IRONMAN 70.3 RACE DATA BETWEEN 2004 AND 2020

Alba Puig Font

NIU: 1636034

Anàlisis de Dades Complexes

2nd MatCAD, UAB

2023-2024

Index

1. Introduction and goals

The Ironman Triathlon is a one-day multisport competition organized by the World Triathlon Corporation (WTC), and consists of 3.9 km swim, a 180.2 km bicycle ride and a marathon 42.2 km run, completed in that order, and adding up to 226.3 km in total.

The Ironman 70.3, Triathlon, also known as a Half Ironman, is a similar competition where the distance of the swim, bike, and run segments is half the distance of that segment in an Ironman Triathlon. "70.3" refers then to the total distance in miles (113.0 km) covered in the Ironman 70.3 race, split as a 1.9 km swim, a 90 km bike ride, and a 21.1 km run.

This study aims to accomplish the following goals:

- To analyse the evolution of results over the years, in other words to examine how the performance results have changed over time. It involves studying trends, patterns, and any significant shifts in the performance of participants in the dataset.
- To determine critical factors affecting the variation in performance: This objective focuses on identifying the key factors that have a significant impact on the variation in performance marks. It involves analysing the relationship between the performance outcome and the different variables available in the dataset, such as age group, “Gender”, country, event year, event location, swim time, bike time, and run time.
- And finally, to make predictions: This objective involves using statistical modelling techniques, such as linear regression, to develop predictive models. These models can estimate performance outcomes based on the selected variables. The objective is to generate predictions for specific scenarios, such as predicting performance for individuals with specific characteristics (e.g., “Gender”, age) participating in the Ironman event.

To sum up, the study aims to gain insights into the evolution of performance results, identify critical factors influencing performance variation, and provide predictions for performance outcomes based on the available variables in the dataset. These findings can contribute to a better understanding of the factors affecting performance in Ironman events and inform training strategies, event planning, and athlete performance management.

2. About the dataset?

All the information has been obtained from Kaggle, an online platform for data scientists to collaborate, compete, and share resources, including datasets, competitions, and notebooks.

The dataset includes 840,075 records from Ironman 70.3 professional (PRO) and recreational triathletes between 2004 and 2020**,** with information about the participants demographics, performance times for each segment of the event and overall completion time.

1. “Gender”: It represents the “Gender” of the participant in the Ironman event. It is a categorical variable with values "M" for male and "F" for female.
2. “AgeGroup”: It categorizes the participants into specific age groups. It is a categorical variable that provides information about the age range to which each participant belongs.
3. “AgeBand”: It represents the age of the participant.
4. CountryISO2: It represents the ISO 2-letter country code for each participant's country.
5. “EventYear”: It indicates the year in which the Ironman event took place.
6. “EventLocation”: It specifies the location or venue where the Ironman event was held.
7. “SwimTime”: It represents the time taken by the participant to complete the swimming portion of the Ironman event.
8. “Transition1Time”: It represents the time taken by the participant to transition from swimming to cycling.
9. “BikeTime”: It represents the time taken by the participant to complete the cycling portion.
10. “Transition2Time”: It represents the time taken by the participant to transition from cycling to running.
11. “RunTime”: It represents the time taken by the participant to complete the running portion.

12. “FinishTime”: It represents the total time taken by the participant to finish the entire Ironman event.

To facilitate calculations and avoid issues in processing a large volume of data, I initially did two main things:

- Remove the variables Transition1Time, Transition2Time and CountryISO2, which might not be as significant as the others, and in my case, I wouldn’t be using them in my study.
- Performs random sampling on the dataset. I used the function *sample()* that randomly selects 10,000 rows (observations) from the dataset

By removing variables and performing random sampling, we have reduced the size of the dataset to a more manageable sample size. This can be helpful when working with large datasets as it reduces computational requirements and processing time, while still providing insights and analysis on a representative subset of the data.

3. Analysis techniques

In this section, various analysis techniques will be employed to examine and interpret the data.

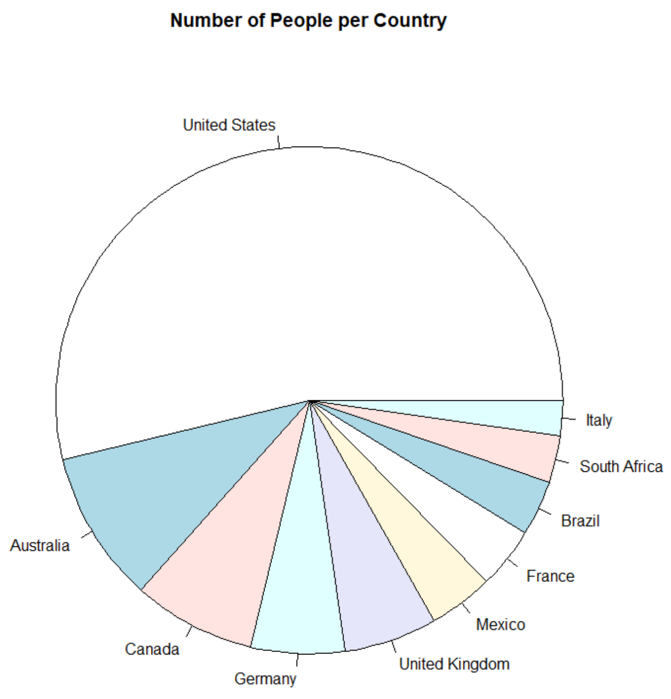
Firstly, we will provide an overview of the techniques employed in our study and explain why we chose to use them.

- **Descriptive Statistics**

  To give insights into the performance and completion times of the athletes in the triathlon event we can use basic R functions to compute the minimum, maximum and mean values of numerical values.

	Minimum	**Maximum**	**Mean**
Swim Time (minutes)	20.016	99.95	39.004
Bike Time (minutes)	108.516	299.883	177.922
Run Time (minutes)	66.7	249.983	126.783
Finish Time (minutes)	216.733	608.816	352.150

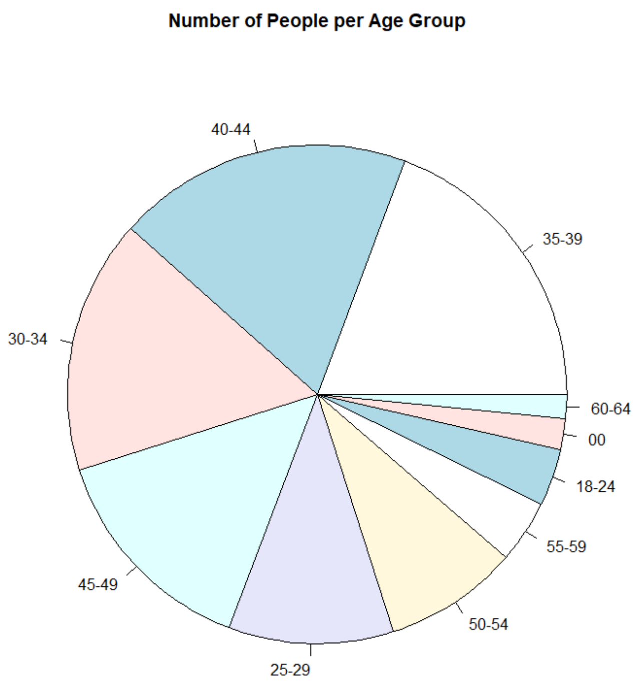
We can analyse the distribution of people based on the different variables in the dataset:



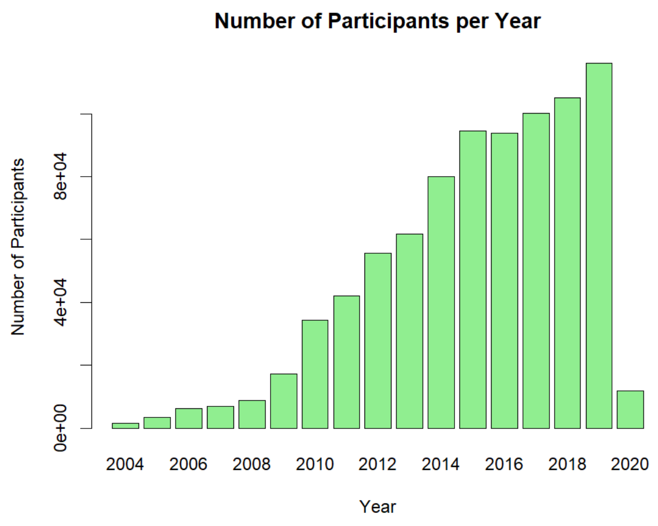
Distribution of people across different countries in a dataset: Analysing the output, we can observe that the United States has the highest frequency, followed by Australia, Canada, Germany, United Kingdom, Mexico, France, Brazil, South Africa, and Italy.

Distribution of people across different age groups in a dataset:

From the output, we can see that the age group "35-39" has the highest frequency, followed by "40-44". These are the most prevalent age groups in the dataset, while groups with lower frequency are "55-59," "18-24," "00" (professionals), and "60-64". It’s remarkable that the number of professional participants is really low compared to other age groups, as low as the highest age group (“60-64”).



