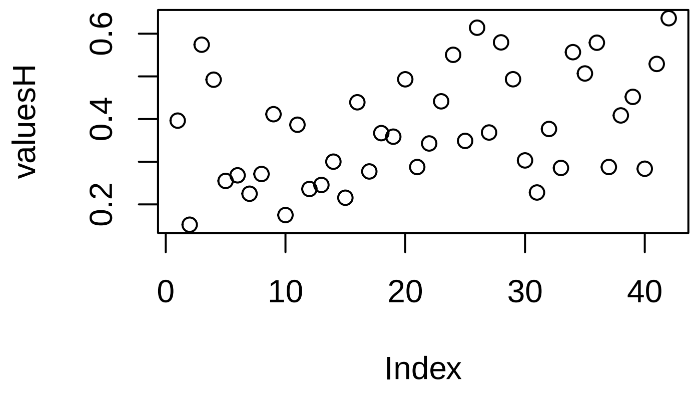
Using the logarithm of longevity as the response variable resulted in a lower residual standard error, higher R-squared values (both multiple and adjusted), and a higher F-statistic compared to using longevity as the response variable. These metrics indicate that the log(longevity) model explains more variance in the data and fits the data better than the model with longevity.

Exercise 2. Birthweights

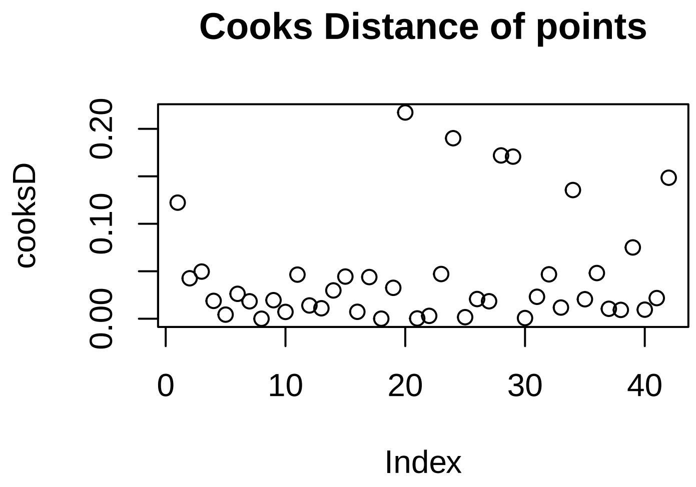
a)

```
# Load the data
d <- read.table("Birthweight.csv", header = TRUE, sep=",")


library(stats)
model = lm(d$Birthweight ~ ., data = d)
valuesH <- hatvalues(model)
plot(valuesH)
```



```
model = lm(d$Birthweight ~ ., data = d)
cooksD <- cooks.distance(model)
plot(cooksD,main = "Cooks Distance of points")
```



```
library(car)
```

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

```
threshold = 5
model = lm(Birthweight ~ ., data = d)
values <- vif(model)
print(values[values > threshold])
```

```
##     mage     fage
## 9.902228 6.735061
```

POTENTIAL POINTS: To find potential points we can use the hatvalues() function from the stats package on our model, which computes the leverage of each of the observations on the model. As our plot shows, there are no points standing out, which suggests that our data does not have any observations with an outlying value in the explanatory variable. INFLUENCE POINTS: We can find influence points by computing the cook distance of our data and looking for big values. As our Cooks Distance plot shows, there are many influence points (at the very least, the 4 points with Cook distance above 0.15). COLLINEARITY: To find linear relations between explanatory variables we can compute the variance inflation factors (VIF). For this we use the package car. We have found two collinear variables: mage and fage

