Valérie BLANCH

2614867B

ASSIGNMENT N°1 - OLYMPIC MEDALS

1. INTRODUCTION

This data set contains 108 observations and 43 variables, related to the Olympic Games from 2000 to 2016. Each observation corresponds to each country participating in the Olympics.

Here is a quick summary of the variables we will be working with, grouped by type:

Character variables:

- *country*: the name of each country
- *country.code*: a three-letter code representing each country

Continuous variables:

- *gdp00* up to *gdp16* (5 variables): the Gross Domestic Product for each observation and each year in millions of USD
- *altitude*: altitude of the capital city

Logical variables (1 for yes, 0 for no):

- *soviet*: if the country was part of the USSR or not
- *comm:* if the country is/was a communist state
- *muslim*: if the country is mostly Muslim
- *oneparty*: if the country is a one-party state
- *host*: if the country hosts the Games

Discrete variables:

- *pop00* up to *pop16* (5 variables): the population in thousands for each year.
- *athletes00* up *to athletes16* (5 variables): the number of athletes per country for each year.
- *totgold00* up to *totgold16* (5 variables): the total number of gold medals won per year
- *totmedals00* up to *totmedals16* (5 variables): the total number of medals won per year
- *gold00* up to *gold16* (5 variables): the number of gold medals won by each country per year
- *tot00* up to *tot16* (5 variables): the number of medals won by each country per year

This project's objective is to predict the number of gold medals won by each country for the year 2016.

Missing values:

3 missing values have been found:

- the GDP of Afghanistan for 2000
- the GDP of Cuba for 2016
- the GDP of the Syrian Arab Republic for 2016

Because there are only 3 missing values, and that they can be easily found on the Internet, they have been imputed manually with their official UN estimations.

These numbers have been found in the database provided by the Statistics Division of the United Nations.

If the dataset had been bigger, with many more missing values, different approaches could have been tested:

- mean imputation: replacing missing values by the mean of each country GDPs.
- model imputation: using the R package MICE to compute the missing values, with methods like linear regression or predictive mean matching.

Data Transformation:

- The variables *gold00, gold04, gold08, gold12* and *gold16* have been regrouped into one variable called *gold.*
- The variables *athletes00, athletes04, athletes08, athletes12* and *athletes16* have been regrouped into one variable called *athletes.*
- The variables *gdp00, gdp04, gdp08, gdp12* and *gdp16* have been regrouped into one variable called *log_gdp* which has been log-transformed to account for high variability.
- The variables *pop00, pop04, pop08, pop12* and *pop16* have been regrouped into one variable called *log_pop* which has been log-transformed to account for high variability.
- The variables *tot00, tot04, tot08, tot12* and *tot16* have been regrouped into one variable called *tot.*
- The variables *totgold00, totgold04, totgold08, totgold12* and *totgold16* have been regrouped into one variable called *totgold.*
- The variables *totmedals00, totmedals04, totmedals08, totmedals12* and *totmedals16* have been regrouped into one variable called *totmedals.*
- The variable *country.code* has been removed since it contains the same information as *country.*
- A column *year* has been added to recognize merged values and separate the data for testing.

As a result, the final complete set contains 540 observations and 15 variables.

2. EXPLORATORY ANALYSIS

The training set contains all variables from the original set (apart from *country* and *year*), and the observations from 2000 to 2012 (row 1 to row 432). It has been used for the following EDA.

