Machine Learning Coursework 3

Imanol Belausteguigoitia
Goldsmiths, University of London

Overview

This document analyses the complete FIFA 19 video game dataset, which contains attributes for every player registered in the latest edition of FIFA 19 database. Exploratory analysis, data mining and modelling to predict the market value of footballers are contained in this coursework.

Workflow

1. Data exploration: review of data dimensionality, completeness, characteristics of variables, column modifications and new columns (which I introduced).
2. Exploratory analysis
3. Case Study: Manchester United's new signings
4. Market value modelling
5. What could have been better: personal reflections of aspects to improve in next projects.

Dataset

This visual shows the first 15 rows and 6 columns:

X	ID	Name	Age	Photo	Nationality
0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina
1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal
2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil
3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain
4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium
5	183277	E. Hazard	27	https://cdn.sofifa.org/players/4/19/183277.png	Belgium
6	177003	L. Modrić	32	https://cdn.sofifa.org/players/4/19/177003.png	Croatia
7	176580	L. Suárez	31	https://cdn.sofifa.org/players/4/19/176580.png	Uruguay
8	155862	Sergio Ramos	32	https://cdn.sofifa.org/players/4/19/155862.png	Spain
9	200389	J. Oblak	25	https://cdn.sofifa.org/players/4/19/200389.png	Slovenia
10	188545	R. Lewandowski	29	https://cdn.sofifa.org/players/4/19/188545.png	Poland
11	182521	T. Kroos	28	https://cdn.sofifa.org/players/4/19/182521.png	Germany
12	182493	D. Godín	32	https://cdn.sofifa.org/players/4/19/182493.png	Uruguay
13	168542	David Silva	32	https://cdn.sofifa.org/players/4/19/168542.png	Spain
14	215914	N. Kanté	27	https://cdn.sofifa.org/players/4/19/215914.png	France

Dimension

18207 observations 89 columns

Percentage of null values

0.001133037%

Variables

There is a total of 87 (ID and index are not included):

Age, Nationality, Overall, Potential, Club, Value, Wage, Preferred Foot, International Reputation, Weak Foot, Skill Moves, Work Rate, Position, Jersey Number, Joined, Loaned From, Contract Valid Until, Height, Weight, LS, ST, RS, LW, LF, CF, RF, RW, LAM, CAM, RAM, LM, LCM, CM, RCM, RM, LWB, LDM, CDM, RDM, RWB, LB, LCB, CB, RCB, RB, Crossing, Finishing, Heading, Accuracy, ShortPassing, Volleys, Dribbling, Curve, FKAccuracy, LongPassing, BallControl, Acceleration, SprintSpeed, Agility, Reactions, Balance, ShotPower, Jumping, Stamina, Strength, LongShots, Aggression, Interceptions, Positioning, Vision, Penalties, Composure, Marking, StandingTackle, SlidingTackle, GKDiving, GKHandling, GKKicking, GKPositioning, GKReflexes, and Release Clause.

Deleted columns

This columns will be deleted because they don't provide meaningful information to analyze in this coursework:

Photo, Flag, ID, Club.Logo, LS, ST, RS, LW, LF, CF, RF, RW, LAM, CAM, RAM, LM, LCM, CM, RCM, RM, LWB, LDM, CDM, RDM, RWB, LB, LCB, CB, RCB, RB.

Modified columns

5 columns were modifies for technical purposes:

Value (1), Wage (2), Release.Clause (3) → This variables contained money signs such as "$", "€", "K", or "M" that had to be removed to be analyzed statistically.

Weight (4), Height (5) → this two features contained characters such as "lbs" that had to be removed in order to run algorithms.

