Assessed Practical I: Predicting the Olympic Games

Sungyeon Lim (201764876)

Data

The data includes information on GDP, population, and the number of medals won by each country in the 2008, 2012, and 2016 Olympic Games. There are a total of 71 countries in the dataset.

```
medals <- read.csv("medal_pop_gdp_data_statlearn.csv", header = TRUE)
head(medals)
```

```
##      Country     GDP Population Medal2008 Medal2012 Medal2016
## 1    Algeria  188.68   37100000         2         1         2
## 2  Argentina  445.99   40117096         6         4         4
## 3    Armenia   10.25    3268500         6         3         4
## 4  Australia 1371.76   22880619        46        35        29
## 5 Azerbaijan   63.40    9111100         7        10        18
## 6    Bahamas    7.79     353658         2         1         2
```

```
dim(medals)
```

```
## [1] 71  6
```

Task 1

A statistical model is defined using the glm function, which takes Population and GDP as input variables and Medal2012 as the output variable. By using the summary function, important information about the model, such as coefficient estimates, standard errors, degrees of freedom and AIC, are shown.

```
model1 <- glm(Medal2012 ~ Population + GDP, data = medals)
summary(model1)
```

```
##
## Call:
## glm(formula = Medal2012 ~ Population + GDP, data = medals)
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.076e+00  1.500e+00   4.051 0.000133 ***
## Population  5.247e-09  7.193e-09   0.729 0.468225
## GDP         7.564e-03  7.325e-04  10.326 1.45e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for gaussian family taken to be 132.1562)
##
##     Null deviance: 28402.8  on 70  degrees of freedom
## Residual deviance:  8986.6  on 68  degrees of freedom
## AIC: 553.19
##
## Number of Fisher Scoring iterations: 2
```

Based on the coefficient estimates, the model can be described with the following linear equation. To further analyse the model, confidence intervals are calculated for each estimate.

```
print(summary(model1)$coefficients[, 1:2])
```

```
##                 Estimate   Std. Error
## (Intercept) 6.076092e+00 1.499954e+00
## Population  5.246750e-09 7.192637e-09
## GDP         7.564081e-03 7.325406e-04
```

$$Medals = 6.076 + 5.247 \times 10^{-9} X_{Population} + 7.564 \times 10^{-3} X_{GDP} + \epsilon$$

The confidence intervals for the intercept and GDP coefficient estimates do not include 0, indicating that these estimates are significant. However, the confidence interval for the Population coefficient estimate includes 0, indicating that this estimate is not significant.

```
coeff1 <- summary(model1)$coefficients[, 1:2]

b0_min1 <- coeff1[1, 1]-(qt(0.975, 68))*coeff1[1, 2]
b0_max1 <- coeff1[1, 1]+(qt(0.975, 68))*coeff1[1, 2]

b1_min1 <- coeff1[2, 1]-(qt(0.975, 68))*coeff1[2, 2]
b1_max1 <- coeff1[2, 1]+(qt(0.975, 68))*coeff1[2, 2]

b2_min1 <- coeff1[3, 1]-(qt(0.975, 68))*coeff1[3, 2]
b2_max1 <- coeff1[3, 1]+(qt(0.975, 68))*coeff1[3, 2]

cat("Confidence interval for b0: ", c(b0_min1, b0_max1), "\n")
```

```
## Confidence interval for b0:  3.082981 9.069203
```

```
cat("Confidence interval for b1: ", c(b1_min1, b1_max1), "\n")
```

```
## Confidence interval for b1:  -9.105934e-09 1.959943e-08
```

```
cat("Confidence interval for b2: ", c(b2_min1, b2_max1))
```

```
## Confidence interval for b2:  0.006102319 0.009025843
```