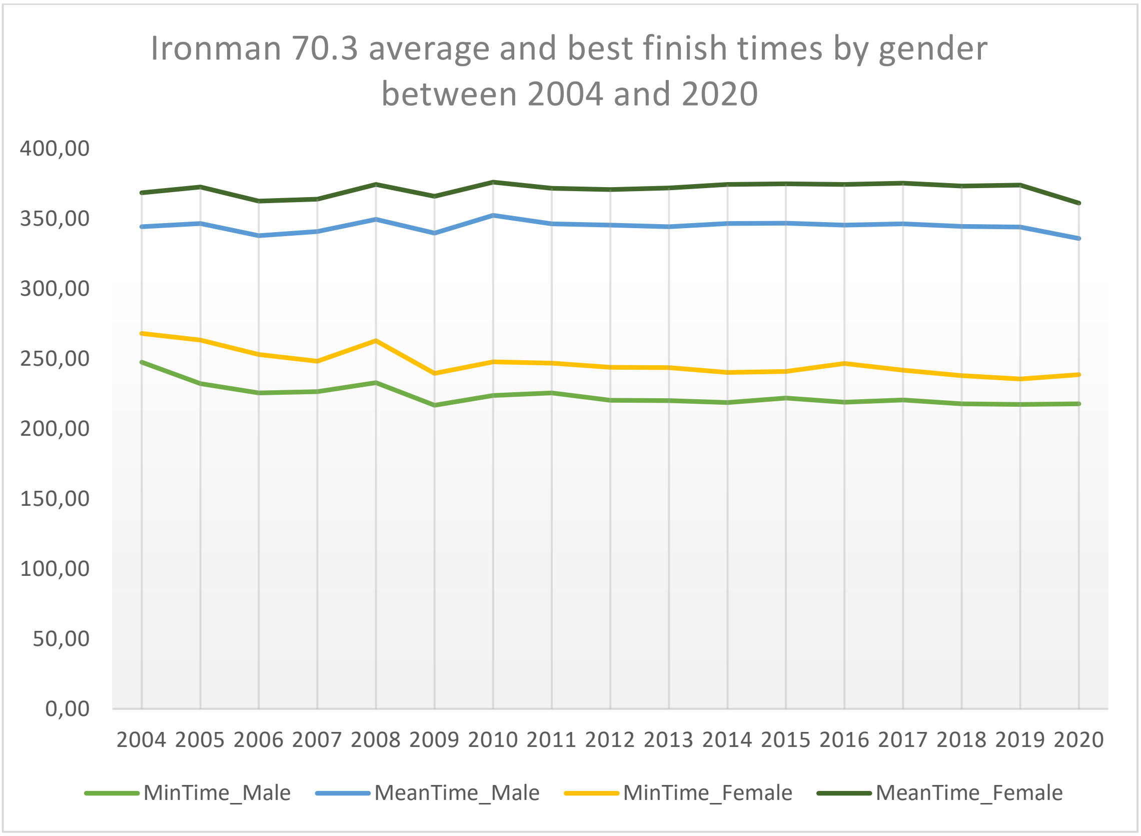
To obtain insights into the trend and growth of participant numbers over time we can calculate and visualize the number of participants each year in a bar plot. This information provides valuable insights into the popularity and interest in Ironman events across different years.



The trend shows a steady increase in the number of participants from 2008 to 2019, with notable jumps in certain years. It indicates the growing popularity and participation in Ironman competitions during this period.

In 2020, there was a significant decrease in the number of participants compared to previous years. The data shows that only 11,804 participants took part in the Ironman competitions in 2020. This decrease can be attributed to the impact of the COVID-19 pandemic, which resulted in the cancellation or postponement of many sporting events worldwide, including the Ironman competitions. The pandemic had widespread effects on travel, gathering restrictions, and safety concerns, leading to a significant decline in participation in various events, including the Ironman competitions.

In order to have a clear vision of the trends in performance of males and females through the years, using Excel we can visualize all the data in the following graphic.



In the year 2008, there is a noticeable increase in both the minimum and mean times for both males and females. This suggests that, on average, participants took longer to complete the Half Ironman race in 2008 compared to the previous years.

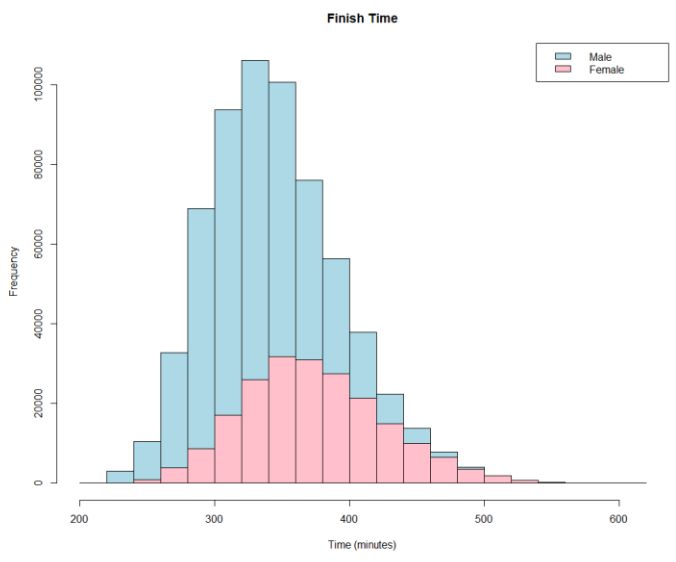
There could be various reasons for this increase in race times. Some possible explanations could include:

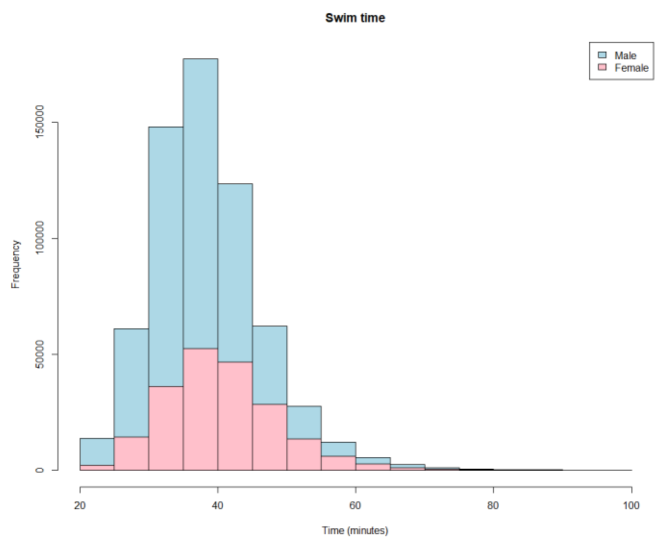
1. Weather conditions: Unfavourable weather conditions such as high temperatures, strong winds, or heavy rain during the 2008 race could have contributed to slower performance.
2. Participant demographics: The participants in the 2008 race might have had different characteristics or experience levels compared to previous years, leading to slower overall performance.
3. Training and preparation: It's also possible that the overall level of training and preparation among participants in 2008 was not as high as in previous years, resulting in slower race times.

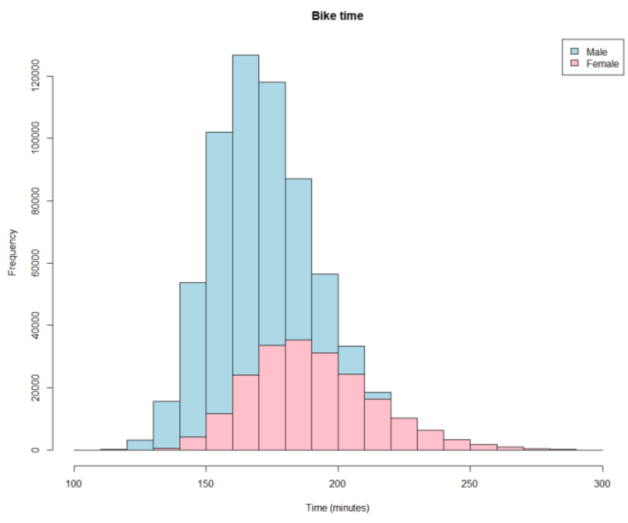
However, looking for information on the internet I found that in 2008 there was a global economic crisis. The global economic crisis in 2008 had significant repercussions across various aspects of society, including the world of sports. Its effects on the Ironman competitions can be observed in several ways. Firstly, there could be a potential decline in sponsorships and financial support, which may impact the resources available for organizing the events. Additionally, the crisis could lead to a decrease in athlete participation as individuals face economic hardships and may be unable to afford the expenses associated with participating in Ironman races. This decline in participation could affect the overall competition and the level of performance among athletes. Finally, the economic limitations faced by athletes during the crisis may hinder their access to necessary training resources and services, influencing their physical preparation and, subsequently, their results.

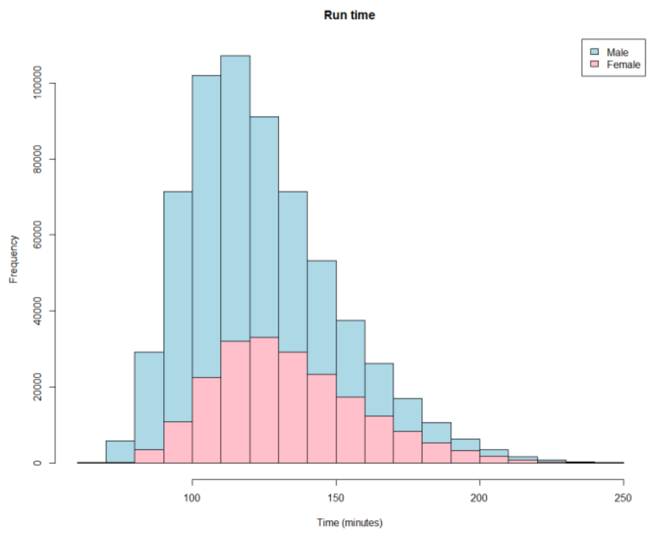
Finally, creating histograms of partial and full race times, separated by “Gender”, we can visualize the distribution of race times for male and female participants in each segment (finish time, swim time, bike time, run time) of the Ironman event.

By plotting histograms, we can examine the shape, central tendency, and variability of the race time distributions for different “Gender”s. This allows us to identify any patterns or differences between male and female participants in terms of their performance.
Overall, these histograms help us gain insights into the distribution and characteristics of race times, providing a visual representation of the performance levels of male and female participants in different segments of the Ironman event.









Interpretation of histograms of each section of Ironman 70.3

a. Finish time

The mean finish time for males (345.4 minutes) is slightly lower than the mean finish time for females (373.3 minutes), suggesting that, on average, male participants complete the race in less time than female participants.

The range of finish times for males (216.7 minutes to 608.8 minutes) is larger than the range for females (235.5 minutes to 598.3 minutes), indicating a wider spread of finish times among male participants.

Overall, the male participants seem to have faster finish times compared to the female participants.

b. "SwimTime"

The mean swim time for males (38.50 minutes) is slightly lower than the mean swim time for females (40.56 minutes), suggesting that, on average, male swimmers complete the swim portion of the race in less time than female swimmers.

The range of swim times for males (20.02 minutes to 99.95 minutes) is larger than the range for females (22.02 minutes to 99.93 minutes), indicating a wider spread of swim times among male swimmers.