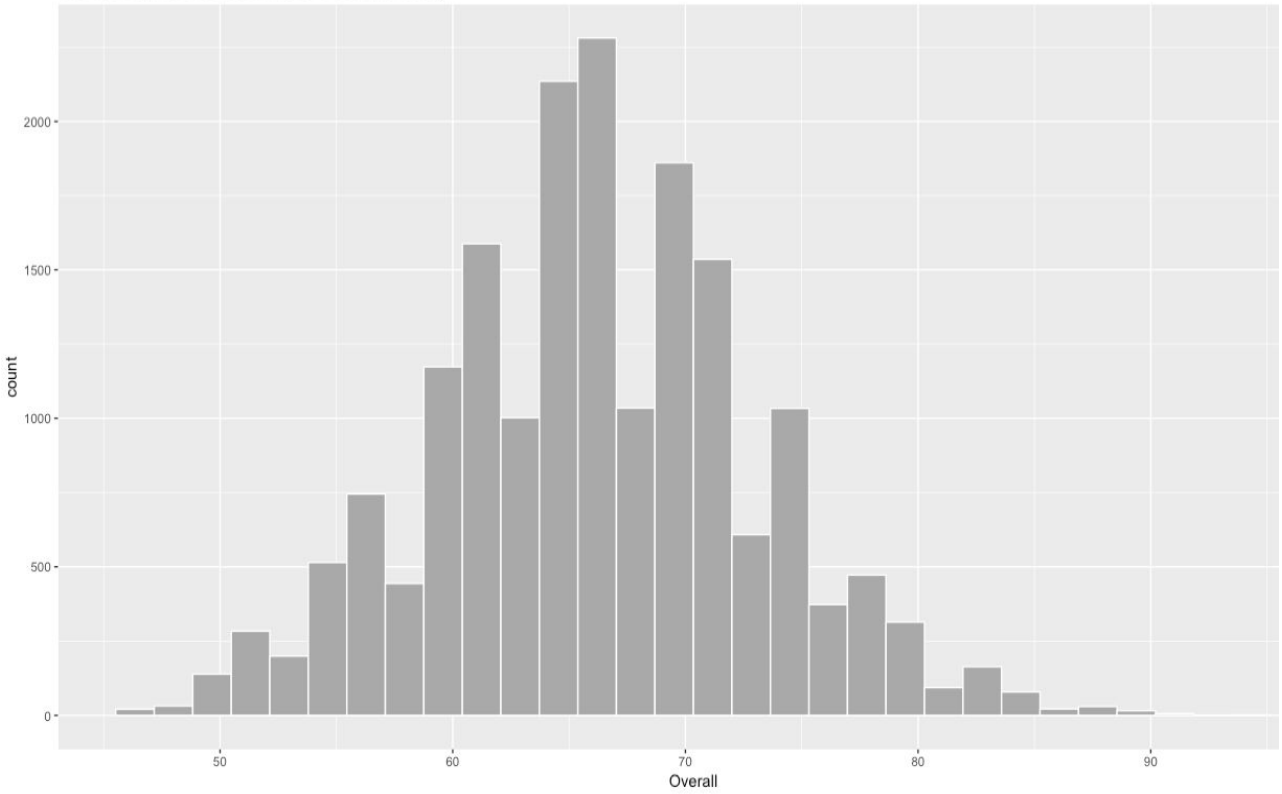
New column

Potential Gap: This is the difference between Potential and Overall. This will be used to predict market value and will be used for footballer signing decisions in the next sections.