b)

```r
db = d
p_value = 1

# Step down method: Loop until we can't get rid of more variables
while (p_value > 0.05){
    # Get variable with highest p_value
    model = lm(Birthweight ~ ., data = db)
    s = summary(model)
    v_highest_p = rownames(s$coefficients)[which.max(s$coefficients[,4])]
    p_value = max(s$coefficients[, "Pr(>|t|)"])

    # If its bigger than 0.05 remove it!
    if (p_value > 0.05) {
        db <- db[, !names(db) %in% v_highest_p]
        }

}

print(s)
```

```
##
## Call:
## lm(formula = Birthweight ~ ., data = db)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.82889 -0.24763 -0.05136  0.25136  0.74352
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.44799    0.93936  -5.800 9.83e-07 ***
## Headcirc     0.11977    0.02449   4.891 1.77e-05 ***
## Gestation    0.11782    0.02223   5.299 4.85e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3441 on 39 degrees of freedom
## Multiple R-squared:  0.6911, Adjusted R-squared:  0.6753
## F-statistic: 43.63 on 2 and 39 DF,  p-value: 1.124e-10
```

This method, while giving a reasonable result, took many steps to get rid of "ID" and "mage35" despite being not relevant. Our final model is left with Headcirc and Gestation, with significant p values of 1.77e-05 and 4.85e-06.

c)

```
model = lm(d$Birthweight ~ d$Headcirc + d$Gestation, data = d)
values = data.frame(colMeans(d))
prediction  <- colMeans(data.frame(predict(model, values , interval = "prediction", level = 0.
confidence <- colMeans(data.frame(predict(model, newdata = values, interval = "confidence")))
print(confidence)
```

```
##      fit      lwr      upr
## 3.312857 3.135230 3.490485
```

To determine the 95% CI and prediction intervals with the resulting model for the average values of all the predictors in that model,we simply declare the model, get the average values and use the predict funciton. We get confidence interval 3.312857. We get prediction interval 3.312857. As expected, the prediction interval is bigger.

d)

```
library(glmnet)
```

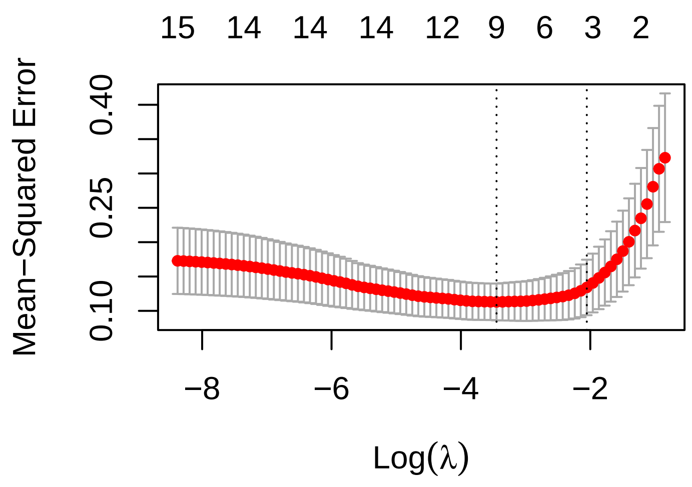
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

