MGSC 401: Statistical Foundations of Data Analytics

Final Project

Tokyo Olympics and the recruitment of future Champions

McGill University
Dec 12, 2022

1 Introduction

We are in 2016 and James is a recruiter from the legendary Spliney International Corporation, also known as SIC. His role as a recruiter is to use data analytics to identify young athletes with high potential and make them an offer to join the very selective program of SIC. In this case, high potential means a great likelihood of winning a medal in the Tokyo Olympics of 2020.

To define the next generation of champions, the Spliney International Corporation focuses on recruiting talent from 16 to 19 years old who participated in the last 2016 Summer games. This year, James came up with the idea of using statistical models to be the most accurate in finding future champions.

His analytics work will be structured in 3 distinct parts. First, he will go through the data to understand the variables and their relationship, while also cleaning the unnecessary data collected from more than 270,000 past observations. Second, he will use classification models such as logistic regression or classification trees to come up with the perfect prediction model. Finally, James will use the model to predict top athletes and draw conclusions from the results.

2 Data Description and Cleaning

The model will try to predict if an athlete is a future champion or not a future champion. Thus, James defines a dependent variable Champion in the training dataset, which takes the value of 1 if someone receives a bronze, silver, or gold medal, and 0 otherwise.

The first step in the recruiter's variable selection was the removal of identifier variables ID, and city. These variables do not have a role in predicting a champion and including these would lead to overfitting the data. Next, duplicate variables were removed such as Team, Games, or Event. Those are very much similar to NOC, Year, and Sport, and would lead to multicollinearity in our model. Lastly, James proceeded to remove observations from all the athletes that had height and weight as unknown due to the importance of these ones in future prediction models. In the following analysis, the recruiter will go over each variable to detect skewness, and outliers that could influence the model.

Numerical Variables

2.1 Age

This predictor tells the age of the athlete at the time of the Olympics. The average age among athletes is 25 years old and the data seem to follow a normal distribution around this mean. The recruiter decided to include this variable in the model as the age factor can be interpreted as the level of experience. Intuitively, the older you are, the more you have experience and chances you have of winning.

Additionally, James removed all the players that were between 16 years old and 19 years old in the games of 2016 in the training dataset as they will be used in the test dataset.

2.2 Weight

Weight is also an important factor when it comes to determining the winner. Like the previous variable Age, the data collected seem to be normally distributed around 70.69 kilograms and no transformation was needed. (Appendix 1A)

2.3 Height

Together with Weight, this variable gives information about the physical form of the athlete. It is naturally normally distributed around an average of 170cm and there is no need in transforming this variable using a log or other functions (Appendix 1B).

2.4 Year

This variable describes the year in which the Olympic Games took place. Due to the changing nature of sports regulations, the recruiter decided to train his model on the Olympic Games that occurred starting in 1996, leaving room for 20 years of data. Trimming the data this way is beneficial as it only focuses on relevant observations and corrects the data that was originally skewed because there are more athletes competing in the last Olympics than in the 19th century (Appendix 2).

Categorical Variables

2.5 Sex

Sex is the variable for the genre of the athlete, either man or woman. This is essential to include in our model as the difference in physical attributes generally differs between women athletes and men athletes.

2.6 NOC

NOC represents the abbreviation of the country's name participating in the Olympics. 210 countries were first identified in the dataset. It is believed that including all of them might lead to overfitting of the model as most of them have few medals won. Therefore, to increase his chances, James decided to focus on recruiting talents in countries that have won more than 100 medals in the last 20 years – which averages to a minimum of 17 medals won per Olympics (Appendix 3).

2.7 Sport

The sport variable gather all the sports that are competed during the Olympics. These sports are rather unique from one to another and some of them are already grouped as one such as athletics. This is the reason why the recruiter decided to keep all of them in the model.

2.8 Multicollinearity

To find whether variables were dependent between them, James created a correlation matrix and performed a Variance Inflation Factor (VIF) test for variables. No significant correlation between most variables was identified, however quite logically, height and weight showed an important sign of correlation. As height has a VIF score higher than 5 and a correlation of more than 0.8 in the matrix, James decided to exclude this variable from the logistic regression model.

	Age	Height	Weight	Year
Age	1.00	0.16	0.20	0.07
Height	0.16	1.00	0.81	0.02
Weight	0.20	0.81	1.00	0.01
Year	0.07	0.02	0.01	1.00

Table 1: Correlation Matrix between Numerical Variables

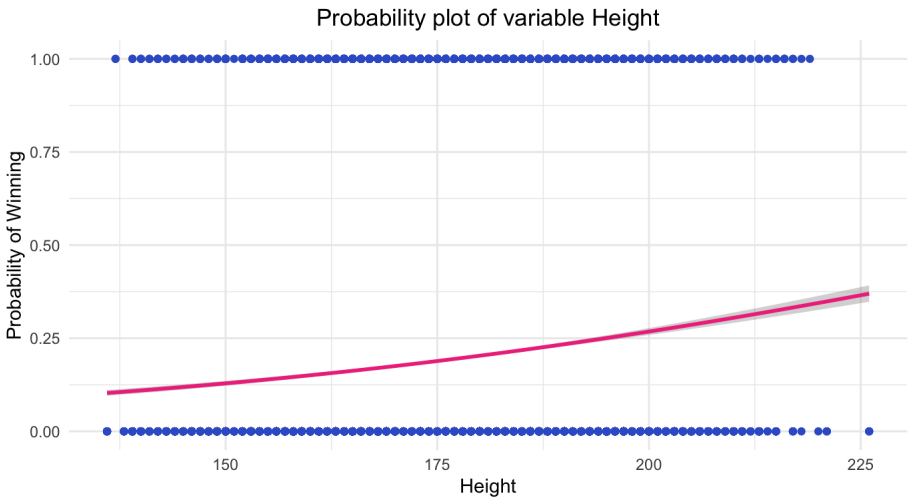
3 Data Analysis and Predictive Model Building