Task 2

Another model is defined with the same inputs and output as before. However, this time, the output variable, Medal2012, is transformed using a logarithmic function.

```
model2 <- glm(log(Medal2012) ~ Population + GDP, data = medals)
summary(model2)
```

```
##
## Call:
## glm(formula = log(Medal2012) ~ Population + GDP, data = medals)
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.569e+00  1.263e-01  12.422  < 2e-16 ***
## Population  1.105e-10  6.058e-10   0.182    0.856
## GDP         3.161e-04  6.170e-05   5.123 2.68e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9376449)
##
##     Null deviance: 96.505  on 70  degrees of freedom
## Residual deviance: 63.760  on 68  degrees of freedom
## AIC: 201.85
##
## Number of Fisher Scoring iterations: 2
```

Again, the model can be described with a linear equation using the coefficient estimates derived from the model.

```
print(summary(model2)$coefficients[, 1:2])
```

```
##                  Estimate   Std. Error
## (Intercept) 1.569420e+00 1.263436e-01
## Population  1.105045e-10 6.058478e-10
## GDP         3.161018e-04 6.170312e-05
```

$$Medals = 1.569 + 1.105 \times 10^{-10} X_{Population} + 3.161 \times 10^{-4} X_{GDP} + \epsilon$$

Upon observing the confidence intervals for the coefficient estimates, it is evident that the intervals for intercept and GDP do not include 0, while the confidence interval for the coefficient estimate of Population includes 0. This indicates that the coefficient estimate of Population is not significant.

```
coeff2 <- summary(model2)$coefficients[, 1:2]

b0_min2 <- coeff2[1, 1]-(qt(0.975, 68))*coeff2[1, 2]
b0_max2 <- coeff2[1, 1]+(qt(0.975, 68))*coeff2[1, 2]

b1_min2 <- coeff2[2, 1]-(qt(0.975, 68))*coeff2[2, 2]
b1_max2 <- coeff2[2, 1]+(qt(0.975, 68))*coeff2[2, 2]
```

```r
b2_min2 <- coeff2[3, 1]-(qt(0.975, 68))*coeff2[3, 2]
b2_max2 <- coeff2[3, 1]+(qt(0.975, 68))*coeff2[3, 2]

cat("Confidence interval for b0: ", c(b0_min1, b0_max1), "\n")
```

```
## Confidence interval for b0:  3.082981 9.069203
```

```r
cat("Confidence interval for b1: ", c(b1_min1, b1_max1), "\n")
```

```
## Confidence interval for b1:  -9.105934e-09 1.959943e-08
```

```r
cat("Confidence interval for b2: ", c(b2_min1, b2_max1))
```

```
## Confidence interval for b2:  0.006102319 0.009025843
```

Task 3

In this dataset, Medal2012 represents the number of medals each country won in 2012. Since these outputs are count data with non-negative integer values, the Poisson distribution is appropriate. A Poisson distribution model with a log link function is attempted. However, the AIC value of the model is significantly larger than the previous two models. This suggests that the data is not suitable for the Poisson distribution.

```r
model3 <- glm(Medal2012 ~ Population + GDP,  data = medals, family=poisson(link="log"))
summary(model3)
```

```
##
## Call:
## glm(formula = Medal2012 ~ Population + GDP, family = poisson(link = "log"),
##     data = medals)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 2.193e+00  4.034e-02  54.360  < 2e-16 ***
## Population  6.049e-10  9.131e-11   6.625 3.48e-11 ***
## GDP         1.715e-04  6.672e-06  25.708  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1331.81  on 70  degrees of freedom
## Residual deviance:  690.27  on 68  degrees of freedom
## AIC: 962.24
##
## Number of Fisher Scoring iterations: 5
```

Further analysis shows that the data does not meet one of the assumptions of the Poisson distribution, which is that the mean and variance of the model are identical. By visualising the distribution of the output variable, it is evident that there are several outliers that can be removed to develop the model further.