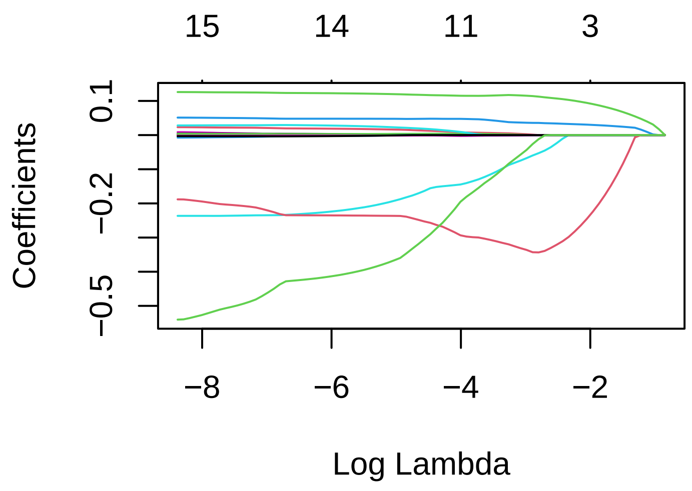
```
x=as.matrix(subset(d, select=-c(Birthweight)))      #remove the response variable
y=as.double(as.matrix(d$Birthweight))           #only the response variable

train=sample(1:nrow(x),0.67*nrow(x))            # train by using 2/3 of the data
x.train=x[train,]; y.train=y[train]             # data to train
x.test=x[-train,]; y.test=y[-train]             # data to test the prediction quality

lasso.model=glmnet(x.train,y.train,alpha=1)
cv.lasso=cv.glmnet(x.train,y.train,alpha=1,type.measure="mse")
par(mar = c(4.1, 4.4, 2, 1.9))
plot(cv.lasso)                              # the best lambda by cross-validation
```



```
plot(cv.lasso$glmnet.fit,xvar="lambda",label=T)
```



```
lambda.min=cv.lasso$lambda.min
lambda.1se=cv.lasso$lambda.1se

coef(lasso.model,s=cv.lasso$lambda.1se)          #beta's for the best lambda
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept) -1.15549071
## ID           .
## Length       .
## Headcirc     0.09431770
## Gestation    0.03060225
## smoker       .
## mage         .
## mnocig       .
## mheight      .
## mppwt        .
## fage         .
## fedyrs       .
## fnocig       .
## fheight      .
## lowbwt      -0.24539434
## mage35       .
```

```
y.pred=predict(lasso.model,s=lambda.1se,newx=x.test)    #predict for test (with lambda.1se!)
mse.lasso=mean((y.test-y.pred)^2)                #mse for the predicted test rows
print(mean(y.pred))
```

```
## [1] 3.359424
```

For predicting the response Birthweight with default parameters as in the lecture and lambda=lambda.1se), we follow the process from Lecture 9 slide 19, and then averaging we get a birthweight of 3.298647, a little higher than with the previous method.

e)

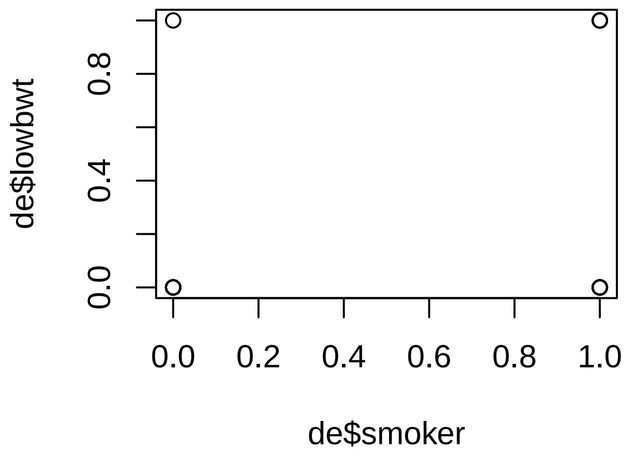
```
de = subset(d, select=c(lowbwt,Gestation,smoker,mage35))
cor.test(de$smoker,de$lowbwt)
```

```
##
##  Pearson's product-moment correlation
##
## data:  de$smoker and de$lowbwt
## t = 1.654, df = 40, p-value = 0.106
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.05516099  0.51717372
## sample estimates:
##       cor
## 0.2530122
```

```
cor.test(de$mage35,de$lowbwt)
```

```
##
##  Pearson's product-moment correlation
##
## data:  de$mage35 and de$lowbwt
## t = 0.6314, df = 40, p-value = 0.5314
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.2109609  0.3914524
## sample estimates:
##        cor
## 0.09933993
```

```
par(mar = c(4, 4,1, 4))
plot(de$smoker,de$lowbwt)
```



At this level of resolution, neither smoking nor older mothers seem to have lighter babies, for their correlations are respectively positive (0.2530122) and negligible (0.09933993). The additional plot shows this further.