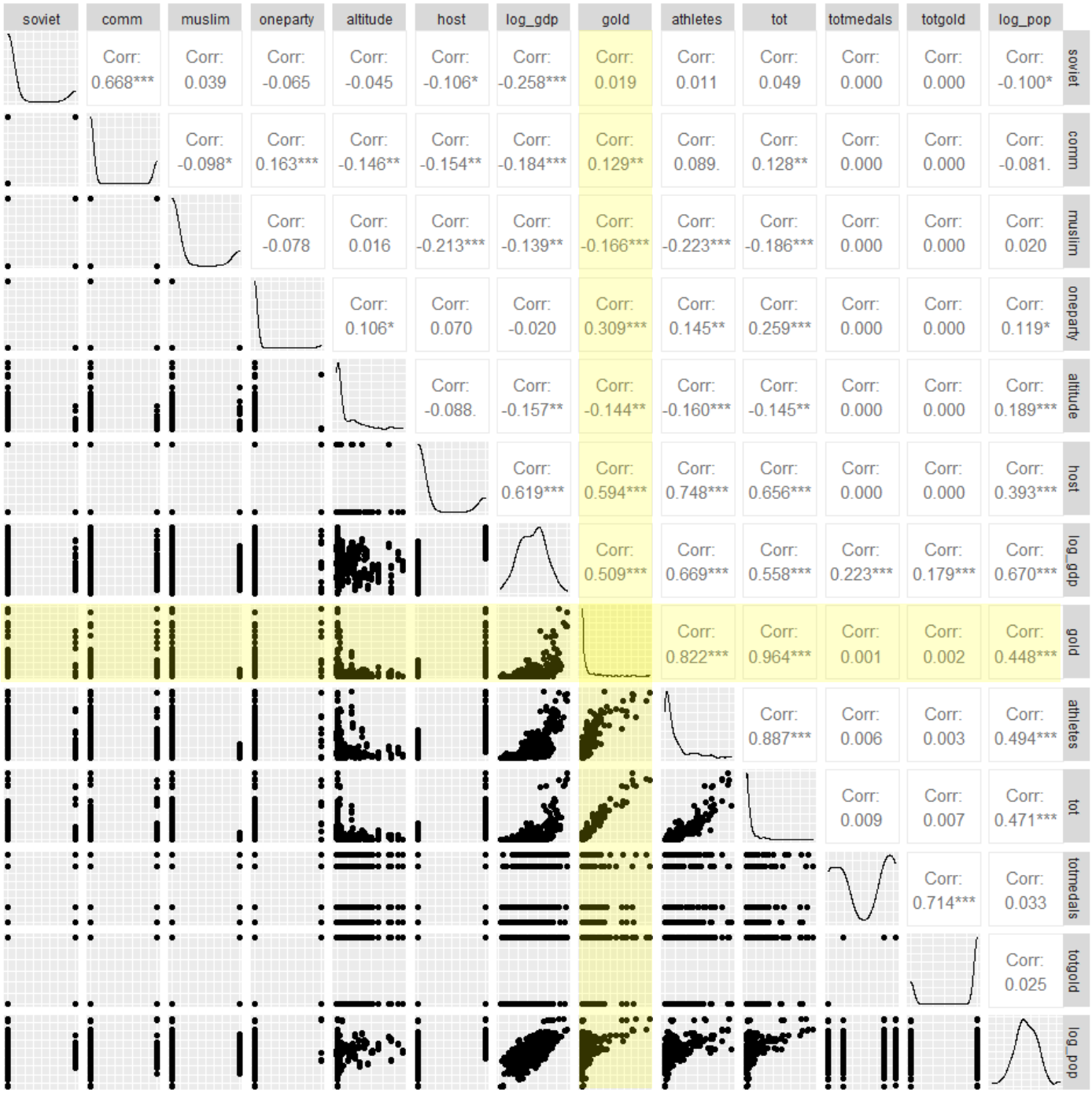
Linearity:

The ggpairs() function from the GGally package computes a correlation plot, where the upper part shows Pearson's correlation coefficients (covariance by the product of the standard deviations).

The coefficients, ranging from -1 to 1, give information about linear relationships between variables of the dataset. The higher the absolute value of the coefficient ρ, the stronger the linear relationship between variables. The sign of the coefficient indicates if the relationship is negative or positive; a value close to zero means no significant correlation.