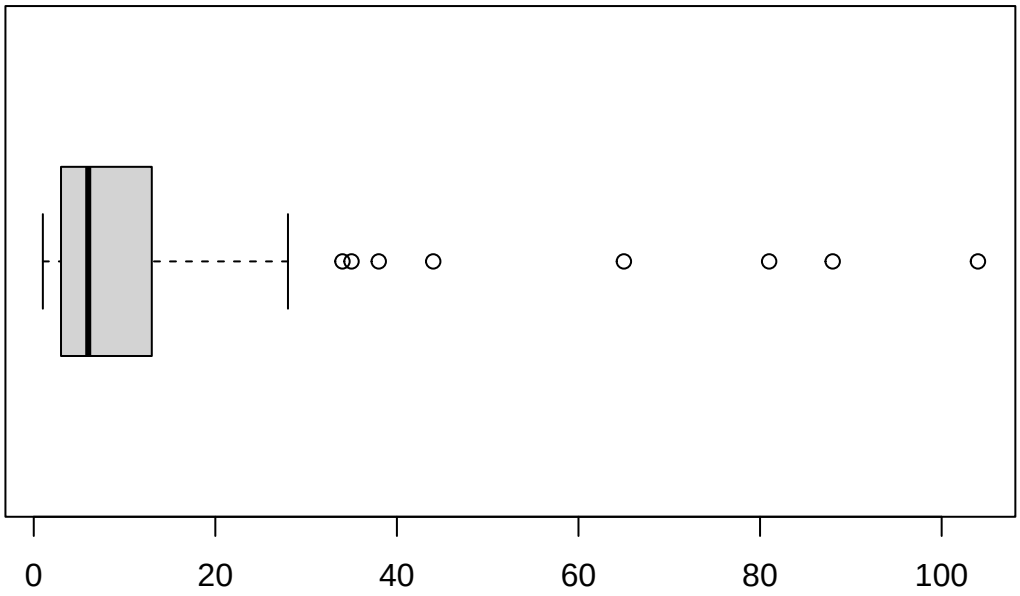
```r
mean(medals$Medal2012)
```

```
## [1] 13.29577
```

```r
var(medals$Medal2012)
```

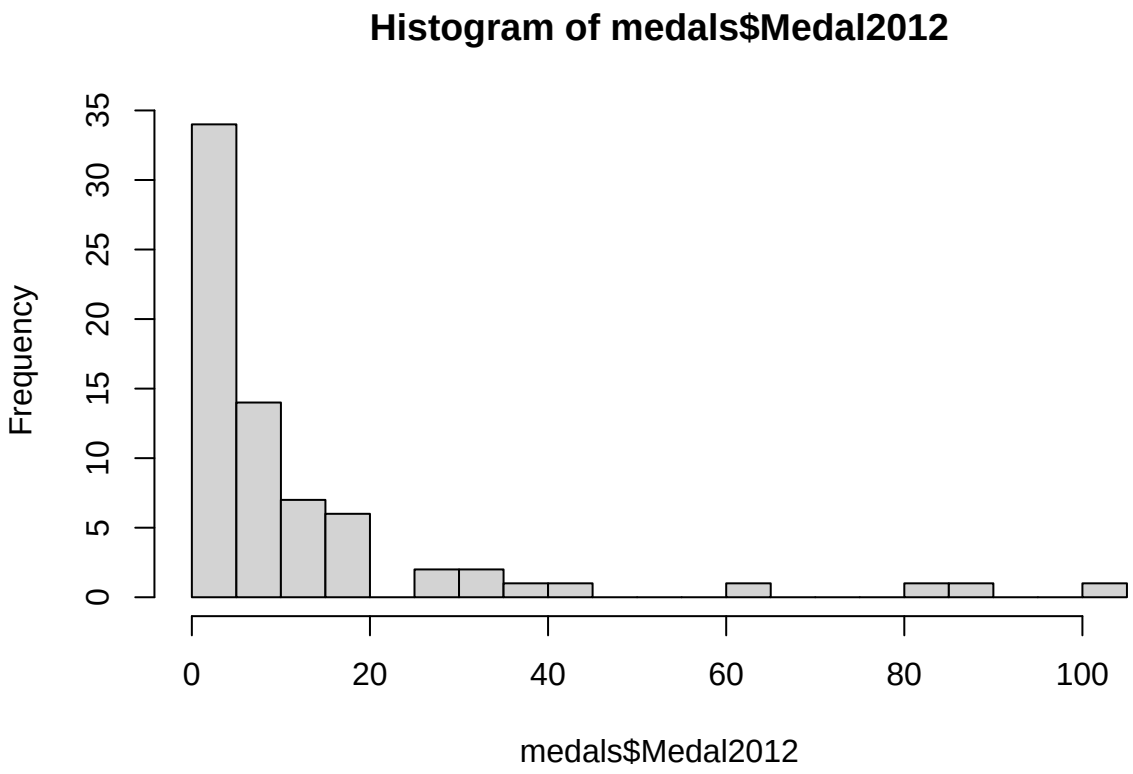
```
## [1] 405.7541
```

```r
boxplot(medals$Medal2012, horizontal=TRUE)
```