3.1 Multiple logistic regression

Moving forward, James decided to build a multiple logistic regression model and use it to calculate the probability that each player will win a medal. This model is based on an algorithm called the Maximum Likelihood Estimation (MLE) that maximizes the coefficients b0 and b1 in the following formula, with X being the dependent variable and Y being the independent variable:

$$Prob(Y = 1) = \frac{e^{b_0+b_1x_1+\ldots+b_px_p}}{1+e^{b_0+b_1x_1+\ldots+b_px_p}}$$

Using a summary of this model, an interesting finding was that most of the predictors were statistically significant, meaning that they played an important role in predicting if someone will win a medal. James also realized that a player had the most chances of winning a medal if he was a man, was older, was heavier and taller, was either Jamaican or from the United States, played Baseball, Softball, or Football, and participated in the most recent Olympics. (Appendix 4). A logistic plot of probabilities is a good manner to visually represent the change in probabilities of an outcome depending on a single factor. As an example, the following probability plot shows that as height increases, the probability of winning a medal also increases.



To understand the extent to which the variables explain the variation in winning or not winning, an R-squared analysis has been performed. James found that overall, the model explained 13.8% of the variation in the outcome variable Champion. Before predicting the results, the recruiter wanted to compare this logistic regression model with another powerful tool that could help him find the most promising athletes: Classification trees.

3.2 Classification trees using Random Forest

In contrast with logistic regression, the random forest method uses classification trees as its base to come up with the best prediction model. This model is an improvement of simple classification trees that suffer from high variance and the bagging method which suffers from biases due to possible multicollinearity. Instead of using all predictors in a tree, the random forest only chooses a random subset of predictors equal to the sqrt(total predictors) for its classification trees. This small improvement makes random forests one of the most powerful machine-learning techniques. In this case, since p=7, each tree will be built using either 2 or 3 predictors.

First, James needs to run a single classification tree and find the optimal cp that minimizes the residual sum of squared errors of this tree. Once the optimal complexity level of each tree is found, he will be able to use the random forest method with 500 optimal classification trees. In this scenario, the optimal complexity level has been found to be 0.001 (Appendix 5).

Using the random forest method, James was able to find an optimized prediction model with a minimized error rate of 16.66%. He was also able to analyze the relative predictive importance of different predictors. As shown in the graph below, the most important predictor in determining if an athlete will be a champion is his/her nationality combined with the sport he/she is competing for. Surprisingly, age, height, and weight do not play the most important role in winning a medal.

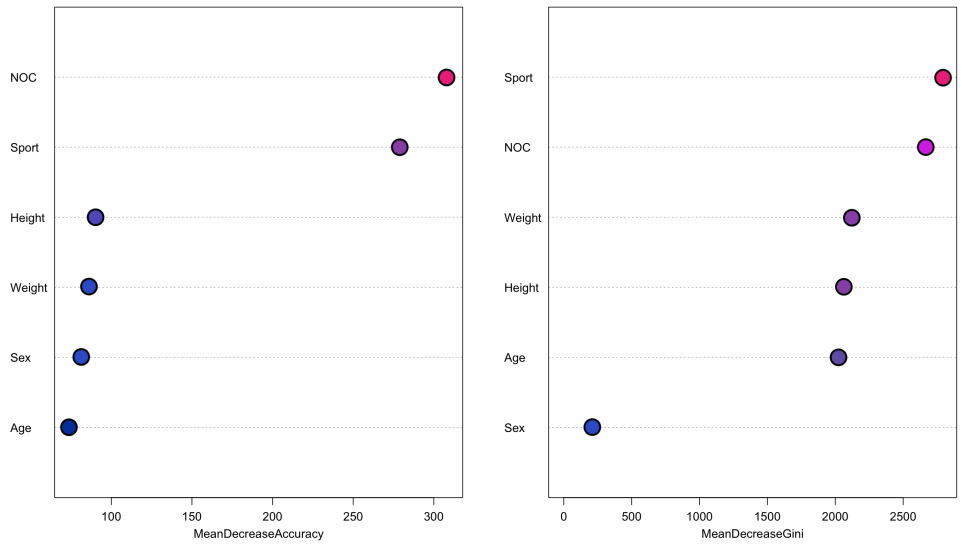


Figure 3.2: Importance Of Predictors in Random Forest

4 Prediction Results

The Spliney International Corporation focuses on recruiting young talents that are in the top 27 winning countries in the last 5 Olympics. As the recruiter James wants to analyze if they would win a medal in the next games occurring 4 years from 2016, he used his statistical models on players having 16 to 19 years old in 2016 and added 4 years to their age, while assuming that their height, weight, nationality, and the sport they play would stay constant.

To measure which of the logistic regression and random forest models would be used to recruit athletes, the recruiter first predicted the results using the test data and compared it with the actual medal winners aged 16 to 19 years old in 2016. While the logistic regression model has a 16.06% error rate in predicting the winners, the random forest has an error rate of 12.95%. As such the recruiter decided to use the 500-tree method of random forest to determine the players to pick. However, one can note that the similarity rate of the two models predictions was 93.09%. This is extremely high and proves that the two models give rather similar predictions overall.

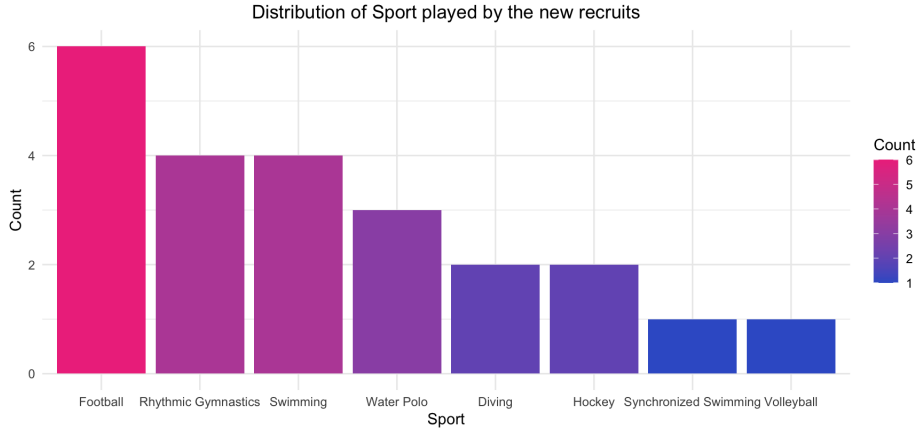
Using the random forest, James identified 23 athletes that are predicted to win a medal in Tokyo Olympics of 2020 and are subject to being recruited by the Spliney Corporation. Out of the 332 athletes on the watch, this recruitment method leads to an acceptance rate of 6.93%. The following table lists the selected athletes and their attributes.