Exploratory analysis

Overall

Player ratings are normally distributed

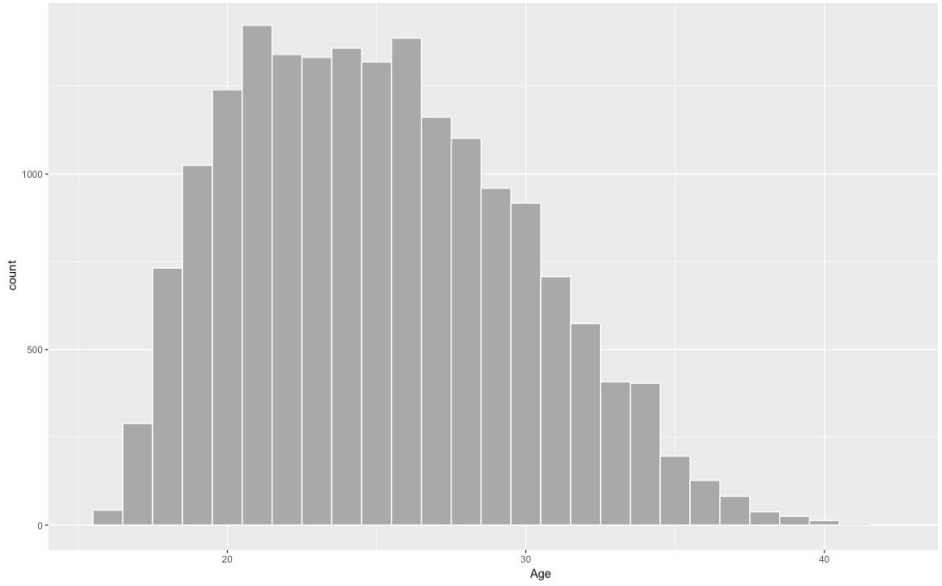


Summary statistics

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
46.00	62.00	66.00	66.24	71.00	94.00

Ages

Player ages **are not** normally distributed.

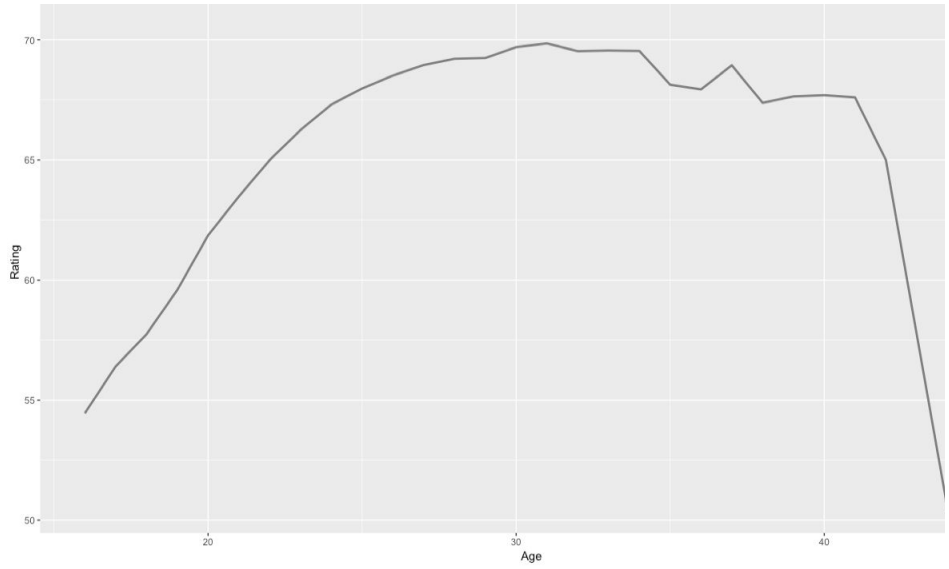


Summary statistics

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
16.00	21.00	25.00	25.12	28.00	45.00