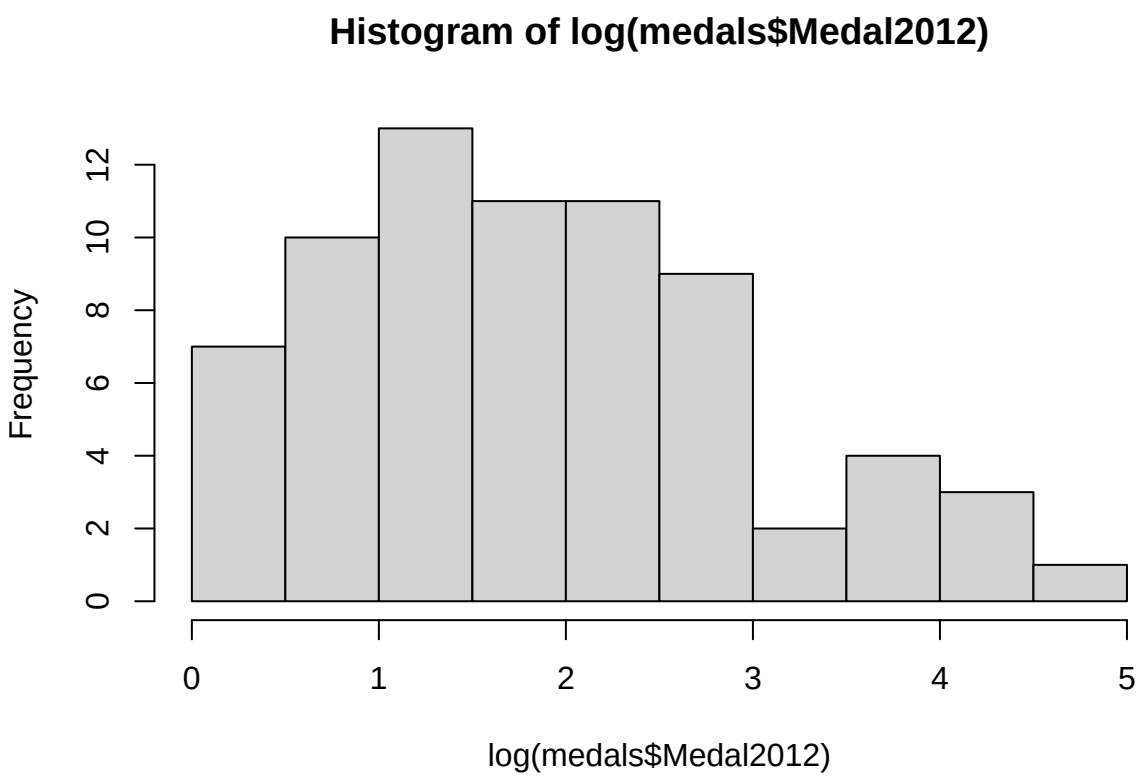


To use the same inputs and outputs from task 1, another model is developed from the model from task 2, which performed well. Since log transformation is applied to the output variable, it becomes continuous, allowing Gaussian distribution to be used. Gaussian distribution assumes that the data is normally distributed. The histograms show that the output variable has been transformed to be normally distributed using the log transformation.

```r
hist(medals$Medal2012, breaks=15)
```




```
hist(log(medals$Medal2012), breaks=15)
```



Another assumption the Gaussian distribution makes is that the relationship between the input and output variables is linear. By checking correlations, it is observed that the linearity of the log-transformed output variable and Population is low. Therefore, by applying log transformation to the Population input variable, the correlation value increased.

```
cor(log(medals$Medal2012), medals$GDP)
```

```
## [1] 0.5822243
```

```
cor(log(medals$Medal2012), medals$Population)
```

```
## [1] 0.2903699
```

```
cor(log(medals$Medal2012), medals$GDP)
```

```
## [1] 0.5822243
```

```
cor(log(medals$Medal2012), log(medals$Population))
```

```
## [1] 0.487317
```

Here, a model is defined with a log-transformed output variable and inputs as log-transformed Population and GDP.

```r
model3 <- glm(log(Medal2012) ~ log(Population) + GDP,  data = medals)
summary(model3)
```

```
##
## Call:
## glm(formula = log(Medal2012) ~ log(Population) + GDP, data = medals)
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.449e+00  1.381e+00  -1.049 0.297922