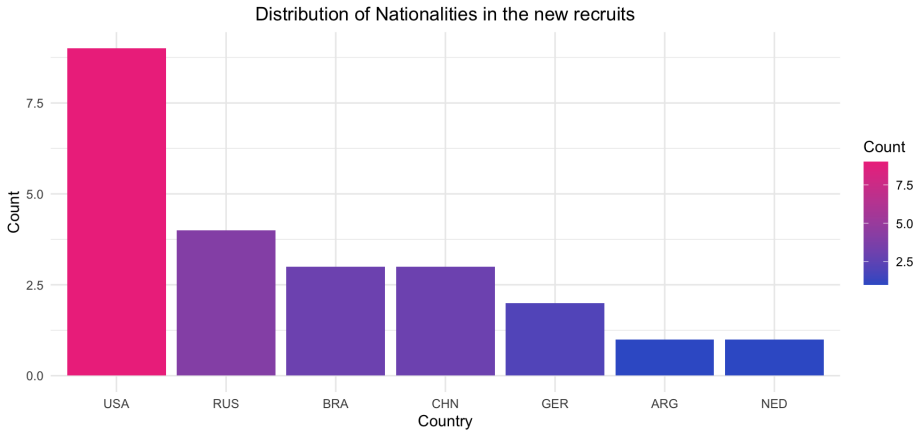
Name	Sex	Age	Height	Weight	NOC	Year	Sport
Anita Alvarez	F	19	170	52	USA	2016	Synchronized Swimming
Santiago Ascacibar	M	19	168	69	ARG	2016	Football
Kathleen Baker	F	19	173	68	USA	2016	Swimming
Vera Leonidovna Biryukova	F	18	168	47	RUS	2016	Rhythmic Gymnastics
Max Christiansen	M	19	187	84	GER	2016	Football
Jorrit Croon	M	17	183	75	NED	2016	Hockey
Caeleb Remel Dressel	M	19	191	86	USA	2016	Swimming
Aria Fischer	F	17	183	78	USA	2016	Water Polo
Makenzie Fischer	F	19	186	74	USA	2016	Water Polo
Gabriel Barbosa Almeida	M	19	178	68	BRA	2016	Football
Gabriel Fernando de Jesus	M	19	175	68	BRA	2016	Football
Francis Townley Haas	M	19	196	84	USA	2016	Swimming
Timm Herzbruch	M	19	180	76	GER	2016	Hockey
Yana Alekseyevna Kudryavtseva	F	18	170	47	RUS	2016	Rhythmic Gymnastics
Liu Huixia	F	18	157	48	CHN	2016	Diving
Madeline "Maddie" Musselman	F	18	181	65	USA	2016	Water Polo
Mallory Diane Pugh	F	18	163	55	USA	2016	Football
Si Yajie	F	17	164	57	CHN	2016	Diving
Anastasiya Alekseyevna Tatareva	F	19	165	44	RUS	2016	Rhythmic Gymnastics
Thiago Maia Alencar	M	19	178	64	BRA	2016	Football
Mariya Yuryevna Tolkachova	F	18	176	53	RUS	2016	Rhythmic Gymnastics
Abbey Weitzeil	F	19	178	68	USA	2016	Swimming
Yuan Xinyue	F	19	201	78	CHN	2016	Volleyball

Table: Predicted Winners

5 Conclusion

At the end, recruits were mainly coming from the USA, Russia, Brazil, and China. This is not surprising as those countries are known to perform well during the Olympics and have a great number of high-potential athletes. Furthermore, the weight factor seems to have an important variance and no recognizable pattern while height is approximately normal around 178 centimeters (Appendix 6). As shown earlier by the importance plot for variables, height and weight are not the most important predictors that determine if an athlete will win or not. Here are two graphs that show the distribution of Countries and Sports among new recruits.