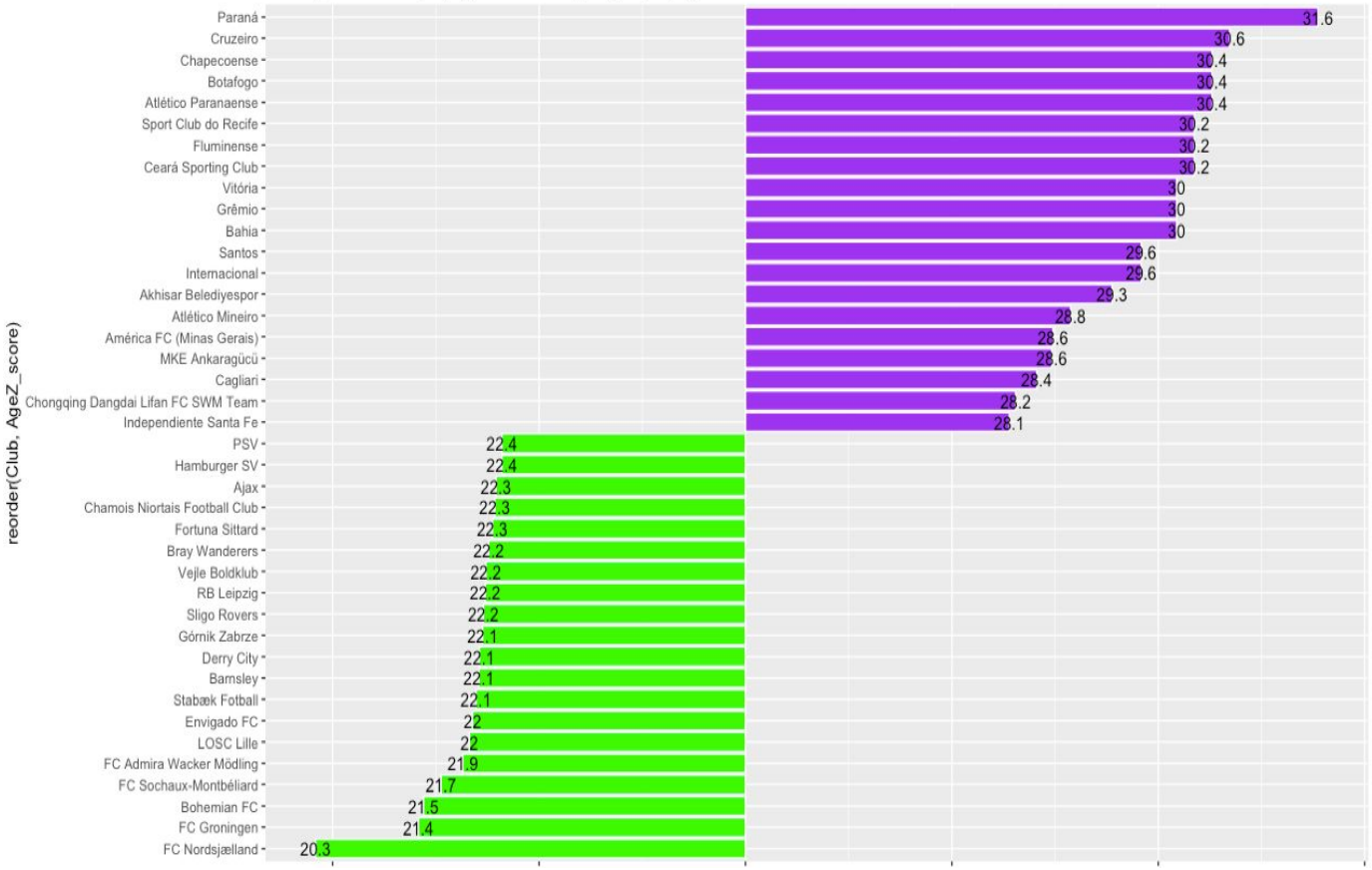
Rating vs Age

Players reach their peak level at low thirties and then decline.