Overall, the male swimmers seem to have faster swim times compared to the female swimmers.

c. “BikeTime”

The mean bike time for females (190.4 minutes) is slightly lower than the mean bike time for males (173.9 minutes), suggesting that, on average, female cyclists complete the bike portion of the race in less time than male cyclists.
The range of bike times for males (108.5 minutes to 299.9 minutes) is larger than the range for females (121.5 minutes to 299.8 minutes), indicating a wider spread of bike times among male cyclists.

d. “RunTime”

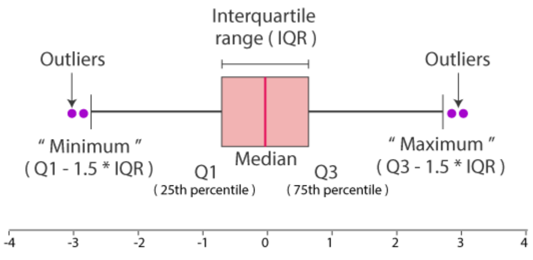
Based on the summary statistics, it appears that male runners have slightly faster run times compared to female runners.
The mean run time for males (124.6 minutes) is lower than the mean run time for females (133.69 minutes), suggesting that, on average, male runners complete the run portion of the race in less time than female runners.
The range of run times for males (66.7 minutes to 250.0 minutes) is larger than the range for females (70.25 minutes to 249.72 minutes), indicating a wider spread of run times among male runners.

- **Race performance by age group and “Gender” (box plot)**

To provide a clear and concise representation of race performance by age group and “Gender”, enabling quick comparisons, trend identification, outlier detection, and data exploration we can use a boxplot representation.

Boxplots are helpful to display and compare the range of values of several groups. Each box represents five key values:

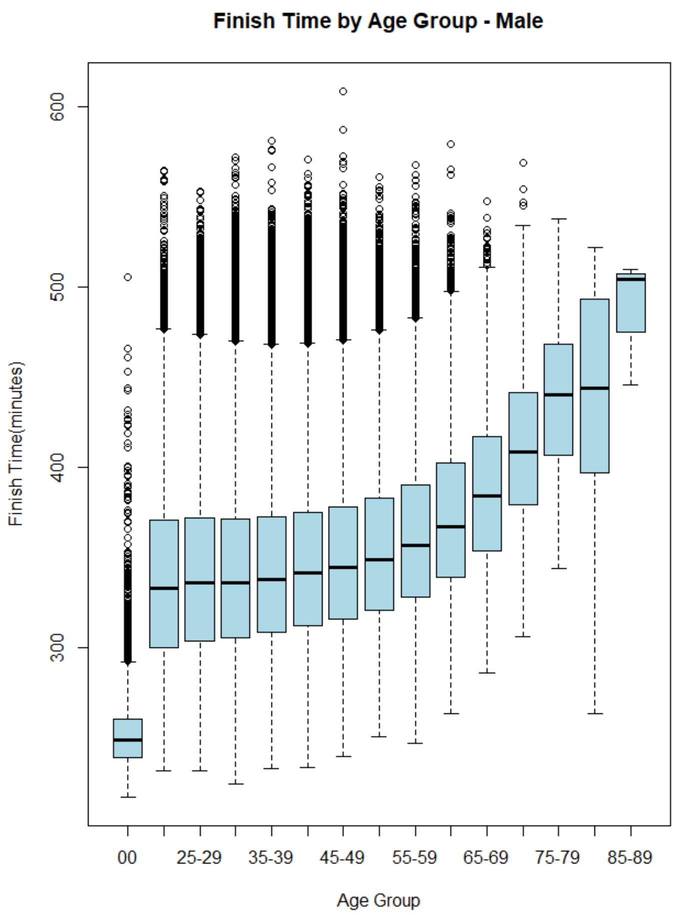
- median (50% percentile, we assume this is the mean when the data has a normal distribution)
- 25% percentiles
- 75% percentiles
- maximum
- minimum

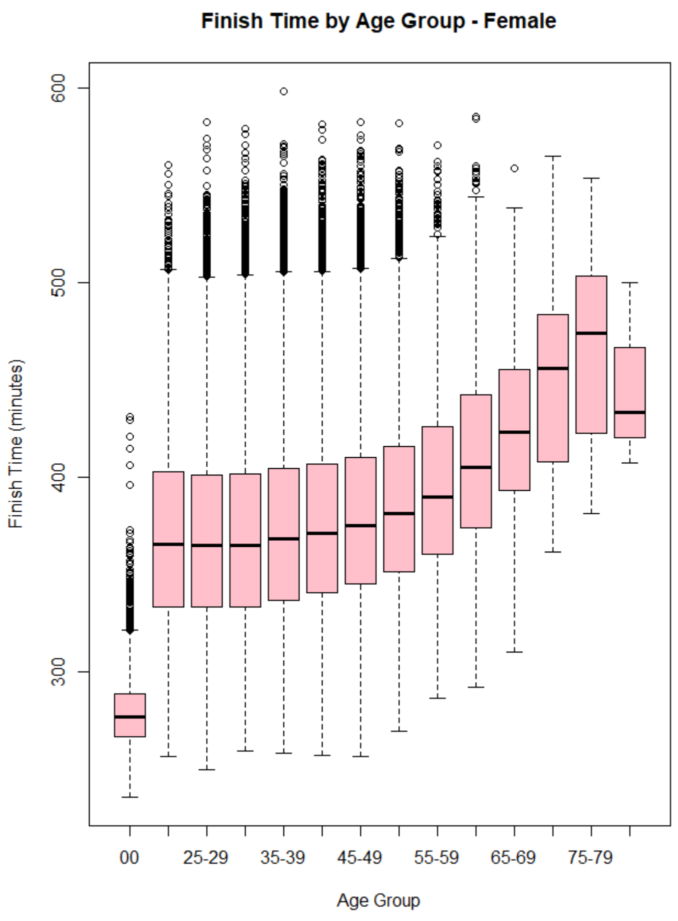


Interquartile range (IQR)
Outliers
Outliers
“ Minimum ” (Q1 - 1.5 * IQR)
Q1 (25th percentile)
Median
Q3 (75th percentile)
“ Maximum ” (Q3 - 1.5 * IQR)
-4
-3
-2
-1
0
1
2
3
4

Different parts of boxplot

The graphic obtained when we run the R code is the following:





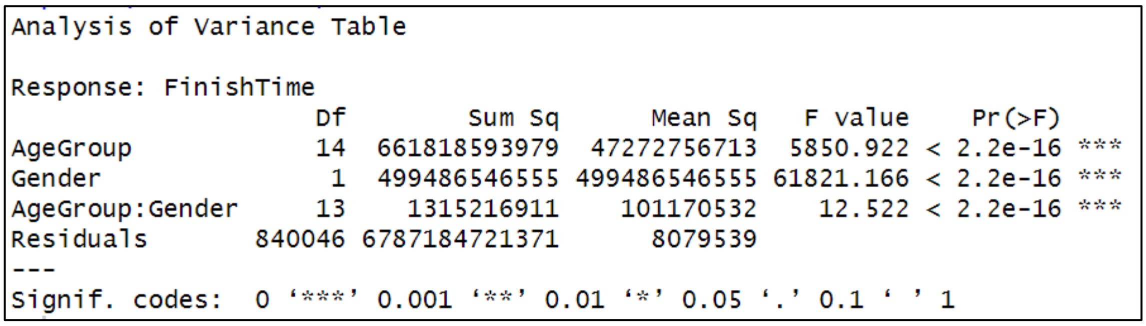
The difference in performance between the PRO group (age group 0) and the age groups is enormous (over 1hr 15 min), but then performance remains almost flat for age groups between 18 and 45.

- **Two way ANOVA test**

The objective of fitting a two-way ANOVA (Analysis of Variance) model and conducting an ANOVA test is to analyse the effects of two categorical variables, “AgeGroup” and “Gender”, on the continuous response variable “FinishTime”.

The *anova()* function calculates the analysis of variance and returns an ANOVA table. The table provides information about the significance of each variables and helps determine whether there are statistically significant differences in “FinishTime” based on the levels of “AgeGroup”, “Gender”, and their interaction.

Once the test has been made, we obtain the following ANOVA table:



```
Analysis of Variance Table

Response: FinishTime
                    Df        Sum Sq      Mean Sq   F value    Pr(>F)    
AgeGroup            14  661818593979  47272756713  5850.922 < 2.2e-16 ***
Gender               1  499486546555 499486546555 61821.166 < 2.2e-16 ***
AgeGroup:Gender     13    1315216911    101170532    12.522 < 2.2e-16 ***
Residuals       840046 6787184721371      8079539                        
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

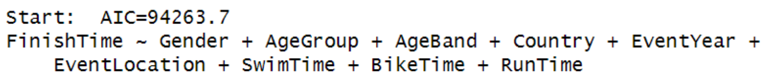
Based on this data table we can confirm that:

- The analysis indicates that “AgeGroup” has a significant effect on “FinishTime” (p < 2.2e-16). In other words, there are statistically significant differences in “FinishTime” across different age groups.
- “Gender” also has a significant effect on “FinishTime” (p < 2.2e-16). This means that there are statistically significant differences in “FinishTime” between males and females.
- The interaction between “AgeGroup” and “Gender” significantly affects “FinishTime” (p < 2.2e-16). This suggests that the relationship between “AgeGroup” and “FinishTime” may vary depending on the “Gender”.
- The residuals represent the unexplained variability in the data after considering the effects of “AgeGroup”, “Gender”, and their interaction. In this case, the mean square of residuals is 8,079,539, which indicates a moderate amount of unexplained variability in the “FinishTime” variable after accounting for the effects of “AgeGroup”, “Gender”, and their interaction.

In summary, the ANOVA results indicate that individually, both “AgeGroup” and “Gender” have significant effects on “FinishTime” when considered alone or combined. Such results suggest that “AgeGroup”, “Gender”, and their interaction are important factors to consider when analyzing the variability in “FinishTime”.

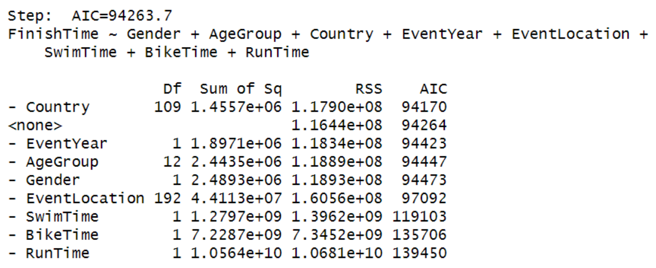
- **Backward variable selection**

The goal is to find the most prudent model that achieves a balance between goodness of fit and model complexity.



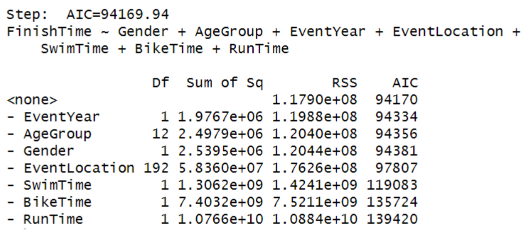
```
Start:  AIC=94263.7
FinishTime ~ Gender + AgeGroup + AgeBand + Country + EventYear + 
    EventLocation + SwimTime + BikeTime + RunTime
```

In the initial step, the full model is considered, including all predictor variables (“Gender”, “AgeGroup”, “AgeBand”, Country, “EventYear”, “EventLocation”, “SwimTime”, “BikeTime”, and “RunTime”). The AIC for this model is 94226.1.