In this plot, the response variable *gold* has been highlighted. It seems that many variables in the data set are positively correlated with it:

- The GDP (positive linear relationship)
- The population (positive linear relationship)
- The number of athletes (positive linear relationship)
- The total number of medals (positive linear relationship)
- And there seems to be what is called a “host effect”: the country that hosts the competition has a tendency to overperform.



Thus, a normal linear or poisson regression model can be used for prediction, since the assumption of linear relationship between predictors and responses is satisfied.

Multicollinearity:

The plot shows strong correlations between the predictors, for example *comm* and *soviet* (which makes sense), *host and tot* etc.

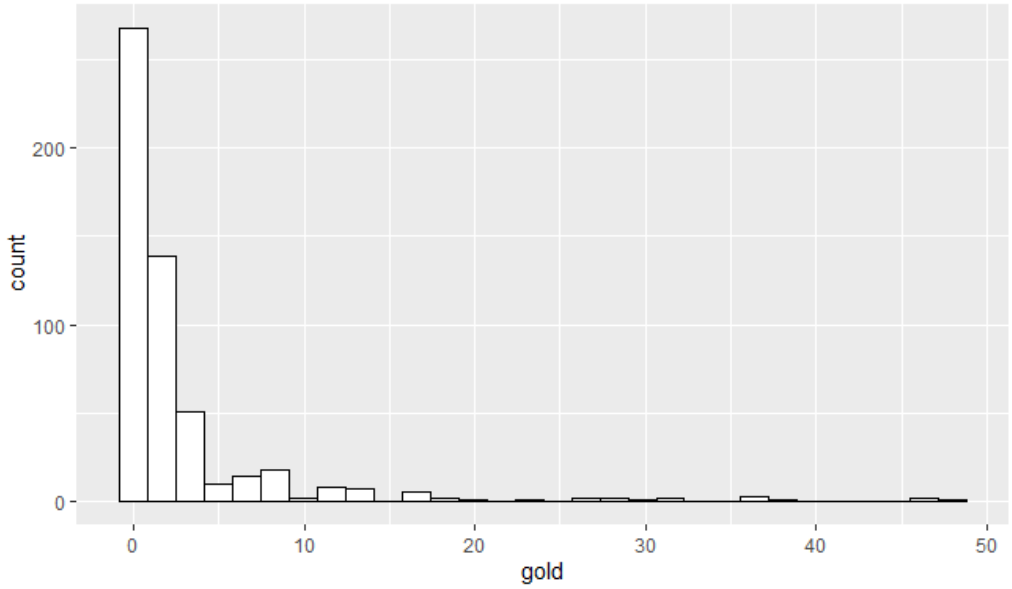
Multicollinearity means that the data is redundant; the variables set must be reduced while maintaining maximum information. Otherwise, issues might arise during modeling, e.g. biased coefficient estimates and p-values. This issue will be adressed during the model selection process.

Distribution of *gold*:

The response *gold* is skewed to the right, and unimodal. The peak in the histogram shows the high number of zeros in the data. Indeed, a lot of countries did not win any gold medals during the years studied. The proportion of zeros in the target variable has been calculated: most than half of the values are equal to zero.

```
> table(oldat_train$gold > 0)

FALSE  TRUE
  217   215
```

Intuitively, a zero-inflated model or hurdle model seem to be appropriate in the context of count data with a lot of zeros.



Therefore, in the next part of this project, 4 models will be trained, tested and compared:

a) A normal linear model, even though it is commonly used for normally distributed data
b) A Poisson regression model, useful for count data
c) A negative binomial model if the issue of overdispersion arises
d) A hurdle model, to account for the large amount of zeros

3. MODEL BUILDING

a) Normal linear model:

A multiple linear model has been fitted using *gold* as a target and all variables as predictors, using the lm() function.

This model has been used as a full model to perform stepwise backward selection, using the stepAIC() function from the MASS package.

The algorithm removes variables (and multicollinearity) from the full model by calculating the Akaike Information Coefficient at each step, until an optimum between simplification of the model and information loss is found (lowest AIC). The AIC is based on MLE and is an appropriate criterion to compare regression models meant for prediction.