## log(Population)  1.846e-01  8.406e-02   2.196 0.031518 *
## GDP              2.487e-04  6.216e-05   4.001 0.000158 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.8759891)
##
##     Null deviance: 96.505  on 70  degrees of freedom
## Residual deviance: 59.567  on 68  degrees of freedom
## AIC: 197.02
##
## Number of Fisher Scoring iterations: 2
```

The coefficient estimates suggest that the model can be expressed using the following linear equation.

```r
print(summary(model3)$coefficients[, 1:2])
```

```
##                      Estimate   Std. Error
## (Intercept)     -1.4490569841 1.381457e+00
## log(Population)  0.1845746479 8.405629e-02
## GDP              0.0002486662 6.215789e-05
```

$$Medals = -1.449 + 0.185 X_{Population} + 0.000249 X_{GDP} + \epsilon$$

Furthermore, it is clear that the confidence intervals for the coefficient estimates of Population and GDP do not include 0. However, the confidence interval for the intercept includes 0.

```r
coeff3 <- summary(model3)$coefficients[, 1:2]

b0_min3 <- coeff3[1, 1]-(qt(0.975, 68))*coeff3[1, 2]
b0_max3 <- coeff3[1, 1]+(qt(0.975, 68))*coeff3[1, 2]

b1_min3 <- coeff3[2, 1]-(qt(0.975, 68))*coeff3[2, 2]
b1_max3 <- coeff3[2, 1]+(qt(0.975, 68))*coeff3[2, 2]

b2_min3 <- coeff3[3, 1]-(qt(0.975, 68))*coeff3[3, 2]
b2_max3 <- coeff3[3, 1]+(qt(0.975, 68))*coeff3[3, 2]

c(b0_min3, b0_max3)
```

```
## [1] -4.205711  1.307597
```

```r
c(b1_min3, b1_max3)
```

```
## [1] 0.01684293 0.35230637
```

```r
c(b2_min3, b2_max3)
```