```
Step:  AIC=94263.7
FinishTime ~ Gender + AgeGroup + Country + EventYear + EventLocation + 
    SwimTime + BikeTime + RunTime

                Df  Sum of Sq        RSS    AIC
- Country      109 1.4557e+06 1.1790e+08  94170
<none>                        1.1644e+08  94264
- EventYear      1 1.8971e+06 1.1834e+08  94423
- AgeGroup      12 2.4435e+06 1.1889e+08  94447
- Gender         1 2.4893e+06 1.1893e+08  94473
- EventLocation 192 4.4113e+07 1.6056e+08  97092
- SwimTime       1 1.2797e+09 1.3962e+09 119103
- BikeTime       1 7.2287e+09 7.3452e+09 135706
- RunTime        1 1.0564e+10 1.0681e+10 139450
```



```
Step:  AIC=94169.94
FinishTime ~ Gender + AgeGroup + EventYear + EventLocation + 
    SwimTime + BikeTime + RunTime

                Df  Sum of Sq        RSS    AIC
<none>                        1.1790e+08  94170
- EventYear      1 1.9767e+06 1.1988e+08  94334
- AgeGroup      12 2.4979e+06 1.2040e+08  94356
- Gender         1 2.5395e+06 1.2044e+08  94381
- EventLocation 192 5.8360e+07 1.7626e+08  97807
- SwimTime       1 1.3062e+09 1.4241e+09 119083
- BikeTime       1 7.4032e+09 7.5211e+09 135724
- RunTime        1 1.0766e+10 1.0884e+10 139420
```

In the subsequent steps, the algorithm removes one predictor variable at a time and compares the resulting models. The variable with the highest AIC value is removed, and the process continues until no further improvement in the AIC can be achieved. In this case, the variable "Country" is removed in the first step, resulting in a model with an AIC of 94139.92.

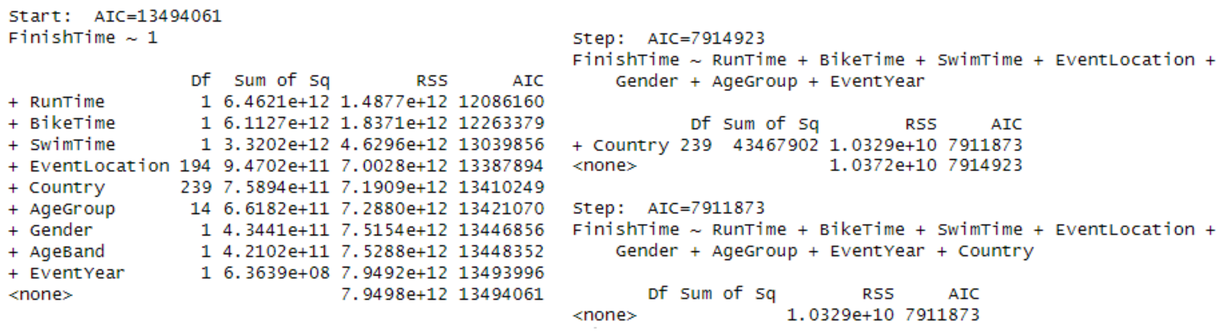
The final selected model includes the remaining variables: “Gender”, “AgeGroup”, “EventYear”, “EventLocation”, “SwimTime”, “BikeTime”, and “RunTime”. This model has an AIC of 94139.92 and an associated residual sum of squares (RSS) of 1.1752e+08.

- **Forward Selection**

Forward variable selection gradually builds the model by adding variables, while backward variable selection simplifies the model by eliminating variables. Forward selection allows for a clear understanding of the impact of each predictor, while backward variable selection helps identify non-significant predictors.

In summary, forward selection is a stepwise variable selection technique that iteratively adds variables to the model based on the lowest Akaike Information Criterion (AIC) value, aiming to find the best-fitting model.

Once the process has been completed, we obtained these results:



```
Start:  AIC=13494061
FinishTime ~ 1

                 Df  Sum of Sq        RSS      AIC
+ RunTime         1 6.4621e+12 1.4877e+12 12086160
+ BikeTime        1 6.1127e+12 1.8371e+12 12263379
+ SwimTime        1 3.3202e+12 4.6296e+12 13039856
+ EventLocation 194 9.4702e+11 7.0028e+12 13387894
+ Country       239 7.5894e+11 7.1909e+12 13410249
+ AgeGroup       14 6.6182e+11 7.2880e+12 13421070
+ Gender          1 4.3441e+11 7.5154e+12 13446856
+ AgeBand         1 4.2102e+11 7.5288e+12 13448352
+ EventYear       1 6.3639e+08 7.9492e+12 13493996
<none>                         7.9498e+12 13494061
```

```
Step:  AIC=7914923
FinishTime ~ RunTime + BikeTime + SwimTime + EventLocation + 
    Gender + AgeGroup + EventYear

           Df Sum of Sq        RSS     AIC
+ Country 239  43467902 1.0329e+10 7911873
<none>                  1.0372e+10 7914923

Step:  AIC=7911873
FinishTime ~ RunTime + BikeTime + SwimTime + EventLocation + 
    Gender + AgeGroup + EventYear + Country