In other words, the stepAIC() function finds the model that fits the data well, with the fewest possible predictors, containing most of the information.

The algorithm found the following optimal combination, with an AIC of 409.45.

```
Step:  AIC=409.45
gold ~ soviet + oneparty + host + athletes + tot

           Df Sum of Sq    RSS     AIC
<none>                  1084.0  409.45
- host      1       9.2 1093.3  411.11
- soviet    1      19.0 1103.0  414.95
- oneparty  1      35.0 1119.0  421.17
- athletes  1      36.4 1120.4  421.70
- tot       1    3988.7 5072.7 1074.11

Call:
lm(formula = gold ~ soviet + oneparty + host + athletes + tot,
    data = oldat_train)

Coefficients:
(Intercept)       soviet     oneparty         host     athletes
  -0.076866    -0.639596     1.838940    -0.588771    -0.006156
        tot
   0.410167
```

However, the AIC that will be used to compare the linear model to the Poisson model is the value obtained from the AIC() function, which accounts for the estimation of the variance of the error:

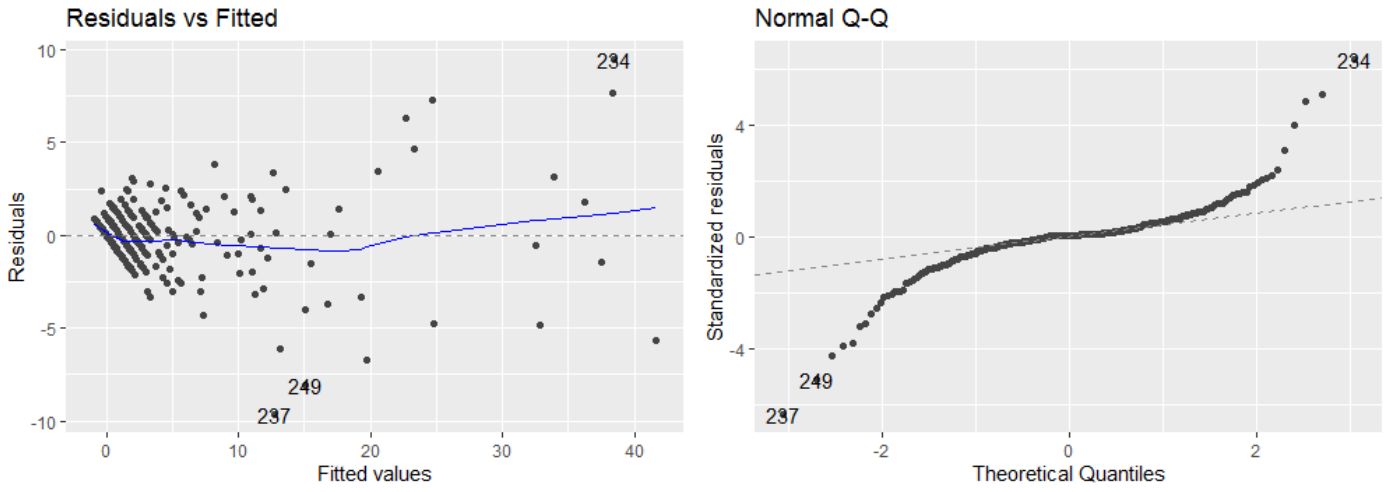
```
> AIC(linear_gold)
[1] 1646.454
```

Goodness-of-Fit:

The summary of the model shows an R-squared and an adjusted R-squared of 0.94, which means that the model explains 94% of the variability of the data. It is a good score, but it gives no information on predictive performances.

```
> summary(linear_gold)$r.squared
[1] 0.9388982

> summary(linear_gold)$adj.r.squared
[1] 0.9380356
```

Residual plots:



The Residuals vs Fitted plot shows a “streaky” pattern and heteroscedasticity (the variance is not constant). This is not surprising, as it reflects a linear model fitted for a discrete target variable.