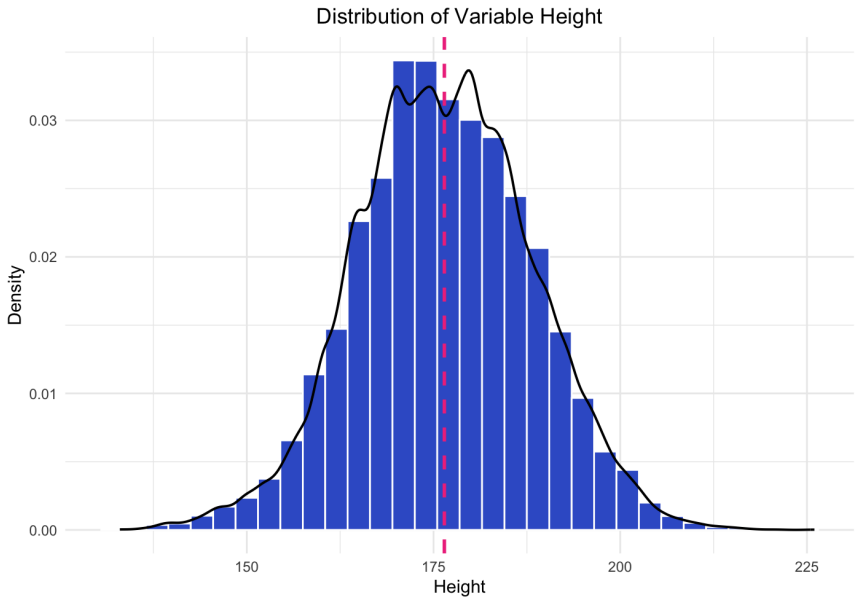


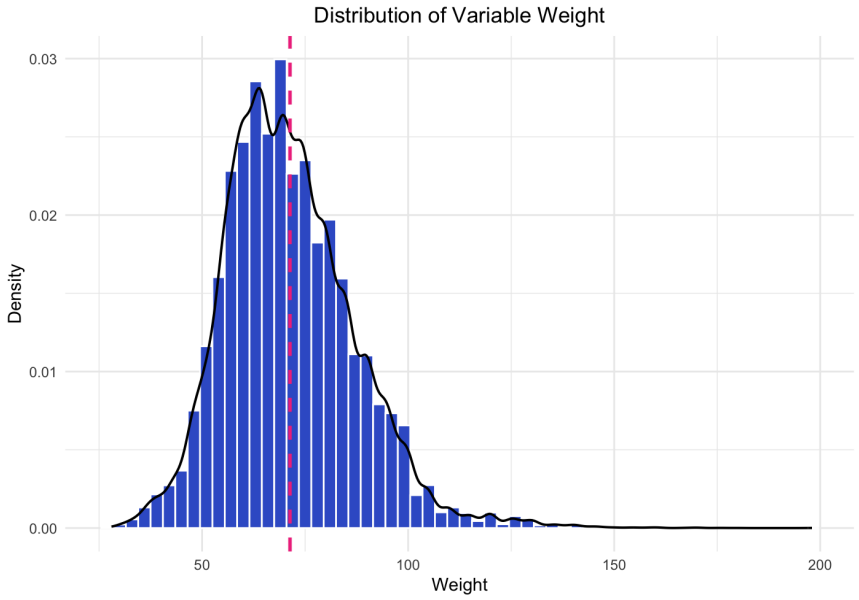


James is happy as its statistical models gave meaningful results and used computer science to sharpen his recruitment approach. Nonetheless, there is no certitude that these players will win a medal in the next Olympics as sport is a lot more complex than just physical attributes and past performance data. As an example, the mental performance of a player also plays a big role in any sport, independent of the sex, weight, height, age, or nationality of the athlete.