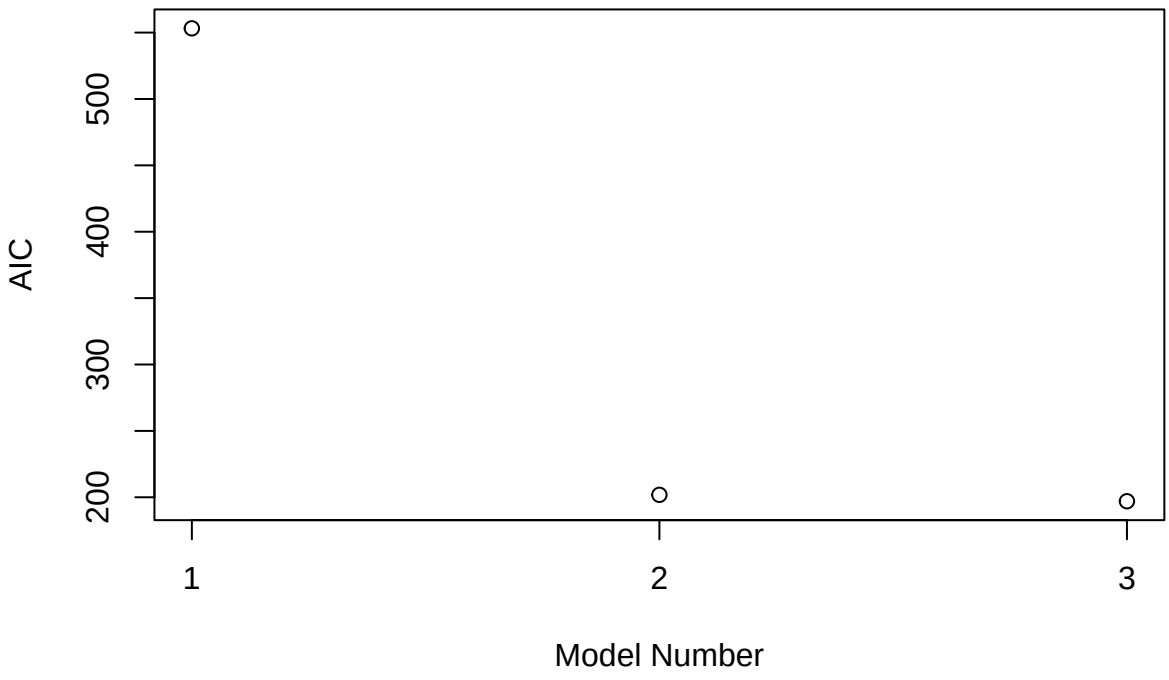
```
## [1] 0.0001246320 0.0003727003
```

This way, the model is designed to be both interpretable and robust. This is because the coefficients for the input variables are significant, which means that they can be used to describe the relationships between the input and output variables. Additionally, the lower AIC value of the model indicates that it is more robust and accurate in its predictions.