As for the QQ plot, the residuals follow the normal line very loosely, and the tails are being strongly pulled by several outliers.

b) Poisson regression model:

Following the same procedure, a full Poisson model has first been built with the glm() function, then selected using the stepAIC() function (backward selection). As a result, the following output has been obtained, with an AIC of 1359.76 which is lower than the AIC of the linear model, indicating a potential and relative better fit.

```
Step:  AIC=1359.76
gold ~ soviet + comm + muslim + host + log_gdp + athletes + tot +
    totmedals

            Df Deviance    AIC
<none>           694.29 1359.8
- soviet     1   697.12 1360.6
- totmedals  1   707.34 1370.8
- muslim     1   707.95 1371.4
- host       1   719.59 1383.1
- tot        1   725.31 1388.8
- log_gdp    1   727.87 1391.3
- athletes   1   728.47 1392.0
- comm       1   785.21 1448.7

Call:  glm(formula = gold ~ soviet + comm + muslim + host + log_gdp +
    athletes + tot + totmedals, family = "poisson", data = oldat_train)

Coefficients:
(Intercept)       soviet         comm       muslim         host
   4.065039    -0.176347     0.771397    -0.608467     0.597947
    log_gdp     athletes          tot    totmedals
   0.180095     0.002464     0.011363    -0.006828

Degrees of Freedom: 431 Total (i.e. Null);  423 Residual
Null Deviance:      3263
Residual Deviance: 694.3        AIC: 1360
```

Goodness-of-Fit:

However, a chi-squared test has been calculated to evaluate the fit of the Poisson regression model:

```
> 1-pchisq(summary(poisson_gold)$deviance,
+          summary(poisson_gold)$df.residual)
[1] 2.220446e-16
```

With 431 degrees of freedom and a residual deviance of 694.3, the p-value of the test statistic is inferior to the significance level of 0.05. This indicates a rather bad fit.

Overdispersion:

The Poisson distribution assumes the variance is equal to the mean. To verify this assumption, a dispersion test can be conducted.

Overdispersion happens when the variance is superior to the mean: the check_overdispersion() function from the *performance* package calculates the dispersion ratio, which is Pearson's Chi-squared statistic divided by the degrees of freedom. In other words, it computes the relative variance (variance/mean), which should be ideally equal to 1.

```
> check_overdispersion(poisson_gold)
# Overdispersion test

       dispersion ratio =   1.729
  Pearson's Chi-Squared = 727.975
              p-value = < 0.001

Overdispersion detected.
```

Since overdispersion has been detected, a negative binomial model will be fitted; it has the advantage of not assuming equality between variance and mean and is often preferred when overdispersion is present.

c) Negative binomial model:

Following the same procedure, the functions glm.nb() then stepAIC() have been used to fit the model. Here is the output:

```
Step:  AIC=1282.24
gold ~ comm + muslim + log_gdp + athletes + tot + totmedals

            Df    AIC
<none>         1282.2
- totmedals  1 1284.5
- athletes   1 1289.5
- muslim     1 1289.8
- comm       1 1305.0
- tot        1 1306.2
- log_gdp    1 1306.7

Call:  glm.nb(formula = gold ~ comm + muslim + log_gdp + athletes +
    tot + totmedals, data = oldat_train, control = glm.control(maxit = 50),
    init.theta = 2.696409903, link = log)

Coefficients:
(Intercept)         comm       muslim      log_gdp     athletes
   3.355600     0.653121    -0.588974     0.230479     0.002668
        tot    totmedals
   0.026515    -0.006828

Degrees of Freedom: 431 Total (i.e. Null);  425 Residual
Null Deviance:      1489
Residual Deviance: 428.4        AIC: 1284
```

The AIC of the negative binomial model is the lowest so far.