       Df Sum of Sq        RSS     AIC
<none>                1.0329e+10 7911873
```

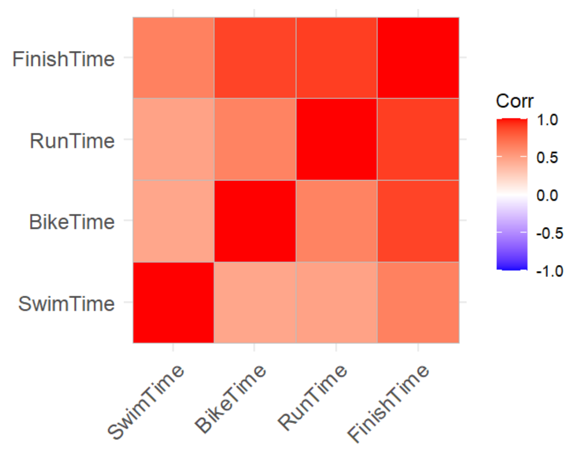
We can observe that, as using the backward variable selection the final selected model includes: “Gender”, “AgeGroup”, “EventYear”, “EventLocation”, “SwimTime”, “BikeTime”, and “RunTime”.

- **Correlation matrix**

The objective of performing a correlation analysis and visualizing the correlation matrix using the ggcorrplot function is to understand the relationships between the variables in the dataset.

By calculating the correlation matrix, we obtain a square matrix where each cell represents the correlation coefficient between two variables. The correlation coefficient ranges from -1 to 1, with values closer to 1 indicating a strong positive correlation, values closer to -1 indicating a strong negative correlation, and values close to 0 indicating a weak or no correlation.

Visualizing the correlation matrix provides a graphical representation of the correlations, making it easier to identify patterns and relationships. The plot shows a matrix of squares, where each square represents the correlation between two variables. The visualization of our matrix is the following:

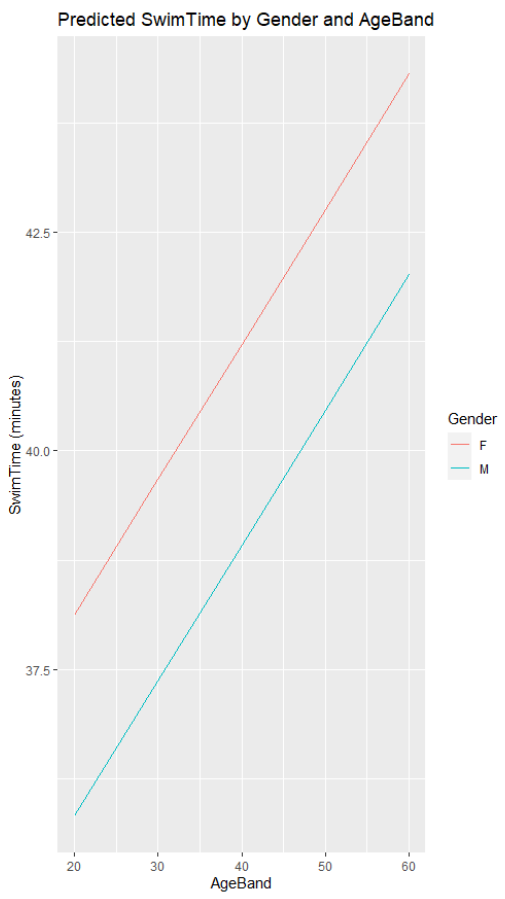


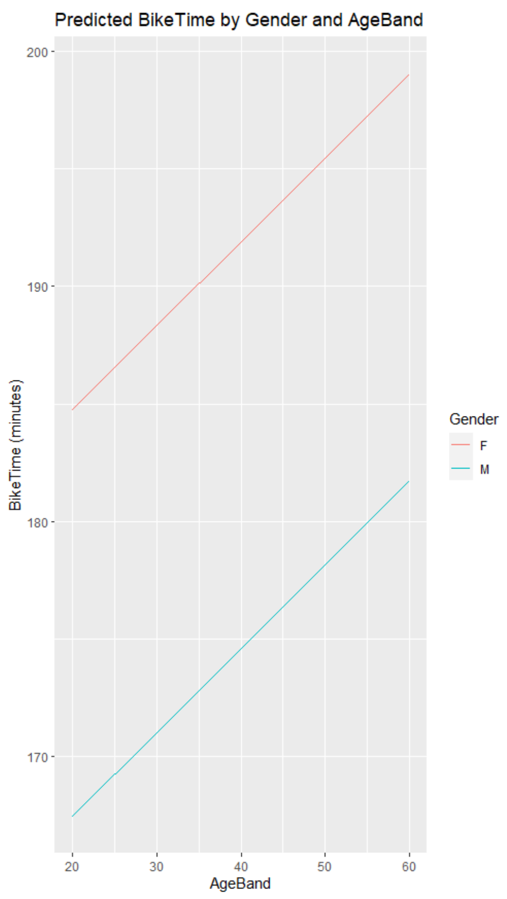
Conclusions we can take from correlation matrix:

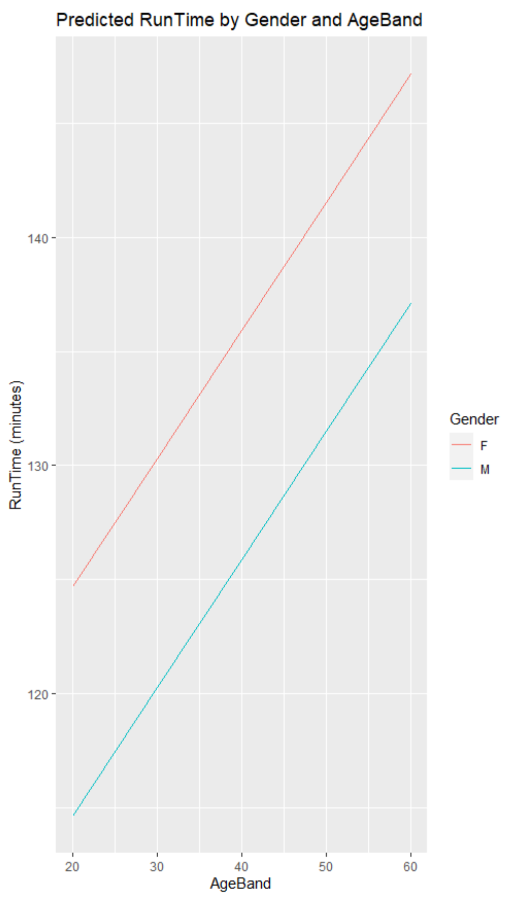
- Each individual discipline has a low effect in others and only has important effect of the final time.
- The strongest correlation effect in final time are "BikeTime" (0,87) and even more "RunTime" (0,90) while swim time is not really significant in the final time. This data can indicate to an athlete that wants to obtain good performance in an Ironman, which discipline must be trained the most.

- **Predict function**

To estimate and visualize the predicted "SwimTime", "BikeTime", and "RunTime" values based on different linear regression models and different combinations of "Gender" and "AgeBand" we can use the function *predict* and we obtain the following graphics.







These allow us to visualize the relationship between "SwimTime", "BikeTime", and "RunTime" with respect to "Gender" and "AgeBand". It helps us understand how the different times change across "AgeBand" and if there are any variations between "Gender" within each "AgeBand".

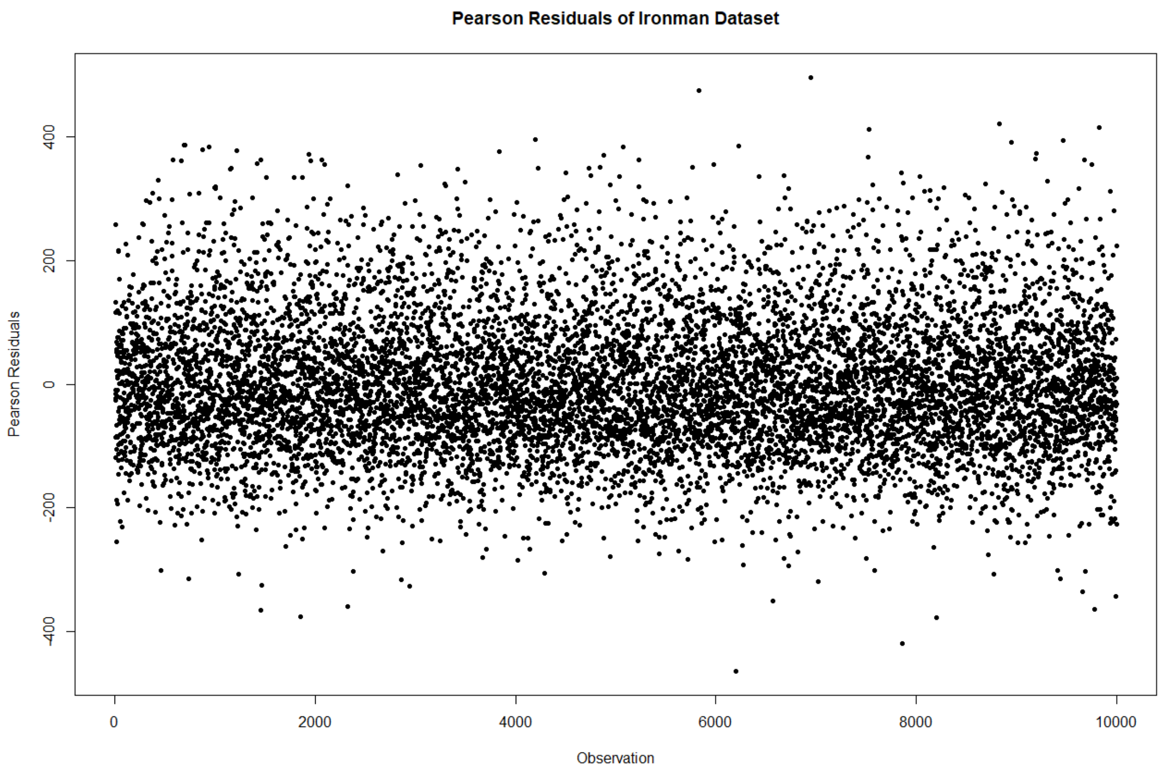
As an example, the results of the predictions for a 20-year-old girl participating in an Ironman70.3 event are as follows:

Swim Time	38.46 minutes
Bike Time	185.26 minutes
Run Time	125.32 minutes
Finish Time	356.69 minutes = 5 hours and 95 minutes

- **Pearson residuals**

Residuals represent the difference between the observed values and the predicted values of the response variable. The plot allows us to assess the pattern and distribution of these residuals.
Each observation represents a specific participant in an Ironman event and contains information about their “Gender”, age, country, event year, event location, swim time, bike time, run time, and finish time. The residual for each observation is calculated as the observed “FinishTime” minus the predicted “FinishTime” from the regression model.

The residuals can provide insights into the model's performance and the quality of the fit. Positive residuals indicate that the actual “FinishTime” values are higher than the predicted values, while negative residuals indicate that the actual values are lower than the predicted values.

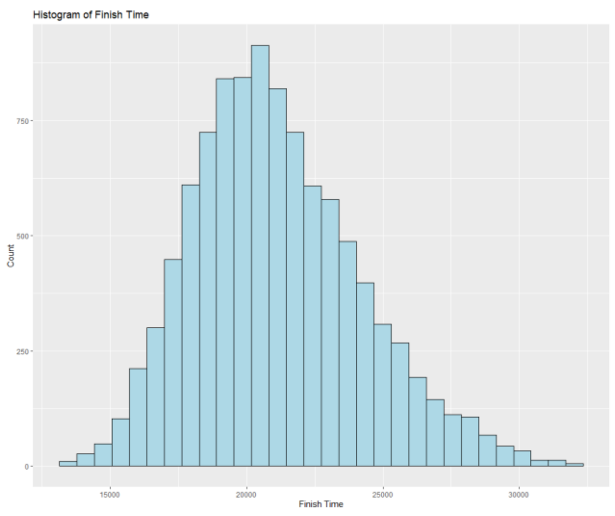


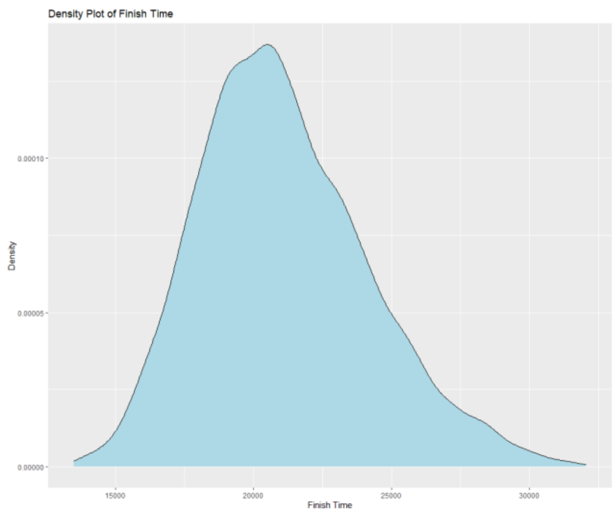
The maximum Pearson residual value of 496.1538 indicates the largest discrepancy between the observed “FinishTime” and the predicted values based on the model.

This residual is associated with the observation 6943 that corresponds to a thirty-year-old male from the United States. He participated in the IRONMAN 70.3 Eagleman in 2013, with a finish time of 354.3667 minutes.
The performance of this athlete compares with others from the same range of "Gender", age and so on, exceeds approximately thirty minutes from the expected value.

- **Parametric bootstrap**

The parametric bootstrap is a resampling technique used to estimate the sampling distribution of a statistic when the underlying distribution is known or can be assumed.
First, we need to identify which distribution the variable "FinishTime" follows. To do that we are going to generate some plots to visually inspect its distribution.

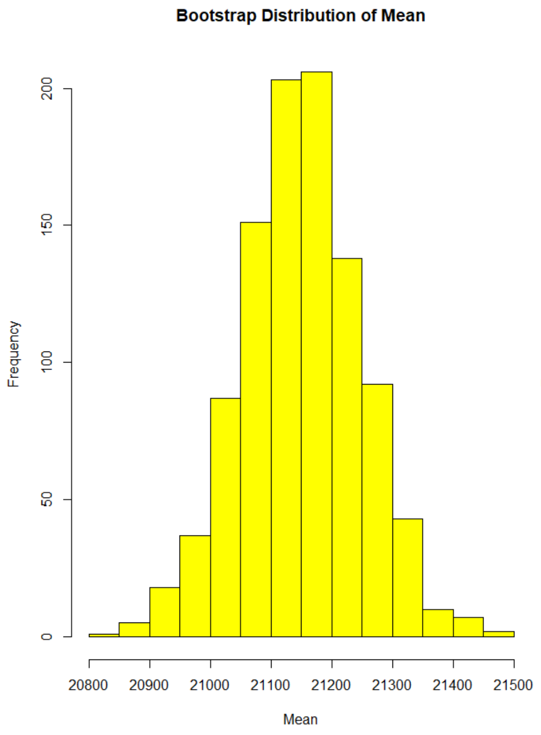


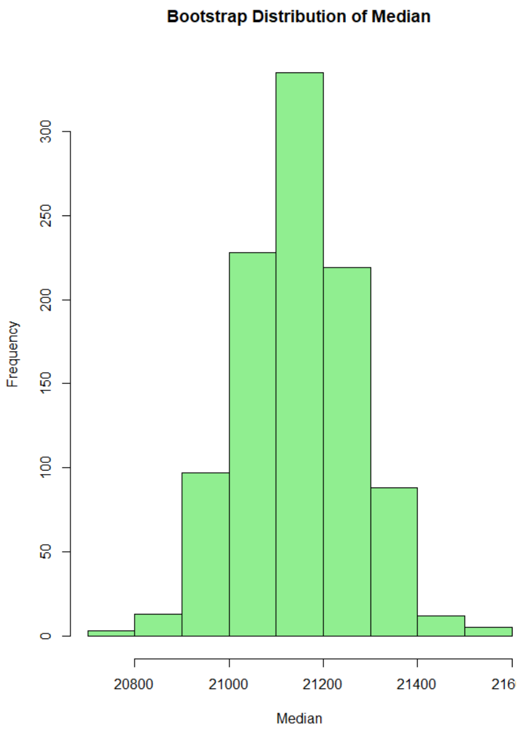


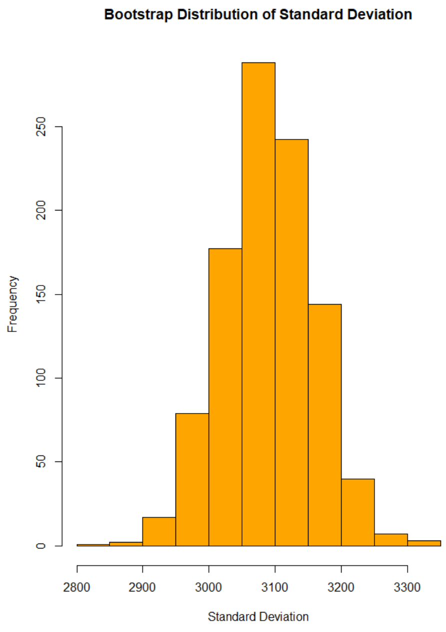
We can observe that its density function is very similar to a Normal distribution, so we are going to assume that the variable "FinishTime" follows a Normal distribution

When performing the bootstrap, multiple samples of the same length as the original sample are generated through sampling with replacement. For each bootstrap sample, the corresponding mean is calculated. These bootstrap mean values are used to construct the bootstrap distribution of the mean.

Now we can perform a parametric bootstrap analysis by resampling from a Normal distribution assumed to represent the "FinishTime" variable. Then, we can calculate bootstrap statistics (mean, median, and standard deviation) from the generated samples and visualize their distributions using histograms.







We observe that, since we assumed that the “FinishTime” variable follows a Normal distribution, the histogram exhibits a symmetric bell-shaped curve.

Additionally, the central tendency of the bootstrap distribution of the different statistics can be assessed by examining the peak or highest point of the histogram. It represents the most probable value of the mean estimate. Moreover, the variability of the bootstrap statistics can be evaluated by considering the width of the histogram or the dispersion of the bars. A wider histogram indicates higher variability, while a narrower histogram suggests lower variability.