Nordic Clubs are Younger than South American Clubs

20 oldest playing list vs 20 youngest playing list

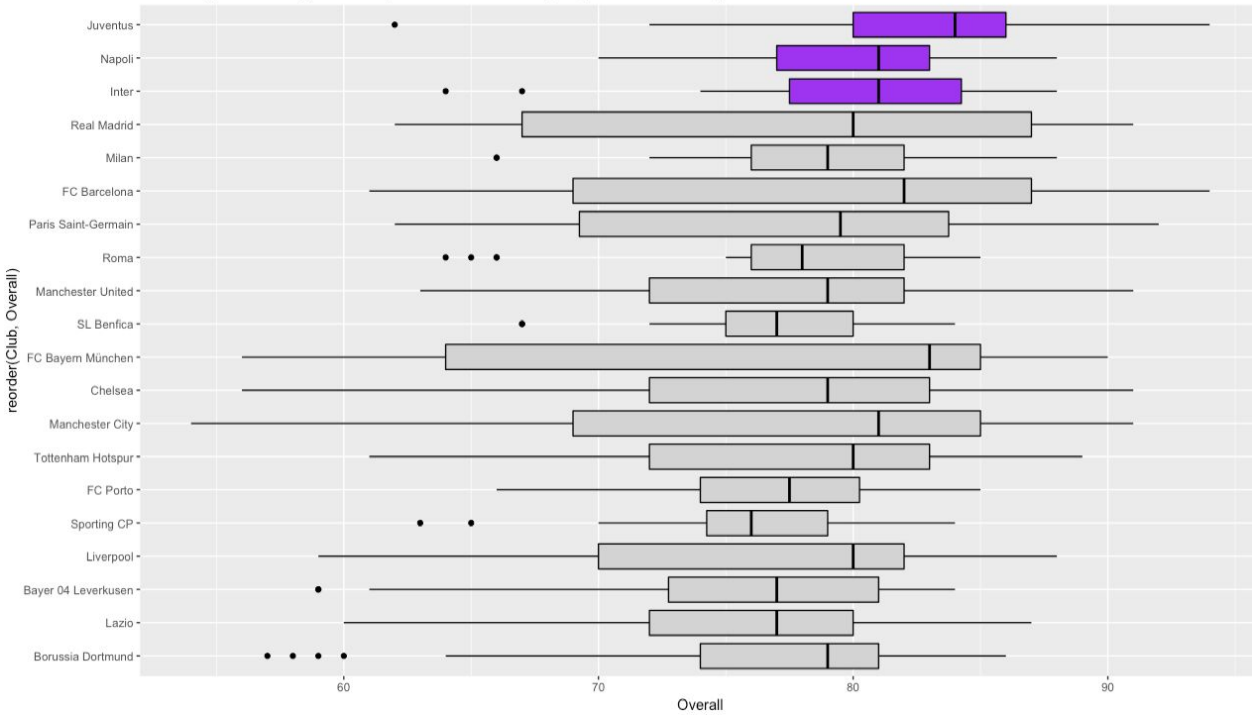


- Brazilian players tend to retire in Brazil after long careers in Europe, that fact is key to understand the extreme above average age for some of the clubs.

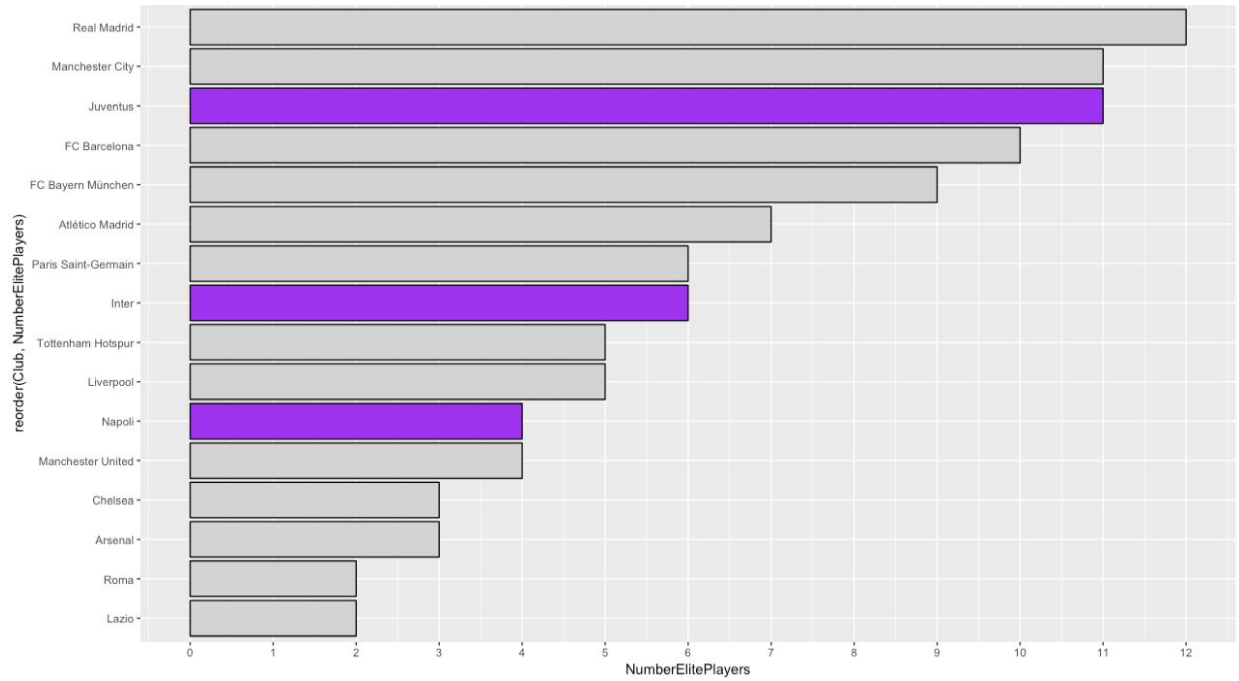
- Nordic countries don't have a big budget to retain young players, as soon as they prove to be talented they are likely to sign with another club.