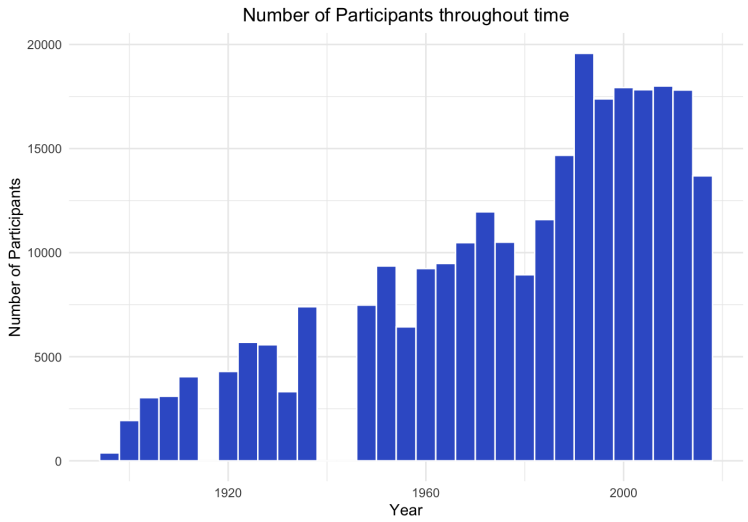
6 Appendix

Appendix 1: Distribution of variable Weight and Height

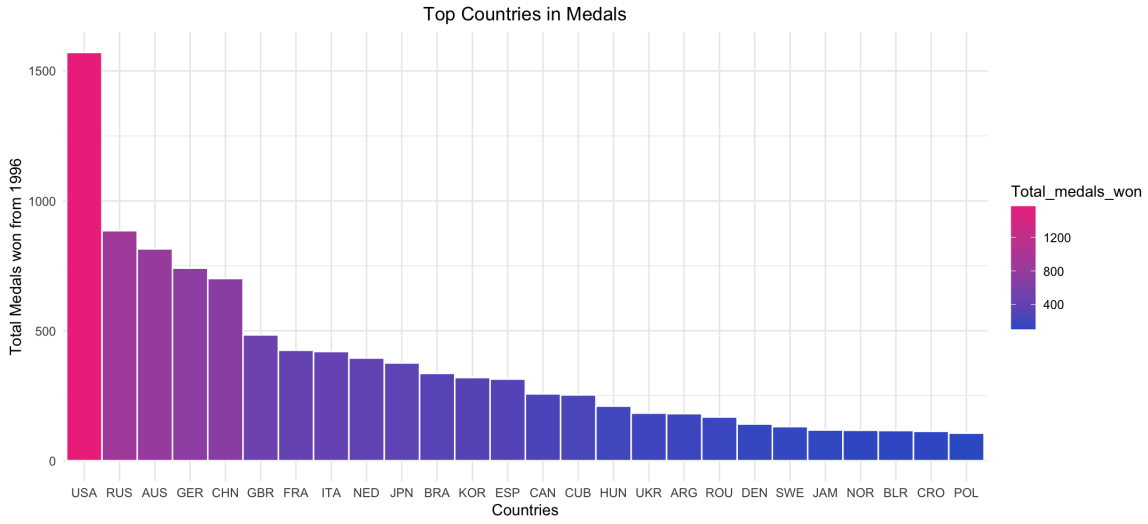




Appendix 2: Skewness in the Number of participants through time



Appendix 3: Countries with more than 100 medals won from 1996

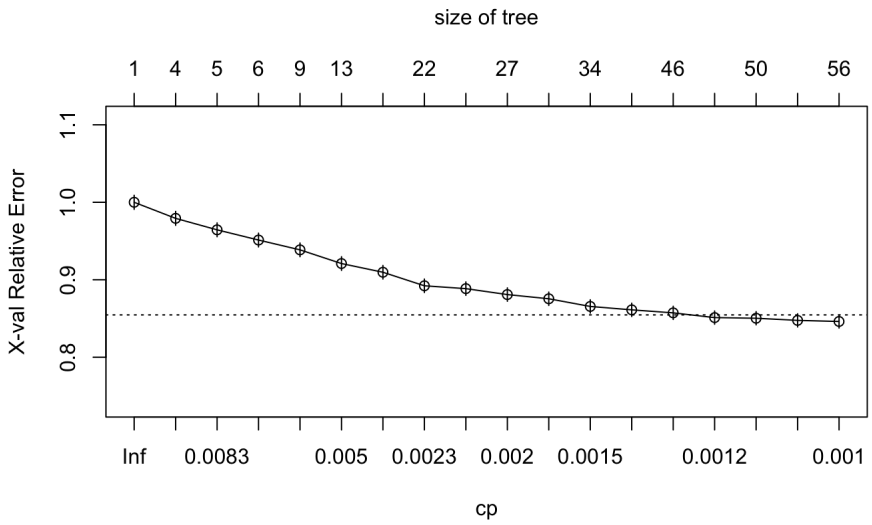


Appendix 4: Logistic Regression Model

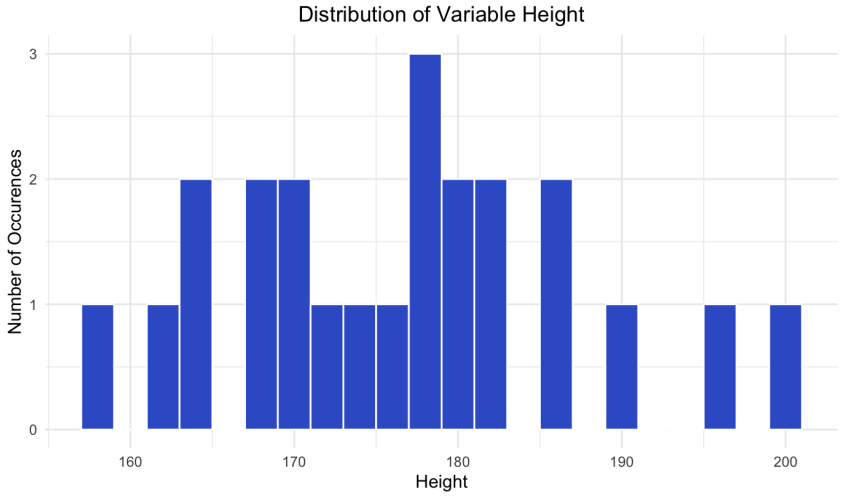
Final Logistic Regression Model

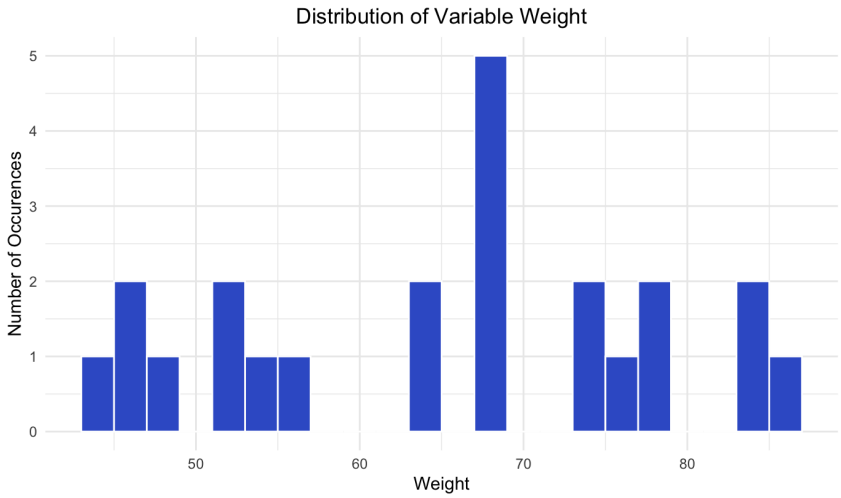
	Dependent variable:
	Champion
Sex	-0.289***
	(0.032)
Age	0.010***
	(0.003)
Weight	0.009***
	(0.001)
Australia	0.467***
	(0.095)
Belarus	-0.234*
	(0.132)
Brazil	0.054
	(0.105)
Canada	-0.507***
	(0.110)
China	0.539***
	(0.097)
Croatia	0.094
	(0.137)
Cuba	0.584***
	(0.116)
Denmark	0.278**
	(0.130)
Spain	-0.070
	(0.105)
France	0.166
	(0.101)
United Kingdom	0.267***
	(0.100)
Germany	0.491***
	(0.096)
Hungary	-0.014
	(0.116)
Italia	0.084
	(0.102)
Jamaica	1.734***
	(0.145)
Japan	0.041
	(0.104)
Korea	0.095
	(0.106)
Netherlands	0.451***
	(0.105)
Norway	0.582***
	(0.140)
Poland	-0.802***
	(0.133)
Roumania	0.281**
	(0.123)
Russia	0.907***
	(0.096)
Sweden	-0.298**
	(0.129)
Ukraine	-0.235**