Finally, we can compute bootstrap confidence intervals for each statistic to provide a measure of uncertainty.

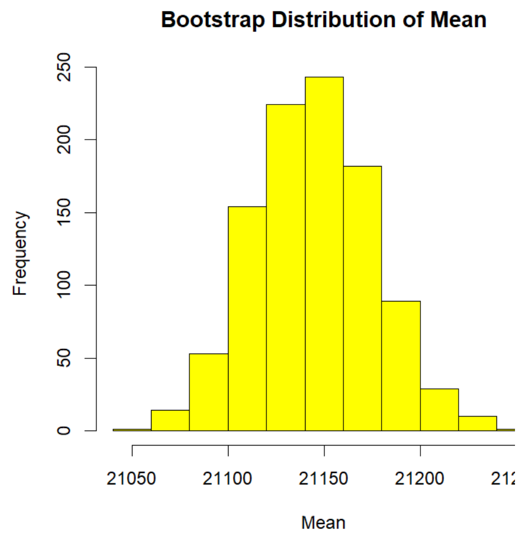
	2.5%	97.5%
Mean	20939.04	21332.15
Median	20902.50	21369.35
Standard Deviation	2947.767	3229.031

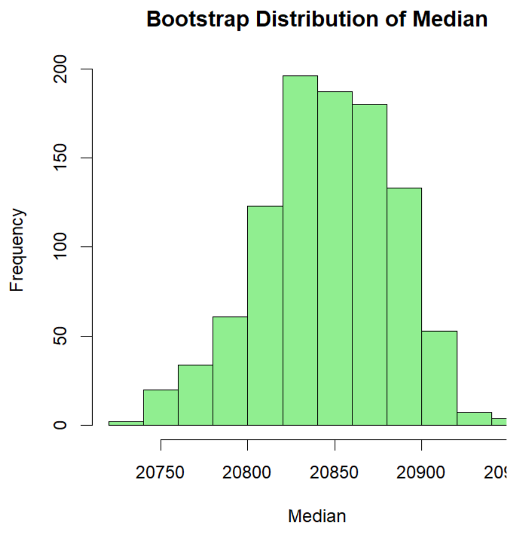
With a 95% confidence level, the true population mean, median, or standard deviation is expected to fall within the corresponding confidence interval.

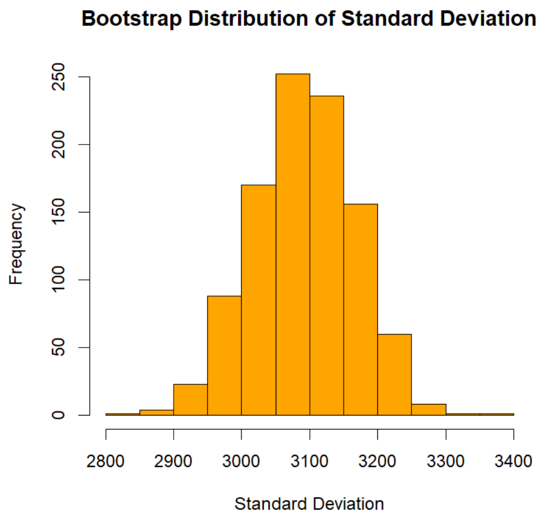
- **Non-parametric bootstrap**

To perform a non-parametric bootstrap, we don't make any assumptions about the underlying distribution of the “FinishTime” variable. Instead, we resample the data with replacement to create bootstrap samples and calculate the desired statistics from these samples.

Once we have performed the non-parametric bootstrap, we obtain the following results:







Based on dataset under study, results obtained in parametric and non-perimetric bootstrap are comparable.

Finally, we can compute bootstrap confidence intervals for each statistic to provide a measure of uncertainty.

	2.5%	97.5%
Mean	21012.85	21128.74
Median	20700.97	20848.10
Standard Deviation	3033.282	3122.074

4 Results of analysis

Overall results of the analysis of the data, including the relevant statistics and graphics

This study aims to analyse the evolution of performance results in Ironman events over time, identify critical factors influencing performance variation, and provide predictions for performance outcomes based on the available variables in the dataset.

In terms of descriptive statistics, some important things that I found interesting are:

- The United States AgeGroup "35-39" has the highest frequency of participants. Additionally, it is remarkable that there are very few professional athletes. Something I didn’t expect.
- The number of participants in Ironman events showed a steady increase from 2008 to 2019, indicating growing popularity and participation. However, there was a significant decrease in participation in 2020 due to the impact of the COVID-19 pandemic.
- Although increase of participants is observed, performance of athletes does not have a significative improvement.
- The analysis of performance trends by year showed variations in race times for both males and females. In 2008, there was an increase in race times, which could be attributed to the global economic crisis mentioned before.
- From the histograms, used to visualize the distribution of race times for male and female participants in different segments of the Ironman event, we can say that overall, male participants seemed to have faster times compared to female participants.
- BikeTime and RunTime seem to be the parts with higher effect on final time. So, if a starter in Triathlon wants to have a good mark and has not much time for training, better focus training in running or biking.

5 Conclusions and discussion.

Regarding the used techniques, each one is useful for different purposes.

Descriptive statistics provide summary measures and visualizations that help us understand the main characteristics of the data, such as the central tendency (mean, median), variability (range, standard deviation), and distribution shape.

Backward and Forward variable selection allows us to identify the most important variables that contribute significantly to the dependent variable, improving the model's simplicity and interpretability.

Correlation matrix I think it is a very practical tool if you want to participate in an Ironman. The matrix tells you in which section you might need to focus more on, because it has a big effect on the final time. Together with the correlation matrix, I believe that the *prediction function,* and the different *graphics* provided before are also really helpful to make a prediction of you participation in an Ironman 70.3.

Predictive model developed is valid and the error in the predicted final time will not be higher than 20 - 30 min from predicted value according to selected variables as show in the *Pearson Residuals* distribution plot.

Concerning some problems I had with the large amount of data, I found two different solutions. On the one hand, my first option was to use the *library biglm,* which we saw in class. The *biglm()* function is used when dealing with large datasets that do not fit into memory. It is specifically designed for fitting linear regression models to such datasets. It was a good idea because it minimizes the memory requirements and it significantly sped up the analysis compared to traditional linear regression functions, such as *lm()* function. However, I had some problems with working with the fitted model when performing other techniques.

On the other hand, another alternative was to reduce the dataset size and make it more manageable for analysis. To do that I randomly sampled the original dataset using the *sample()* function. The *sample()* function selects a specified number of rows (size = 10000) from the dataset without replacement (replace = FALSE).

6 Bibliography

- Ironman 70.3 race data between 2004 and 2020 | Kaggle. https:///datasets/aiaiaidavid/ironman-703-race-data-between-2004-and-2020
- Ironman 70.3 - Wikipedia. (2010, November 9) https://en.wikipedia.org/wiki/Ironman_70.3
- R Box Plot (With Examples). https://www.datamentor.io/r-programming/box-plot
- Crisis financiera de 2007-2008 - Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Crisis_financiera_de_2007-2008
- Prajapati, C. (2019, March 8). Bootstrap R tutorial : Learn about parametric and nonparametric bootstrap through simulation. Medium. https://medium.com/@statisticianonline/bootstrap-r-tutorial-learn-about-parametric-and-nonparametric-bootstrap-through-simulation-f6be4d4fc532
- Principal Component Analysis in R https://www.datacamp.com/tutorial/pca-analysis-r

7 Appendix with the R script