Italian teams (purple) have the highest overall ratings

The average overall rating of the 20 highest rated teams in descending order



However they don't have the most elite players (defined as players with 85 or more overall score)



10 facts about the FIFA 19 Dataset

1. ### Percentage of left footed players
   30 percent

2. ### Nationalities with most players
   England (1662), Germany (1198), Spain (1072), Argentina(937), and France(914).

3. ### Nationality with most player in the top 100
   Spain (14)

4. ### Oldest player
   Mexican goalkeeper Oscar Pérez (45)

5. ### Best 10 player under 21 years
   K. Mbappé (88), M. de Ligt (82), G. Donnarumma (82), M. Rashford (81), L. Bailey (81)

6. ### England's best rated player
   Harry Kane (89)

7. ### Average age of the entire dataset
   25.12 years

8. ### Total teams
   652

9. ### Best 3 players
   Lionel Messi (94), Cristiano Ronaldo (94), Neymar (92)

10. ### Players with most expectations (defined as [Overall-Potential])
    J. von Moos, D. Campbell, Y. Lenze, B. Mumba, K. Askildsen

Manchester United Case Study

Context

Let's pretend we are part of the Manchester United recruitment team. For next season (2019-20) the team wants to sign players in order to compete for the Premier League title. There are four players that expire contract this season and need to be replaced:

- Alexis Sánchez, right winger
- Matteo Darmian, left back
- Juan Mata, right midfielder
- Ander Herrera, center midfielder

Suppose the club wants to find one replacement for each of those four player, plus one central defender to make their defense stronger since last season they lacked of top defenders.

The last seven years have not being good in terms of performance for the team. Since Sir Alex Ferguson, former manager leaved Manchester United, the club has not made the best managing decisions. Now with a new manager called Ole Gunnar Solskjaer the club wants to return to the previous signing strategy which contemplates the following points:

- Sign players that are **not at their peak level but they have potential** to be elite players (like they did with Cristiano, Beckham or Diego Forlan).
- Young players than **can be part of the team for at least 4 years** in a great level (less than 27 years old).
- **Avoid spending enormous amounts of money** for one player, that may be too risky (there are many examples in the post Ferguson era like Angel di María or Radamel Falcao).
- **Players that are talented today** and don't need more than one season to give great performances.

Approach

We will make a subset of players from the whole dataset that have the following characteristics:

- **FIFA19$Age < 27**: Players with less than 27 years old.
- **FIFA19$Overall>80**: Player with high levels of talent.
- **FIFA19$Potential.Gap > 0**: Players that are expected to improve in the following years.
- **FIFA19$Value < 80**: The maximum price they will pay is $ 80 million **USD to contract them**.
- **FIFA19$Release.Clause < 80**: The club wont sign footballers with high release clauses, the threshold selected is $ 80 million USD.

There is a pool of 123 players that have all the conditions above. Now that we have that pool, we we find players that have the same position as the players we want to replace. The players with the highest “Overall” score and that have the same positions as the players leaving the club (plus a centre back) are the ones that will be signed.