	(0.118)
USA	1.224***
	(0.092)
Year	0.005**
	(0.002)
Athletics	-0.757***
	(0.107)
Badminton	-0.216
	(0.146)
Baseball	1.315***
	(0.131)
Basketball	0.555***
	(0.119)
Beach Volleyball	-0.305*
	(0.171)
Boxing	0.582***
	(0.135)
Canoeing	0.391***
	(0.115)
Cycling	-0.110
	(0.114)
Diving	0.323**
	(0.133)
Equestrianism	-0.126
	(0.126)
Fencing	0.299**
	(0.116)
Football	0.980***
	(0.115)
Golf	-1.248**
	(0.532)
Gymnastics	-0.706***
	(0.116)
Handball	0.765***
	(0.115)
Hockey	0.965***
	(0.114)
Judo	0.258**
	(0.123)
Modern Pentathlon	-0.652***
	(0.239)
Rhythmic Gymnastics	0.772***
	(0.162)
Rowing	0.612***
	(0.109)
Rugby Sevens	-0.065
	(0.216)
Sailing	-0.024
	(0.122)
Shooting	-0.665***
	(0.124)
Softball	1.400***
	(0.150)
Swimming	0.029
	(0.106)
Synchronized Swimming	1.038***
	(0.138)
Table Tennis	-0.279*
	(0.143)
Taekwondo	0.930***
	(0.170)
Tennis	-0.760***
	(0.148)
Trampolining	0.465*
	(0.242)
Triathlon	-1.315***
	(0.264)
Volleyball	0.570***
	(0.118)
Water Polo	0.601***
	(0.121)
Weightlifting	0.246*
	(0.144)
Wrestling	0.356***
	(0.126)
Intercept	-11.992***
	(3.628)
Observations	50,146
Log Likelihood	-22,486.940
Akaike Inf. Crit.	45,103.870
Note:	*p<0.1; **p<0.05; ***p<0.01