f)

```
de = subset(d, select=c(lowbwt,Gestation,smoker,mage35))
model <- glm(de$lowbwt~de$Gestation+de$smoker+de$mage35, family="binomial", data=de)
exp(coef(model))
```

```
##  (Intercept) de$Gestation    de$smoker    de$mage35
## 1.892150e+21 2.314776e-01 2.326480e+02 1.380242e+00
```

To get the logistic model, we use the glm function, while to interpret the results in terms of odds we exponentiate the coefficients. Here, old mothers and smokers are 1.380242 and 232.648 times more likely to have a lighter baby (against results in e), which is reasonable, while with additional gestation time, it is 0.2314776 time more likely to have a lighter baby (meaning it is less likely).

h)

```
final_model <- lm(de$lowbwt~de$smoker*de$Gestation,data=de)
values <- data.frame(subset(de, Gestation == 40))
prediction  <- colMeans(data.frame(predict(final_model, values)))
```

```
## Warning: 'newdata' had 9 rows but variables found have 42 rows
```

The probability of low baby weight for each combination of levels of the involved factors and the gestation of 40 weeks is 0.1428571, which is reasonable considering that in our data none of the babies with that gestation were actually underweight.

i)

```r
table1 <- table(de$smoker,de$lowbwt)
c_test <- chisq.test(table1)
```

```
## Warning in chisq.test(table1): Chi-squared approximation may be incorrect
```

```r
table2 <- table(de$mage35,de$lowbwt)
c_test <- chisq.test(table2)
```

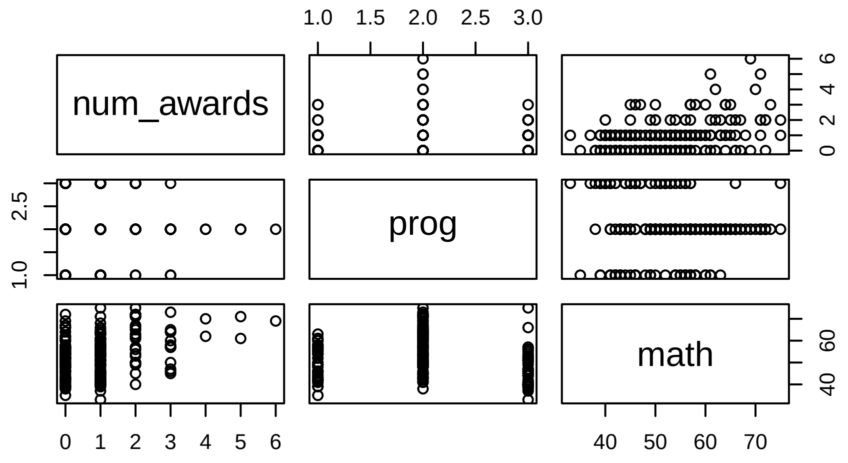
```
## Warning in chisq.test(table2): Chi-squared approximation may be incorrect
```

To apply a contingency table test we can use the table function and apply a Chi-squared test. We get p-values of 0.2308 and 1, which suggest that neither smoking nor older mothers have lighter babies. As seen in other sections, this is incorrect, and shows one disadvantage of this approach, that it is sensitive to a lack of enough data, making it a wrong approach for this case despite its advantage of being simple.

Exercise 3. School awards

```r
# Load the data
awards <- read.table("awards.txt", header = TRUE)

# Analyze the data
plot(awards)
```



a)

```r
# Perform Poisson regression without considering the variable math
poisson_model <- glm(num_awards ~ prog, family=poisson, data=awards)
# Display the summary of the model
summary(poisson_model)
```

```
##
## Call:
## glm(formula = num_awards ~ prog, family = poisson, data = awards)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.3485     0.2311  -1.508    0.131
## prog          0.1543     0.1047   1.474    0.141
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 228.83  on 199  degrees of freedom
## Residual deviance: 226.65  on 198  degrees of freedom
## AIC: 520.97
##
## Number of Fisher Scoring iterations: 5
```

```r
# Estimate numbers of awards for all three types of programs
prog_types <- unique(awards$prog)
for (prog_type in prog_types) {
  awards_estimate <- exp(predict(poisson_model, newdata = data.frame(prog = prog_type), type =
  cat("Estimated number of awards for program type", prog_type, ":", awards_estimate, "\n")
}
```

```
## Estimated number of awards for program type 3 : 3.068293
## Estimated number of awards for program type 1 : 2.278388
## Estimated number of awards for program type 2 : 2.613883
```