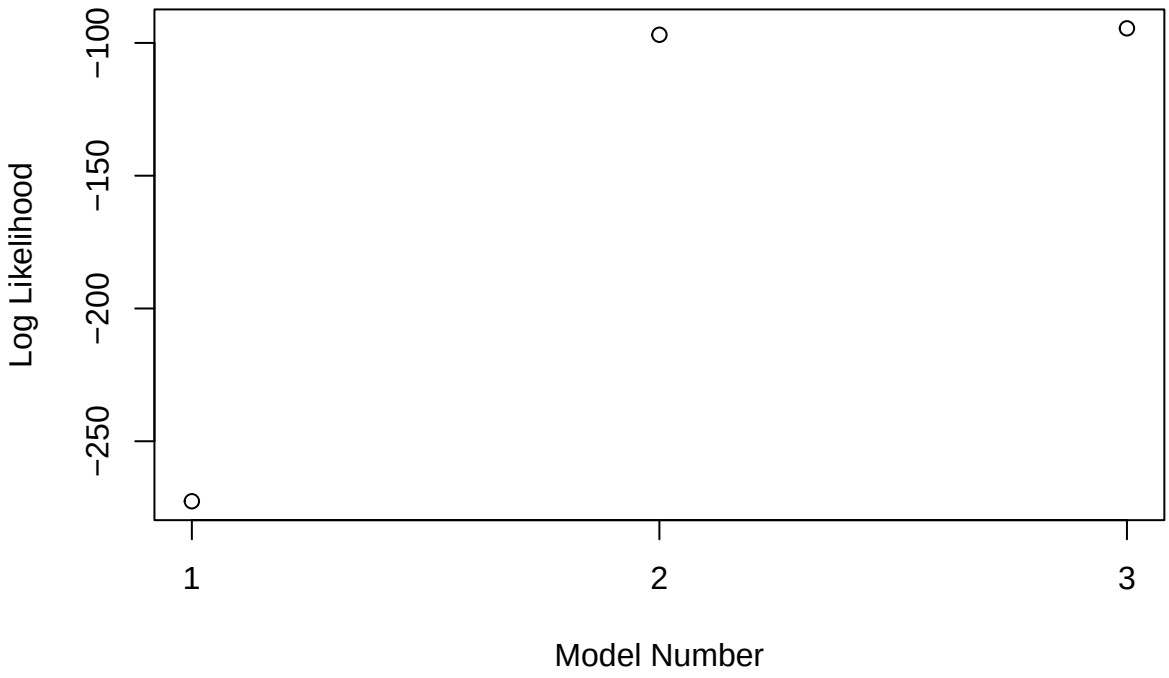
Task 4

After comparing the AIC and log-likelihood values for the three models from tasks 1 to 3, it is clear that model3 performs the best. The graphs below show that model3 has slightly lower AIC and slightly higher log-likelihood values compared to model2.

```r
model_aic <- rep(NA, 3)
model_aic[1] <- model1$aic
model_aic[2] <- model2$aic
model_aic[3] <- model3$aic

plot(1:length(model_aic), model_aic, xlab="Model Number", ylab="AIC", xaxt='n')
axis(side=1, at=seq(1, 3, by=1))
```




```
loglikeli <- rep(NA, 3)
loglikeli[1] <- logLik(model1)
loglikeli[2] <- logLik(model2)
loglikeli[3] <- logLik(model3)

plot(1:length(loglikeli), loglikeli, xlab="Model Number", ylab="Log Likelihood", xaxt='n')
axis(side=1, at=seq(1, 3, by=1))
```



To further analyse the results, cross-validation is applied. The data is divided into two datasets: training and test. Then, the three models are defined the predictive log-likelihood values are calculated using sigma and predicted values from the trained models. This process is repeated 100 times and the number of times each model had the highest predictive log-likelihood is calculated. According to the frequency graph, model3 had the highest frequency, indicating it is the best among the three models in the 100 trials.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