Above are the relevant code lines:

```
posible_man_united <- subset(FIFA19, FIFA19$Age < 27 & FIFA19$Overall>80 & FIFA19$Potential.Gap > 0 & FIFA19$Value < 80 & FIFA19$Release.Clause < 80)
dim(posible_man_united)

sanchez_replacement <- subset(posible_man_united, Position =="RW")
dim(sanchez_replacement)
head(sanchez_replacement[,2:8],5)
new_sanchez <- subset(head(sanchez_replacement[,2:8],1))

mata_replacement <- subset(posible_man_united, Position =="RM")
dim(mata_replacement)
head(mata_replacement[,2:8],5)
new_mata <-subset(head(mata_replacement[,2:8],1))

herrera_replacement <- subset(posible_man_united, Position =="CM")
dim(herrera_replacement)
head(herrera_replacement[,2:8],5)
new_herrera <- subset(head(herrera_replacement[,2:8],1))

new_centerback <- subset(posible_man_united, Position =="CB")
dim(new_centerback)
head(new_centerback[,2:8],5)
new_1_centerback <- subset(head(new_centerback[,2:8],1))

darmian_replacement <- subset(posible_man_united, Position =="LB")

dim(darmian_replacement)
head(darmian_replacement[,2:8],5)
new_darmian <- subset(head(darmian_replacement[,2:8],1))
```

New Signings

Rank	Name	Age	Nationality	Overall	Potential	Club	Value
168	T. Werner[1]	22	Germany	83	87	RB Leipzig	34.5 M
123	F. Thauvin[2]	25	France	84	87	Olympique de Marseille	39.0 M
122	Jorginho[3]	26	Italy	84	87	Chelsea	38.0 M
116	N. Süle[4]	22	Germany	84	90	FC Bayern München	36.5 M
86	D. Alaba[5]	26	Austria	85	87	FC Bayern München	38.0 M

Total Expenditure

186 million euros

Predicting Market Value

Context

Suppose Manchester United wanted to make a model to predict the market value of footballers in order to assign a market price to their own players and want to know the market value of players outside the club so that they make the best decisions in the next in the market windows.

Methodology

Target variable: Value (in million of dollars)

Type: regression

Evaluation metric: RMSE

Models deployed:

- Linear Regression (4 models)
- Stochastic Gradient Boosting (2 models)

[1] Alexis Sánchez's replacement
[2] Juan Mata's replacement
[3] Ander Herrera's replacement
[4] New centre back
[5] Matteo Darmian's replacement

- eXtreme Gradient Boosting (4 models)
- Random Forest (2 models)
- Support Vector Machines (1 model)

Data Pre Processing

The following steps were made to the original dataset after cleaning the data.[6] It is important to note that all models deployed only take numerical data.

The steps are detailed in the coding document.

Data preprocessing steps[7]

I. Dummification of the following variables:

- Position
- Prefered.Foot
- Work.Rate

II. Filtering to use only numerical data

I decided to do this in order to speed the modelling process, this lets me experiment more with tuning parameters and feature selection.

II. Drop near zero variance columns

III. Data center and scale data

IV. Data imputation (mean imputation)

V. Reduce skewness (Box-Cox Transformation)

At this stage I decided to create **two datasets:**

a) model_data.csv → A dataset without highly correlated values (the threshold was 0.7 of correlation coefficient). The dimension is of 59 columns and 18207 observations.

b) pca_data.csv → PCA was applied in this dataset. The dimension is of 47 columns and 18207 observations.

[6] Cleaning steps are explained in page 2.

[7] https://topepo.github.io/caret/available-models.html

Training Control

10 fold cross validation was applied to all models.[8]

Below is the formula used:

```
tc <- trainControl(method = "cv", number = 10)
```

Modelling

Linear Models (LM)

Four models were built: the first one contains all the regressors as shown below:

```
Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-23.430  -1.146  -0.022   1.037  79.299
```

It throws the following results:

```
Residual standard error: 3.533 on 18148 degrees of freedom
Multiple R-squared:  0.6153,    Adjusted R-squared:  0.614
F-statistic: 500.4 on 58 and 18148 DF,  p-value: < 2.2e-16
```

2nd model

This models contains feature selection with 20 regressors with p-value less than 0.001. This were the results:

```
Residual standard error: 3.585 on 18187 degrees of freedom
Multiple R-squared:  0.603,     Adjusted R-squared:  0.6026
F-statistic:  1454 on 19 and 18187 DF,  p-value: < 2.2e-16
```

[8] caret package was used is all the modelling process.