```r
#FINAL ASSIGNMENT

#Alba Puig Font
#NIU: 1636034

# Set working directory and load required packages
setwd("C:/Users/albap/OneDrive/Escriptori/UNIVERSITAT/Anàlisi de Dades 
Complexes/FINAL ASSIGNMENT")
library("readxl")

# Read the data from CSV file
data <- read.csv("Half_Ironman_df6.csv")

# Remove variables
data <- subset(data, select = -c(Transition1Time, Transition2Time, 
CountryISO2))
summary(data)
head(data)
View(data)

#-----------------------------------------------------------------------
#FIT THE MODEL
#-----------------------------------------------------------------------
# Perform random sampling to reduce the dataset size
sampled_data <- data[sample(nrow(data), size = 10000, replace = 
FALSE),]

# View the summary of the sampled data
summary(sampled_data)

#Now, we can design a model to be analysed for the dataset:
model_full<-lm(FinishTime ~.,data=sampled_data)
summary(model_full)

# Fit the full model using BIGLM
library(biglm)
model_full_biglm <- biglm(formula = FinishTime ~ Gender + AgeGroup + 
AgeBand + Country + EventYear + EventLocation + SwimTime + BikeTime + 
RunTime, data = data)

#------------------------------------------------------------
# Descriptive statistics
#------------------------------------------------------------

#Maximum, minimum and Mean of the values

#SwimTime
min(data$SwimTime)/60
max(data$SwimTime)/60
mean(data$SwimTime)/60

#BikeTime
min(data$BikeTime)/60
max(data$BikeTime)/60
mean(data$BikeTime)/60

#RunTime
min(data$RunTime)/60
max(data$RunTime)/60
```

```
mean(data$RunTime)/60

#FinishTime
min(data$FinishTime)/60
max(data$FinishTime)/60
mean(data$FinishTime)/60

#---------------------------------------
#Frequency of countries

# Calculate the frequency of each country
country_freq <- table(data$Country)

# Sort the frequencies in descending order
sorted_freq <- sort(country_freq, decreasing = TRUE)

# Subset the sorted frequencies table to include only the top N
countries
top_countries <- head(sorted_freq, n = 10)
top_countries
# Create a pie chart of the number of people per country for the top
countries
pie(top_countries, main = "Number of People per Country", labels =
names(top_countries))

#----------------------------------------------
#Frequency of AgeGroups

# Calculate the frequency of each country
agegroup_freq <- table(data$AgeGroup)

# Sort the frequencies in descending order
sorted_freq <- sort(agegroup_freq, decreasing = TRUE)

# Select only the top 5 countries
top10_agegroup <- sorted_freq[1:10]
top10_agegroup

# Create a bar plot of the number of Ironman competitions per country
pie(top10_agegroup, main = "Number of People per Age Group", labels =
names(top10_agegroup))

#------------------------------------------------
#Number of participants each year

# Calculate the number of participants each year
participants_per_year <- table(data$EventYear)

# Create a bar plot of the number of participants per year
barplot(participants_per_year, main = "Number of Participants per
Year", xlab = "Year", ylab = "Number of Participants",col =
"lightgreen")

#-----------------------------------------------
#Histograms of partial and full race times

#FINISH TIME
male_finish_times<-(data$FinishTime[data$Gender == "M"])/60
female_finish_times<-data$FinishTime[data$Gender=="F"])/60

summary(male_finish_times)
summary(female_finish_times)

hm<-hist(male_finish_times)
hf<-hist(female_finish_times)

# Plot the male histogram
plot(hm, col = 'lightblue',xlab = "Time (minutes)", main = "Finish
Time")

# Add the female histogram to the plot
plot(hf, col = 'pink', add = TRUE)
legend("topright", c("Male","Female"),fill = c("lightblue","pink"))

#SWIM TIME
male_swim_times <- (data$SwimTime[data$Gender == "M"])/60
female_swim_times <- (data$SwimTime[data$Gender == "F"])/60

summary(male_swim_times)
summary(female_swim_times)

hm<-hist(male_swim_times)
hf<-hist(female_swim_times)

# Plot the male histogram
plot(hm, col = 'lightblue',xlab = "Time (minutes)", main = "Swim
time")

# Add the female histogram to the plot
plot(hf, col = 'pink', add = TRUE)
legend("topright",c("Male","Female"),fill=c("lightblue", "pink"))


#BIKE TIME
male_bike_times <- (data$BikeTime[data$Gender == "M"])/60
female_bike_times <- (data$BikeTime[data$Gender == "F"])/60

summary(male_bike_times)
summary(female_bike_times)

hm<-hist(male_bike_times)
hf<-hist(female_bike_times)

# Plot the male histogram
plot(hm, col = 'lightblue',xlab = "Time (minutes)", main = "Bike
time")

# Add the female histogram to the plot
plot(hf, col = 'pink', add = TRUE)
legend("topright",c("Male","Female"),fill=c("lightblue", "pink"))


#RUN TIME
male_run_times <- (data$RunTime[data$Gender == "M"])/60
female_run_times <- (data$RunTime[data$Gender == "F"])/60

summary(male_run_times)
summary(female_run_times)

hm<-hist(male_run_times)
hf<-hist(female_run_times)
```

```
# Plot the male histogram
plot(hm, col = 'lightblue',xlab = "Time (minutes)", main = "Run time")

# Add the female histogram to the plot
plot(hf, col = 'pink', add = TRUE)
legend("topright",c("Male","Female"),fill=("lightblue", "pink"))

#-------------------------------------------------------
#Race performance by age group and gender

# Create a subset of data for males and females
male_data <- data[data$Gender == "M", ]
female_data <- data[data$Gender == "F", ]

# Create separate box plots for male and female finish times
par(mfrow = c(1, 2))  # Set the plotting layout to have two plots side
by side

# Plot for males
boxplot(FinishTime/60 ~ AgeGroup, data = male_data,
        main = "Finish Time by Age Group - Male",
        xlab = "Age Group", ylab = "Finish Time(minutes)",
        col = "lightblue")

# Plot for females
boxplot(FinishTime/60 ~ AgeGroup, data = female_data,
        main = "Finish Time by Age Group - Female",
        xlab = "Age Group", ylab = "Finish Time (minutes)",
        col = "pink")


#--------------------------------------------------------
#Race performance EXCEL DATA

# Subset the data for males and females
male_data <- data[data$Gender == "M", ]
female_data <- data[data$Gender == "F", ]

# Aggregate the minimum and mean time by year for males
male_min_time <- aggregate(FinishTime ~ EventYear, data = male_data,
FUN = min)
male_mean_time <- aggregate(FinishTime ~ EventYear, data = male_data,
FUN = mean)

# Aggregate the minimum and mean time by year for females
female_min_time <- aggregate(FinishTime ~ EventYear, data =
female_data, FUN = min)
female_mean_time <- aggregate(FinishTime ~ EventYear, data =
female_data, FUN = mean)

# Convert time values from seconds to minutes
male_min_time$FinishTime <- male_min_time$FinishTime / 60
male_mean_time$FinishTime <- male_mean_time$FinishTime / 60
female_min_time$FinishTime <- female_min_time$FinishTime / 60
female_mean_time$FinishTime <- female_mean_time$FinishTime / 60

# Combine the results into a single data frame
results <- data.frame(EventYear = male_min_time$EventYear,
                      MinTime_Male = male_min_time$FinishTime,
                      MeanTime_Male = male_mean_time$FinishTime,
                      MinTime_Female = female_min_time$FinishTime,
                           MeanTime_Female = female_mean_time$FinishTime)

# View the results
results
```

EventYear	MinTime_Male	MeanTime_Male	MinTime_Female	MeanTime_Female
2004	247,48	344,19	268,00	368,51
2005	232,10	346,58	263,28	372,45
2006	225,62	337,75	252,97	362,41
2007	226,47	340,73	248,28	363,76
2008	232,85	349,47	262,87	374,47
2009	216,73	339,59	239,55	365,88
2010	223,73	352,32	247,82	376,01
2011	225,62	346,32	246,72	371,53
2012	220,22	345,39	243,92	370,73
2013	219,98	344,21	243,63	371,80
2014	218,70	346,43	240,23	374,41
2015	221,78	346,79	240,80	374,88
2016	218,88	345,44	246,60	374,40
2017	220,40	346,30	241,67	375,39
2018	217,68	344,35	237,90	373,28
2019	217,25	343,88	235,48	373,95
2020	217,67	335,80	238,62	361,06

```
#-----------------------------------------------------------
#TWO-WAY ANOVA TEST
#-----------------------------------------------------------

# Fit the two-way ANOVA model
model <- aov(FinishTime ~ AgeGroup * Gender, data = data)

# Conduct the ANOVA test
anova_result <- anova(model)

# Print the ANOVA table
print(anova_result)

#-----------------------------------------------------------------------
---------------------------------
#BACKWARD AND FORWARD VARIABLE SELECTION
#-----------------------------------------------------------------------
---------------------------------

library(MASS)

model_backward<-stepAIC(model_full,trace=TRUE,direction="backward")


model_null <- lm(FinishTime ~ 1, data = data)

# Perform forward variable selection using stepAIC
model_forward <- stepAIC(model_null, scope = list(lower = ~1, upper =
~ Gender + AgeGroup + AgeBand + Country + EventYear + EventLocation +
SwimTime + BikeTime + RunTime), direction = "forward")
```

```
#-----------------------------------------------------------------------
#PARAMETRIC BOOTSTRAP
#-----------------------------------------------------------------------
# Load required library
library(ggplot2)

#First we need to identify which distribution the variable FinishTime
follows.