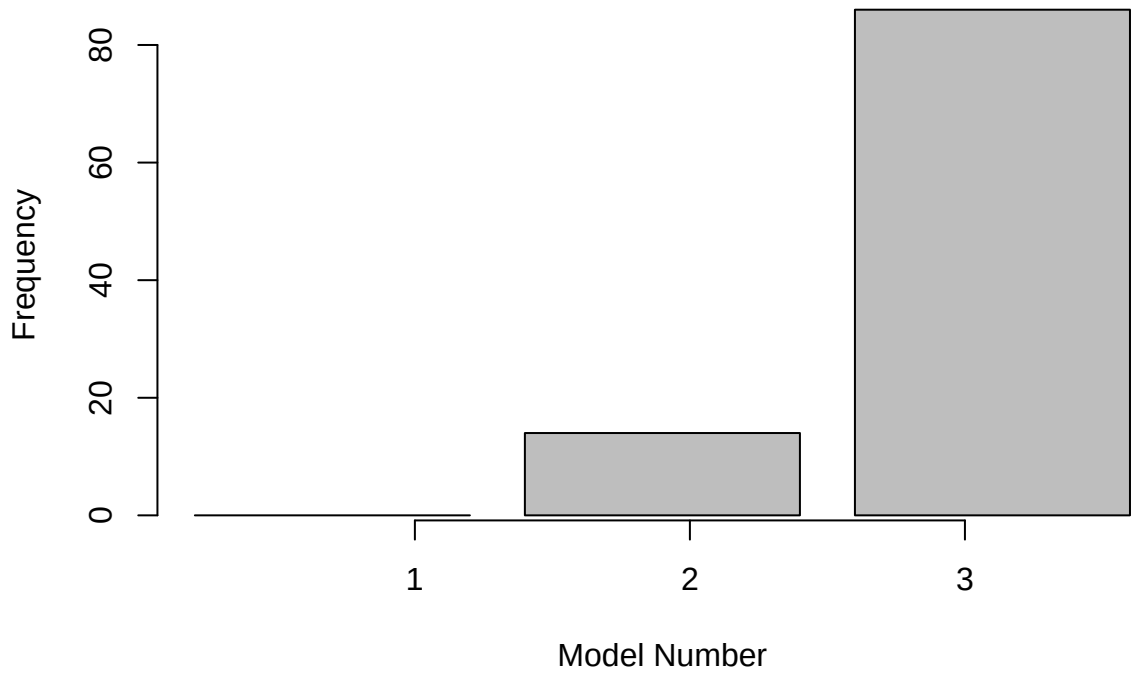
```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## Loading required package: lattice
```

```
best_counts <- c(0, 0, 0)

for (i in 1:100){
  set.seed(i)
  index <- createDataPartition(medals$Medal2012, p=0.7, list=FALSE)
  train_data <- medals[index,]
  test_data <- medals[-index, ]
```

```r
  model1 <- glm(Medal2012 ~ Population + GDP, data = train_data)
  model2 <- glm(log(Medal2012) ~ Population + GDP, data = train_data)
  model3 <- glm(log(Medal2012) ~ log(Population) + GDP,  data = train_data)

  sigma1 <- sqrt(summary(model1)$dispersion)
  sigma2 <- sqrt(summary(model2)$dispersion)
  sigma3 <- sqrt(summary(model3)$dispersion)

  pred1 <- predict(model1, test_data)
  pred2 <- predict(model2, test_data)
  pred3 <- predict(model3, test_data)

  model1_like <- sum(dnorm(test_data$Medal2012, pred1, sigma1, log=TRUE))
  model2_like <- sum(dnorm(log(test_data$Medal2012), pred2, sigma2, log=TRUE))
  model3_like <- sum(dnorm(log(test_data$Medal2012), pred3, sigma3, log=TRUE))

  best_idx <- which.max(c(model1_like, model2_like, model3_like))
  best_counts[best_idx] <- best_counts[best_idx]+1
}

barplot(best_counts, xlab="Model Number", ylab="Frequency")
axis(side=1, at=seq(1, 3, by=1))
```



In conclusion, based on AIC, log-likelihood, and predicted log-likelihood values, it is evident that model3 is suitable for accurately predicting the medal count.

Task 5

Since the model3 is a linear regression model with a Gaussian distribution, the probability where the medal count being at least one can be calculated using the following equation.

$$P(y \geq 1) = 1 - \Phi(1; \mu, \sigma)$$

Based on model3 and the Population and GDP values of Great Britain, the probability of Great Britain winning at least one medal is 0.94.

```
model3 <- glm(log(Medal2012) ~ log(Population) + GDP,  data = medals)

GB_Population <- 62262000
GB_GDP <- 2431.59
GB_data <- data.frame(Population=GB_Population, GDP=GB_GDP)

GB_mu <- predict(model3, newdata=GB_data)
GB_sd <- sqrt(sum(model3$residuals^2)/(length(model3$residuals)-length(model3$coefficients)))

1-pnorm(1, mean=GB_mu, sd=GB_sd)
```

```
## [1] 0.941631
```