The model will probably not solve the excess of zeros, which is why a hurdle model will be fitted at last.

d) Hurdle model:

After using the hurdle() function from the *pscl* package, then stepAIC() to optimise the formula, the following output has been printed:

```
Step:  AIC=1111.4
gold ~ soviet + comm + altitude + host + athletes + tot

           Df    AIC
<none>        1111.4
- altitude  2 1112.9
- athletes  2 1116.4
- soviet    2 1122.4
- comm      2 1141.7
- host      2 1159.0
- tot       2 1297.6
```

The AIC is lower than the others.

The summary shows that the hurdle model is composed of two separate models using the same variables:

A truncated Poisson model which ignores the zeros in the target variable, and a logistic regression model, which focuses on the binomial probability of not winning versus winning gold medals.

```
Call:
hurdle(formula = gold ~ soviet + comm + altitude + host + athletes +
    tot, data = oldat_train)

Pearson residuals:
    Min      1Q  Median      3Q     Max
-2.3466 -0.4050 -0.2604  0.1746  3.9789

Count model coefficients (truncated poisson with log link):
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.031e-01  8.793e-02   4.584 4.55e-06 ***
soviet      -3.856e-01  1.019e-01  -3.784 0.000154 ***
comm         4.823e-01  8.183e-02   5.894 3.77e-09 ***
altitude    -2.087e-04  6.807e-05  -3.066 0.002167 **
host         7.651e-01  1.074e-01   7.124 1.05e-12 ***
athletes     1.229e-03  4.067e-04   3.022 0.002511 **
tot          1.831e-02  1.954e-03   9.369  < 2e-16 ***
Zero hurdle model coefficients (binomial with logit link):
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.4330303  0.2984207  -8.153 3.55e-16 ***
soviet       0.3190338  0.6285121   0.508    0.612
comm        -0.3771758  0.5442508  -0.693    0.488
altitude    -0.0001190  0.0002383  -0.499    0.617
host        -0.9355147  0.6400367  -1.462    0.144
athletes     0.0033553  0.0055334   0.606    0.544
tot          0.7204993  0.0961114   7.497 6.55e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Number of iterations in BFGS optimization: 15
Log-likelihood: -541.7 on 14 Df
```

4. PREDICTIONS

A validation set has been created using the rows 433 to 540, corresponding to the year 2016.