b) The Kruskal-Wallis test is appropriate for situations where the dependent variable is ordinal or continuous and the independent variable is categorical with two or more groups. In the context of the provided problem, where the dependent variable is the number of awards (count data) and the independent variable is the type of program (categorical), the Kruskal-Wallis test can indeed be used.

```r
# Perform the Kruskal-Wallis test
kruskal_test <- kruskal.test(num_awards ~ prog, data = awards)

# Display the results of the test
kruskal_test
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  num_awards by prog
## Kruskal-Wallis chi-squared = 10.755, df = 2, p-value = 0.00462
```

c)

```
# Perform Poisson regression with prog, math, and their interaction
poisson_model_with_math <- glm(num_awards ~ prog * math, family = "poisson", data = awards)

# Display the summary of the model
summary(poisson_model_with_math)
```

```
##
## Call:
## glm(formula = num_awards ~ prog * math, family = "poisson", data = awards)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.25454    1.72253  -2.470   0.0135 *
## prog         0.95195    0.74186   1.283   0.1994
## math         0.06911    0.03239   2.134   0.0328 *
## prog:math   -0.01348    0.01434  -0.940   0.3473
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 228.83  on 199  degrees of freedom
## Residual deviance: 198.17  on 196  degrees of freedom
## AIC: 496.49
##
## Number of Fisher Scoring iterations: 5
```

To assess the best program type for award numbers, analyze coefficient estimates for `prog` and interpret their significance. A positive coefficient suggests students in that program type typically earn more awards than the reference category, while a negative coefficient implies the opposite.

The interaction term (`prog * math`) helps determine if the program type effect on award numbers depends on the student's math score.

```
# Perform Poisson regression with prog, math, and their interaction
poisson_model_with_math <- glm(num_awards ~ prog * math, family = "poisson", data = awards)

# Display the summary of the model
print(poisson_model_with_math)
```

```
##
## Call:  glm(formula = num_awards ~ prog * math, family = "poisson", data = awards)
##
## Coefficients:
## (Intercept)         prog         math    prog:math
##    -4.25454      0.95195      0.06911     -0.01348
##
## Degrees of Freedom: 199 Total (i.e. Null);  196 Residual
## Null Deviance:       228.8
## Residual Deviance: 198.2     AIC: 496.5
```

```r
# Convert 'prog' to a factor variable
awards$prog <- factor(awards$prog)

# Perform Poisson regression with prog, math, and their interaction
poisson_model_prog_factor <- glm(num_awards ~ prog * math, family = "poisson", data = awards)

print(poisson_model_prog_factor)
```

```
##
## Call:  glm(formula = num_awards ~ prog * math, family = "poisson", data = awards)
##
## Coefficients:
## (Intercept)        prog2        prog3         math   prog2:math   prog3:math
##   -1.578440    -1.061226     0.962144     0.020365     0.027437    -0.009441
##
## Degrees of Freedom: 199 Total (i.e. Null);  194 Residual
## Null Deviance:       228.8
## Residual Deviance: 194.4     AIC: 496.7
```

Based on the model's results, none of the program types show a statistically significant effect on the number of awards earned by students. Additionally, the interaction between program type and math score does not significantly influence the number of awards.

```r
# Create a data frame with the predictor values
new_data <- data.frame(prog = c(1, 2, 3), math = 56)

# Predict the numbers of awards using the model
predicted_awards <- predict(poisson_model_with_math, newdata = new_data, type = "response")

# Display the predicted numbers of awards
predicted_awards
```

```
##         1         2         3
## 0.8292491 1.0097604 1.2295655
```