Appendix 5: Optimal Complexity level for each Tree



Appendix 6: Weight and Height in new recruited Athletes





```
# install packages
#install.packages("ggplot2")
#install.packages("stargazer")
#install.packages("rms")
#install.packages(c('tibble', 'dplyr'))
#install.packages("stargazer")
#install.packages("rms")
#install.packages("randomForest")
#install.packages("tree")
#install.packages("rpart.plot")
#install.packages("ggthemes")

library(tree)
library(rpart)
library(rpart.plot)
library(ggplot2)
library(stargazer)
require(methods)
require(psych)
library("car")
require(caTools)
library(MASS)
require(rms)
library(dplyr)
library(randomForest)
library(ggthemes)

#II) Data description
#import data
Olympics = read.csv("/Users/liu/Documents/FALL 2022/MGSC 401 -Stat founds of data
analytics/final project/Olympic events data.csv")
attach(Olympics)

#remove columns ID, city, Team, Games, Season, and Event
Olympics = Olympics[,-c(1,7,9,12,14)]
attach(Olympics)

#remove NA observations that are in the variables: Age, height, weight, NOC, Year, Sport

Olympics1 <- Olympics[!is.na(Olympics$Height), ]
Olympics1 <- Olympics1[!is.na(Olympics1$Weight), ]
Olympics1<- Olympics1[!is.na(Olympics1$Age), ]
```

```
Olympics1 <- Olympics1[!is.na(Olympics1$NOC), ]
Olympics1 <- Olympics1[!is.na(Olympics1$Year), ]
Olympics1 <- Olympics1[!is.na(Olympics1$Sport), ]

#create column name to see if an athlete is a Champion (won Gold, silver, or bronze) and remove medal column
library(tibble)
library(dplyr)
Olympics1 <- Olympics1 %>%
  add_column(Champion = if_else(is.na(Olympics1$Medal) , "0", "1"))
Olympics1 <- Olympics1[,-c(10)]

#remove all observations before the games of 1996 (leaving room for 20 years of data) and winter season
Olympics1 <- subset(Olympics1, Year >=1996)
Olympics1 <- subset(Olympics1, Season =="Summer")
Olympics1 <- Olympics1[,-c(8)]

#use groupby function to analyze the variables in terms of the number of Medals won
NOC_groupby <- Olympics1 %>%
  group_by(NOC) %>%
  dplyr::summarise(Total_medals_won = sum(as.numeric(Champion)))

## Numerical variables (description using GGplot)
#Age
ggplot(Olympics1, aes(x=Age))+geom_histogram(aes(y=..density..), position="identity", binwidth = 3,color="white",fill="royalblue3") +xlab("Age")+ylab("Density")+ggtitle("Distribution of Variable Age")+
  theme_minimal()+theme(plot.title = element_text(hjust=0.5))+geom_density(size=0.7)+geom_vline(aes(xintercept=mean(Age)),color="violetred2",linetype="dashed", size=1)

ggplot(Olympics1, aes(y=Age))+geom_boxplot(color="black",fill="royalblue3",outlier.colour = "violetred2")+ylab("Age")+ggtitle("Boxplot of Age")+theme_minimal()+theme(plot.title = element_text(hjust=0.5))

#Height
ggplot(Olympics1, aes(x=Height))+geom_histogram(aes(y=..density..), position="identity", binwidth = 3,color="white",fill="royalblue3") +geom_density(size=0.7)+
  xlab("Height")+ylab("Density")+theme_minimal()+ ggtitle("Distribution of Variable Height")+theme(plot.title =
```

```
element_text(hjust=0.5))+geom_vline(aes(xintercept=mean(Height)),color="violetred2",linetype
="dashed", size=1)
ggplot(Olympics1, aes(y=Height))+geom_boxplot(color="black",fill="royalblue3",outlier.colour =
"violetred2")+ylab("Height")+ggtitle("Boxplot of Height")+theme_minimal()+theme(plot.title =
element_text(hjust=0.5))

#Weight
ggplot(Olympics1, aes(x=Weight))+geom_histogram(aes(y=..density..), position="identity",
binwidth = 3,color="white",fill="royalblue3") +geom_density(size=0.7)+
  xlab("Weight")+ylab("Density")+theme_minimal()+ ggtitle("Distribution of Variable
Weight")+theme(plot.title =
element_text(hjust=0.5))+geom_vline(aes(xintercept=mean(Weight)),color="violetred2",linetyp
e="dashed", size=1)
ggplot(Olympics1, aes(y=Weight))+geom_boxplot(color="black",fill="royalblue3",outlier.colour =
"violetred2")+ylab("Weight")+ggtitle("Boxplot of Weight")+theme_minimal()+theme(plot.title =
element_text(hjust=0.5))
mean(Olympics1$Weight)

#Year
ggplot(Olympics, aes(x=Year))+geom_histogram(binwidth = 4,color="white",fill="royalblue3")
+xlab("Year")+ylab("Number of Participants")+ggtitle("Number of Participants throughout
time")+theme_minimal()+theme(plot.title = element_text(hjust=0.5))
ggplot(Olympics1, aes(y=Year))+geom_boxplot(color="black",fill="royalblue3",outlier.colour =
"violetred2")+ylab("Year")+ggtitle("Boxplot of Year")+theme_minimal()+theme(plot.title =
element_text(hjust=0.5))

## Categorical variables (description using GGplot)
#NOC
ggplot(Olympics1, aes(x=NOC))+geom_bar(color="black",fill="lightskyblue2")+
  xlab("countries")+ylab("Number of Occurences")+ggtitle("Distribution of the
countries")+theme_minimal()+
  theme(plot.title = element_text(hjust=0.5))

ggplot(NOC_groupby, aes(NOC, Total_medals_won))+geom_col()
NTOP_NOC <- subset(NOC_groupby, Total_medals_won>100)

Olympics1=Olympics1[grepl("USA|UKR|SWE|RUS|ROU|POL|NOR|NED|KOR|JPN|JAM|ITA|HUN|
GER|GBR|FRA|ESP|DEN|CUB|CRO|CHN|CAN|BRA|BLR|AUS|ARG",Olympics1$NOC),]
attach(Olympics1)

#Graph to show the top countries (NTOP_NOC)
```

```
ggplot(NTOP_NOC,aes(x=reorder(NOC,-Total_medals_won),y=Total_medals_won,
fill=Total_medals_won))+geom_bar(width=1,color="white",stat='identity')+
  xlab("Countries")+ylab("Total Medals won from
1996")+scale_fill_gradient(low="royalblue3",high="violetred2")+ggtitle("Top Countries in
Medals")+theme_minimal()+theme(plot.title = element_text(hjust=0.5))

#Sex
ggplot(Olympics1, aes(x=Sex),fill=Sex)+geom_bar(color="white")+
  xlab("Sex")+ylab("Number of Occurences")+ggtitle("Distribution of Athletes'
Sex")+theme_minimal()+scale_fill_manual(values=c("violetred2","royalblue3"))+
  theme(plot.title = element_text(hjust=0.5))

#Sport
ggplot(Olympics1, aes(x=Sport))+geom_bar(binwidth = 1,color="black",fill="grey")+
  xlab("Sport")+ylab("Number of Occurences")+
  ggtitle("Distribution of Sport")+theme_minimal()+theme(plot.title = element_text(hjust=0.5))

#Create Dummies for our categorical variables
Olympics1$Champion=as.factor(Olympics1$Champion)
Olympics1$Sex=as.factor(Olympics1$Sex)
Olympics1$NOC=as.factor(Olympics1$NOC)
Olympics1$Sport=as.factor(Olympics1$Sport)
attach(Olympics1)

#remove test data from training data
Olympics2 =
Olympics1[!((Olympics1$Age==16|Olympics1$Age==17|Olympics1$Age==18|Olympics1$Age==
19)&Olympics1$Year==2016),]

#Correlation matrix for numerical variables on stargazer
numerical_games =Olympics1[,unlist(lapply(Olympics1,is.numeric))]
corr_matrix=cor(numerical_games)
round(corr_matrix,2)
stargazer(corr_matrix,title=c('Correlation Matrix'),type="html")

attach(Olympics2)
vif_test=glm(Champion~Sex+Age+Height+Weight+NOC+Year+Sport, family="binomial")
vif(vif_test)


### III)  Model Building
## 3.1) Logistic regression
```

```
attach(Olympics2)
mlogit=glm(Champion~Sex+Age+Weight+NOC+Year+Sport, family="binomial")