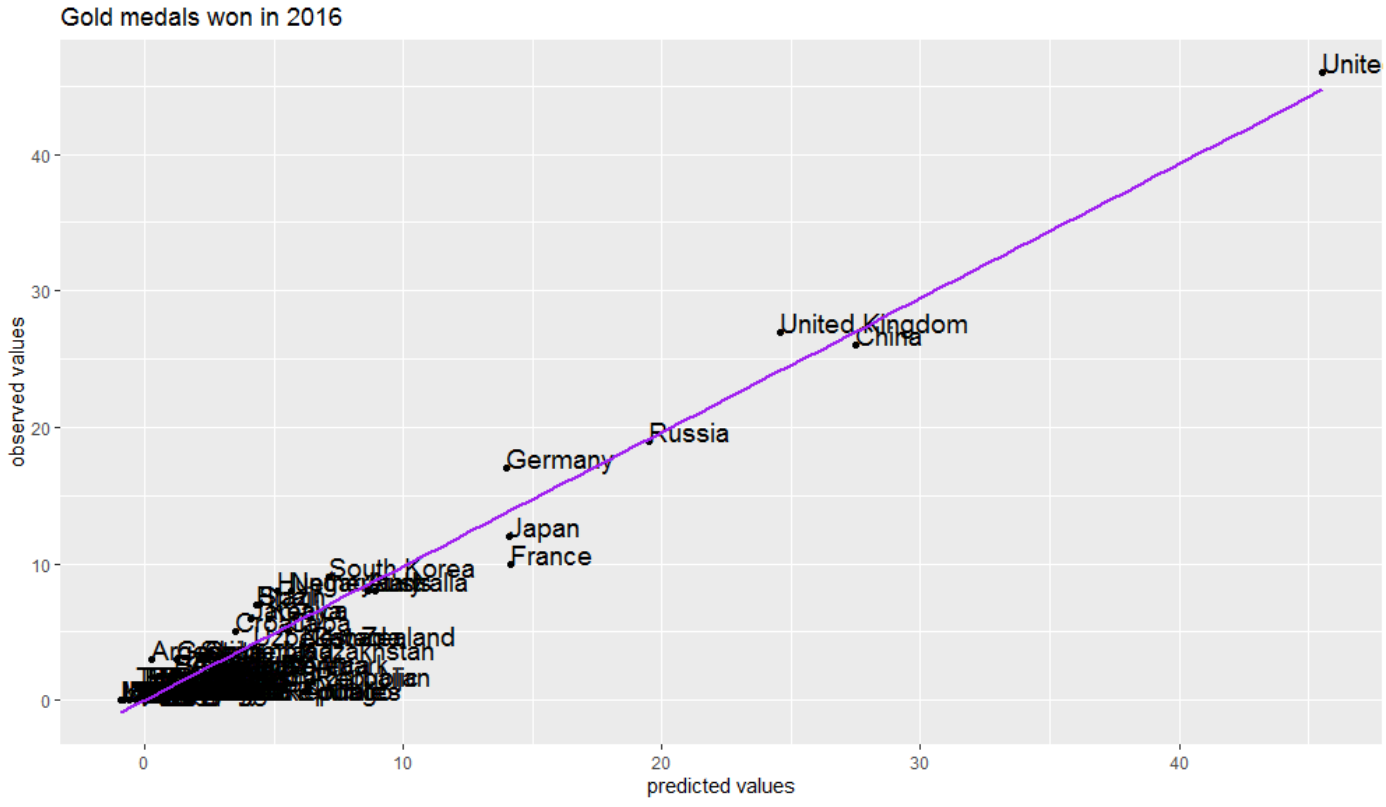
Using the predict() function then the rmse() function, the latter comparing the predictions to the observed values, the RMSE of each models has been calculated.

```
> rmse(oldat[433:540,10], linear_predicted)
[1] 1.327704

> rmse(oldat[433:540,10], poisson_predicted)
[1] 6.094208

> rmse(oldat[433:540,10], negbi_predicted)
[1] 5.973234

> rmse(oldat[433:540,10], hurdle_predicted)
[1] 3.240479
```

Summarising the findings of this project into a table, we can conclude that, surprisingly, the normal linear model is the best predictive model for this particular data set, as it has the lowest RMSE, even if its AIC is the highest.

Model	AIC	RMSE
Normal linear	1646.45	1.33
Poisson	1359.76	6.09
Negative Binomial	1282.24	5.97
Hurdle	1111.40	3.24

To visualise the results of the normal linear model, the fitted values versus the observed ones has been drawn:



A regression line has been added to visualise which countries have been underestimated by the model in their ability to win gold medals in 2016 – Germany, United Kingdom – and which countries have been overestimated – Japan and France.

5. CONCLUSION

The objective of this project was to predict the number of gold medals won in 2016 by 108 countries.

First, the data set has been transformed to fit a regression analysis: the missing values have been replaced, and many variables modified/merged.

By looking at the distribution of the target variable, which is skewed to the right with half of its values equal to zero, it has been decided to test several models:

- A normal linear model, which proved to be the most performant in the end – perhaps because the variables are very strongly linearly related – even though the residual plots were not typical.
- A Poisson model, which is usually preferred for count data, however in this case it performed relatively poorly, with a RMSE of 6.09, probably because of the overdispersion of the data.
- A negative binomial model to compensate for the overdispersion, but did not give proper results
- A hurdle model, with a reasonably good RMSE, but still inferior to the linear model's statistic.

Improvements:

- More relevant predictors could be added to the data set, e.g. GDP per capita and/or HDI.
- Winter games and summer games could be separated in the data set and thus bring more insight (and make the variable *altitude* more relevant).
- More years could be added.
- Computing cross-validation to account for the relatively small size of the data set.
- Zero-inflated Poisson or Negative Binomial models could be fitted.