# Plot a histogram of FinishTime
ggplot(sampled_data, aes(x = FinishTime)) +
  geom_histogram(fill = "lightblue", color = "black") +
  labs(x = "Finish Time", y = "Count", title = "Histogram of Finish
Time")

# Plot a density plot of FinishTime
ggplot(sampled_data, aes(x = FinishTime)) +
  geom_density(fill = "lightblue", color = "black") +
  labs(x = "Finish Time", y = "Density", title = "Density Plot of
Finish Time")

#To perform the parametric bootstrap assuming that the "FinishTime"
variable follows a normal distribution, we can follow these steps:

#Estimate the parameters of the normal distribution
mu <- mean(sampled_data$FinishTime)/60
mu
sigma<-sd(sampled_data$FinishTime)/60
sigma

#Generate a random sample of the same size as the original dataset
from the assumed normal distribution.
boot_sample <- rnorm(n = nrow(sampled_data), mean=mu, sd=sigma)

#Perform the desired analysis or calculation using the bootstrap
sample.
boot_mean <- mean(boot_sample)
boot_median <- median(boot_sample)

n_boot <- 1000  # Number of bootstrap iterations
boot_means <- numeric(n_boot)
boot_medians <- numeric(n_boot)
boot_sd <- numeric(n_boot)

for (i in 1:n_boot) {

  #Generate a random sample of the same size as the original dataset
from the assumed normal distribution
  boot_sample = rnorm(1000, mu, sigma)

  boot_means[i] <- mean(boot_sample)
  boot_medians[i] <- median(boot_sample)
  boot_sd[i] <- sd(boot_sample)

}

#Analyze the distribution of the bootstrap statistics.
hist(boot_means, main = "Bootstrap Distribution of Mean", xlab =
"Mean", col="yellow")
```

```
hist(boot_medians, main = "Bootstrap Distribution of Median", xlab =
"Median",col="lightgreen")
hist(boot_sd, main = "Bootstrap Distribution of Standard Deviation",
xlab = "Standard Deviation",col="orange")

boot_mean_ci <- quantile(boot_means, c(0.025, 0.975))
boot_mean_ci
boot_median_ci <- quantile(boot_medians, c(0.025, 0.975))
boot_median_ci
boot_sd_ci <- quantile(boot_sd, c(0.025, 0.975))
boot_sd_ci


#----------------------------------------------------------------------
#NON-PARAMETRIC BOOTSTRAP
#----------------------------------------------------------------------

num_boot <- 1000  # Number of bootstrap iterations
n <- nrow(sampled_data)  # Number of observations in the dataset

# Create empty vectors to store bootstrap statistics
bootstrap_mean <- numeric(num_boot)
bootstrap_median <- numeric(num_boot)
bootstrap_sd <- numeric(num_boot)

# Perform bootstrap iterations
for (i in 1:num_boot) {
  # Generate bootstrap sample by resampling with replacement
  bootstrap_sample <- sampled_data[sample(n, replace = TRUE), ]

  # Calculate statistics from the bootstrap sample
  bootstrap_mean[i] <- mean(bootstrap_sample$FinishTime)
  bootstrap_median[i] <- median(bootstrap_sample$FinishTime)
  bootstrap_sd[i] <- sd(bootstrap_sample$FinishTime)
}

#Analyze the distribution of the bootstrap statistics.
hist(bootstrap_mean, main = "Bootstrap Distribution of Mean", xlab =
"Mean", col="yellow")
hist(bootstrap_median, main = "Bootstrap Distribution of Median", xlab
= "Median",col="lightgreen")
hist(boot_sd, main = "Bootstrap Distribution of Standard Deviation",
xlab = "Standard Deviation",col="orange")

# Calculate confidence intervals
confidence_interval_mean <- quantile(bootstrap_mean, c(0.025, 0.975))
confidence_interval_mean
confidence_interval_median <- quantile(bootstrap_median, c(0.025,
0.975))
confidence_interval_median
confidence_interval_sd <- quantile(bootstrap_sd, c(0.025, 0.975))
confidence_interval_sd

#---------------------------------------
# Obtain the original dataset and extract the "Gender" variable
gender_data <- sampled_data$Gender

# Determine the sample size of the dataset
n <- length(gender_data)

# Set the number of bootstrap iterations
```

```
n_boot <- 1000

# Create an empty matrix to store the bootstrap sample statistics
bootstrap_stats <- matrix(NA, n_boot, 2)  # 2 columns for Male and
Female proportions

# Perform the bootstrap procedure
for (i in 1:n_boot) {
  bootstrap_sample <- sample(gender_data, replace = TRUE)
  bootstrap_prop <- prop.table(table(bootstrap_sample))  # Calculate
the proportion of each gender category
  bootstrap_stats[i, ] <- bootstrap_prop
}

# Analyze the distribution of the bootstrap statistics
hist(bootstrap_stats[, 1], main = "Bootstrap Distribution of Male
Proportion", xlab = "Proportion")
hist(bootstrap_stats[,2], main = "Bootstrap Distribution of Female
Proportion", xlab = "Proportion")



#-----------------------------------------------------------------------
#CORRELATION MATRIX
#-----------------------------------------------------------------------
library(ggplot2)
library(ggcorrplot)
library(factoextra)

# Select the numeric variables of interest from the sampled_data
numeric_data <- sampled_data[, c("SwimTime", "BikeTime", "RunTime",
"FinishTime")]

# Normalize the numeric data by scaling it
data_normalized <- scale(numeric_data)

# Calculate the correlation matrix
corr_matrix <- cor(data_normalized)
corr_matrix

# Visualize the correlation matrix using ggcorrplot
ggcorrplot(corr_matrix)

#The result of the correlation matrix can be interpreted as follow:
#The higher the value, the most positively correlated the two
variables are.
#The closer the value to -1, the most negatively correlated they are.

#-----------------------------------------------------------------------
#PREDICT FUNCTION
#-----------------------------------------------------------------------

# Fit linear regression models for SwimTime, BikeTime, and RunTime
model_1<-lm(SwimTime/60 ~ Gender+AgeBand,data=sampled_data)

model_2<-lm(BikeTime/60 ~ Gender+AgeBand,data=sampled_data)

model_3<-lm(RunTime/60 ~ Gender++AgeBand,data=sampled_data)

# Create a grid of values for Gender and AgeBand
gender <- c("M", "F")
```

```r
age_band <- c(20, 30, 40, 50, 60)

# Create an empty data frame to store the predicted values
predictions <- data.frame(Gender = character(),
                          AgeBand = numeric(),
                          SwimTime = numeric(),
                          BikeTime = numeric(),
                          RunTime = numeric())

# Generate predictions for each combination of Gender and AgeBand
for (g in gender) {
  for (a in age_band) {
    new_data <- data.frame(Gender = g, AgeBand = a)

    # Predict SwimTime, BikeTime, and RunTime using the respective
models
    swim_pred <- predict(model_1, newdata = new_data, type =
"response")
    bike_pred <- predict(model_2, newdata = new_data, type =
"response")
    run_pred <- predict(model_3, newdata = new_data, type =
"response")

    # Append the predictions to the data frame
    predictions <- rbind(predictions, data.frame(Gender = g, AgeBand =
a,
                                                 SwimTime = swim_pred,
                                                 BikeTime = bike_pred,
                                                 RunTime = run_pred))
  }
}

# Plot the predictions
library(ggplot2)

# SwimTime
swim_plot <- ggplot(predictions, aes(x = AgeBand, y = SwimTime, color
= Gender)) +
  geom_line() +
  labs(title = "Predicted SwimTime by Gender and AgeBand",
       x = "AgeBand", y = "SwimTime (minutes)")


# BikeTime
bike_plot <- ggplot(predictions, aes(x = AgeBand, y = BikeTime, color
= Gender)) +
  geom_line() +
  labs(title = "Predicted BikeTime by Gender and AgeBand",
       x = "AgeBand", y = "BikeTime (minutes)")

# RunTime
run_plot <- ggplot(predictions, aes(x = AgeBand, y = RunTime, color =
Gender))+  geom_line() +
  labs(title = "Predicted RunTime by Gender and AgeBand",
       x = "AgeBand", y = "RunTime (minutes)")

# Display the plots side by side using grid.arrange
library(gridExtra)
grid.arrange(swim_plot, bike_plot,run_plot, nrow = 1)

#----------------------------------------------------
```

```
#Example prediction

newdata = data.frame(Gender="F", AgeBand = 20)

model1<-lm(SwimTime/60 ~ Gender+AgeBand,data=sampled_data)
predict(model1, newdata = newdata, type = "response")

model2<-lm(BikeTime/60 ~ Gender+AgeBand,data=sampled_data)
predict(model2, newdata = newdata, type = "response")

model3<-lm(RunTime/60 ~ Gender+AgeBand,data=sampled_data)
predict(model3, newdata = newdata, type = "response")

model4<-lm(FinishTime/60 ~ Gender+AgeBand,data=sampled_data)
predict(model4, newdata = newdata, type = "response")


#---------------------------------------------------------------------
#PEARSON RESIDUALS
#---------------------------------------------------------------------

# Calculate Pearson residuals using the full model
res=residuals(model_full,"pearson")

plot(res, pch = 20, xlab = "Observation", ylab = "Pearson Residuals",
     main = "Pearson Residuals of Ironman Dataset")

# Find the maximum Pearson residual
max(res)

# Find the index of the outlier with the highest Pearson residual
outlier<-which.max(res)
outlier

# Print the details of the outlier
sampled_data$FinishTime[outlier]/60 #lowest time
sampled_data$Gender[outlier]
sampled_data$AgeBand[outlier]
sampled_data$Country[outlier]
sampled_data$EventYear[outlier]
sampled_data$EventLocation[outlier]

newdata = data.frame(Gender="M", AgeBand = 30)

model1<-lm(FinishTime/60 ~ Gender+AgeBand,data=sampled_data)
predict(model1, newdata = newdata, type = "response")
```