summary(mlogit)

stargazer(mlogit,title=c('Final Logistic Regression
Model'),covariate.labels=c("Sex","Age","Weight","Australia","Belarus","Brazil","Canada","China","Cr
oatia",
                                                    "Cuba", "Denmark", "Spain", "France", "United Kingdom",
"Germany", "Hungary",
                                                    "Italia", "Jamaica", "Japan", "Korea", "Netherlands",
"Norway", "Poland", "Roumania",
                                                    "Russia", "Sweden", "Ukraine", "USA","Year","Athletics",
"Badminton", "Baseball", "Basketball", "Beach Volleyball", "Boxing", "Canoeing", "Cycling",
"Diving",
                                                    "Equestrianism", "Fencing", "Football", "Golf",
"Gymnastics", "Handball", "Hockey", "Judo", "Modern Pentathlon", "Rhythmic Gymnastics",
                                                    "Rowing", "Rugby Sevens", "Sailing", "Shooting",
"Softball", "Swimming", "Synchronized Swimming", "Table Tennis", "Taekwondo",
"Tennis","Trampolining",
                                                    "Triathlon", "Volleyball", "Water Polo", "Weightlifting",
"Wrestling",
                                                "Intercept"),no.space=T,type="html")


#plotting logistic function

plot1=ggplot(Olympics2, aes(y=as.numeric(Champion)-1, x=Height))+ylab("Probability of
Winning")+xlab("Height")+
  ggtitle("Probability plot of variable Height")+theme_minimal()+theme(plot.title =
element_text(hjust=0.5))
scatter=geom_point(color="royalblue3")
line=geom_smooth(method="glm", formula=y~x,
method.args=list(family=binomial),colour="violetred2")
plot1+scatter+line

plot2=ggplot(Olympics2, aes(y=as.numeric(Champion)-1, x=Weight))+ylab("Probability of
Winning")+xlab("Height")+
  ggtitle("Probability plot of variable Weight")+theme_minimal()+theme(plot.title =
element_text(hjust=0.5))
scatter=geom_point(color="royalblue3")
line=geom_smooth(method="glm", formula=y~x,
method.args=list(family=binomial),colour="violetred2")
```

```
plot2+scatter+line

plot3=ggplot(Olympics2, aes(y=as.numeric(Champion)-1, x=Age))+ylab("Probability of
Winning")+xlab("Height")+
  ggtitle("Probability plot of variable Age")+theme_minimal()+theme(plot.title =
element_text(hjust=0.5))
scatter=geom_point(color="royalblue3")
line=geom_smooth(method="glm", formula=y~x,
method.args=list(family=binomial),colour="violetred2")
plot3+scatter+line


# R-squared in logistic models
require(rms)
mlogit2 =lrm(Champion~Sex+Age+Weight+NOC+Sport,data = Olympics2)
mlogit2


## 3.2) Random forest (with best cp and sqrt(7) parameters)
# find optimal cp
attach(Olympics2)
mytree=rpart(Champion~Sex+Age+Height+Weight+NOC+Sport,control=rpart.control(cp=0.001)
)
rpart.plot(mytree)
summary(mytree)

printcp(mytree)
plotcp(mytree)

opt_cp=mytree$cptable[which.min(mytree$cptable[,"xerror"]),"CP"]
opt_cp

#Use random forest method
myforest=randomForest(Champion~Sex+Age+Height+Weight+NOC+Sport, data=Olympics2,
cp=0.001, importance=TRUE,  na.action = na.omit)
myforest

importance(myforest)
varImpPlot(myforest)

### IV) Predictions and results
#test data on unique players
```

```
TestData=Olympics1[Olympics1$Age==16|Olympics1$Age==17|Olympics1$Age==18|Olympics
1$Age==19,]
TestData=TestData[TestData$Year==2016,]

#predicting using logistic function
TestData$Logistic_Probabilities<-predict(mlogit, TestData, type="response")

TestData$Logistic_predictions = ""
for (i in 1:length(TestData$Name)){
  TestData$Logistic_predictions[i]=ifelse(TestData$Logistic_Probabilities[i]>0.4,1,0)
}

#predicting using Random forest
TestData$RandomForest_Prediction <- predict(myforest, TestData, type="response")

# error rate compared to the players that actually won in 2016
#logistic
attach(TestData)
k=0
for (j in 1:579){
  k=ifelse(TestData$Logistic_predictions[j]==TestData$Champion[j],k+1,k+0)
}
Logistic_error_rate=1-k/579
Logistic_error_rate # = 0.1606

#random forest
r=0
for (j in 1:579){
  r=ifelse(TestData$RandomForest_Prediction[j]==TestData$Champion[j],r+1,r+0)
}
RandomForest_error_rate=1-r/579
RandomForest_error_rate # = 0.1295

# similarity rate between the two models
d=0
for (j in 1:579){
  d=ifelse(TestData$Logistic_predictions[j]==TestData$RandomForest_Prediction[j],d+1,d+0)
}
Models_similarity_rate=d/579
Models_similarity_rate # = 0.9309

# List of players using random Forest
```

```
New_Recruits <- subset(TestData, RandomForest_Prediction==1)
New_Recruits <- New_Recruits[,-c(9,10,11,12)]
New_Recruits <- New_Recruits[!duplicated(New_Recruits), ] #23 recruits

# total number of athletes aged 16 to 19 in 2016
potential_athletes = TestData[,-c(9,10,11,12)]
potential_athletes <- potential_athletes[!duplicated(potential_athletes), ]

# Acceptance rate of 0.072289
Acceptance_rate = nrow(New_Recruits)/nrow(potential_athletes)

# Graphs to analyze what are the characteristics shared by the new recruits
#women and men recruits
NR_Sex <- New_Recruits %>%
  group_by(Sex) %>%
  dplyr::summarise(Count = length(as.numeric(Sex)))

ggplot(NR_Sex,aes(x=reorder(Sex,-Count),y=Count, fill=Sex))+geom_bar(stat='identity')+
  xlab("Sex")+ylab("Count")+scale_fill_manual(values =
c('violetred2',"royalblue3"))+ggtitle("Distribution of Men and Women in the new
recruits")+theme_minimal()+theme(plot.title = element_text(hjust=0.5))

#Country recruits
NR_Country <- New_Recruits %>%
  group_by(NOC) %>%
  dplyr::summarise(Count = length(as.numeric(NOC)))

ggplot(NR_Country,aes(x=reorder(NOC,-Count),y=Count,
fill=Count))+geom_bar(stat='identity')+
  xlab("Country")+ylab("Count")+ggtitle("Distribution of Nationalities in the new
recruits")+scale_fill_gradient(low="royalblue3",high="violetred2")+theme_minimal()+theme(pl
ot.title = element_text(hjust=0.5))

#Sports recruits
NR_Sports <- New_Recruits %>%
  group_by(Sport) %>%
  dplyr::summarise(Count = length(as.numeric(Sport)))

ggplot(NR_Sports,aes(x=reorder(Sport,-Count),y=Count, fill=Count))+geom_bar(stat='identity')+
  xlab("Sport")+ylab("Count")+ggtitle("Distribution of Sport played by the new
recruits")+scale_fill_gradient(low="royalblue3",high="violetred2")+theme_minimal()+theme(pl
ot.title = element_text(hjust=0.5))
```

```
#Height distribution
ggplot(New_Recruits, aes(x=Height))+geom_histogram(binwidth =
2,color="white",fill="royalblue3") +
  xlab("Height")+ylab("Number of Occurences") +
  ggtitle("Distribution of Variable Height") + theme_minimal()+theme(plot.title =
element_text(hjust=0.5))

#Weight distribution
ggplot(New_Recruits, aes(x=Weight))+geom_histogram(binwidth =
2,color="white",fill="royalblue3") +
  xlab("Weight")+ylab("Number of Occurences") +
  ggtitle("Distribution of Variable Weight") + theme_minimal()+theme(plot.title =
element_text(hjust=0.5))
```