As it can be observed the results are marginally worse.

3rd Model

I tried another linear model with few regressors (less than 6) in order to experiment if there would be a significant change, but no important change in the performance was observed. This was what I would call and intuitive model, where according to my logic would be good predictors. There was practically no difference in performance since the RMSE was still 3.5.

4th Model

PCA was applied in a model that considered all the regressor and the improvement was dramatic as shown below:

```
Residual standard error: 0.959 on 18159 degrees of freedom
Multiple R-squared:  0.9716,    Adjusted R-squared:  0.9716
F-statistic: 1.324e+04 on 47 and 18159 DF,  p-value: < 2.2e-16
```

The RMSE was 0.959 as opposed to 3.5 of the three previous models.

From this point I decided to do only PCA with the rest of algorithms.

Stochastic Gradient Boosting (GBM)

1st model

This is the architecture of the first GBM model:

```
gbmFit1_pca <- train(Value ~ ., data = pca_data_1,
                     method = "gbm",
                     trControl = tc,
                     verbose = FALSE,
                     ## Only a single model can be passed to the
                     ## function when no resampling is used:
                     tuneGrid = data.frame(interaction.depth = 4,
                                           n.trees = 100,
                                           shrinkage = c(.1,.2,.3),
                                           n.minobsinnode = 20),
                     metric = "RMSE")
```

The results were not very promising as you may see below:

shrinkage	RMSE	Rsquared	MAE
0.1	1.194179	0.9558519	0.6275035
0.2	1.110323	0.9618886	0.5831319
0.3	1.121747	0.9611187	0.5821859

Therefore I decided to make a **second model** with more power and increased the number of trees and I left the best shrinkage observed in the previous model which was 0.2:

```
gbmFit2_pca <- train(Value ~ ., data = pca_data_1,
                     method = "gbm",
                     trControl = tc,
                     verbose = FALSE,
                     ## Only a single model can be passed to the
                     ## function when no resampling is used:
                     tuneGrid = data.frame(interaction.depth = 4,
                                           n.trees = 300,
                                           shrinkage = .2,
                                           n.minobsinnode = 20),
                     metric = "RMSE")
```

The results improved slightly, but not impressively so I decided to try with other algorithms.

RMSE	Rsquared	MAE
0.9893568	0.9699607	0.4971661

eXtreme Gradient Boosting (XGB)

The process of experimentation and testing made in the previous machine learning algorithm was applied for eXtreme gradient boosting.

These are the architectures of the four models deployed:

Model 1

```
xgbGrid_1 <- expand.grid(nrounds = c(1, 10, 15),
                         max_depth =  4,
                         eta = .1,
                         gamma = 0,
                         colsample_bytree = .7,
                         min_child_weight = 1,
                         subsample = c(.5,.8,1))

xgbFit1_pca <- train(Value ~ ., data = pca_data_1,
                     method = "xgbTree",
                     trControl = tc,
                     verbose = FALSE,
                     ## Only a single model can be passed to the
                     ## function when no resampling is used:
                     tuneGrid = xgbGrid_1,
                     metric = "RMSE")
```

Model 2

```
xgbGrid_2 <- expand.grid(nrounds = c(100, 150, 200),
                         max_depth =  4,
                         eta = .1,
                         gamma = 0,
                         colsample_bytree = .7,
                         min_child_weight = 1,
                         subsample = c(.5,.8,1))


xgbFit2_pca <- train(Value ~ ., data = pca_data_1,
                     method = "xgbTree",
                     trControl = tc,
                     verbose = FALSE,
                     ## Only a single model can be passed to the
                     ## function when no resampling is used:
                     tuneGrid = xgbGrid_2,
                     metric = "RMSE")
```

Model 3

```
xgbGrid_3 <- expand.grid(nrounds = c(250,400),
                         max_depth =  4,
                         eta = .1,
                         gamma = 0,
                         colsample_bytree = .7,
                         min_child_weight = 1,
                         subsample = c(.5,.8,1))


xgbFit3_pca <- train(Value ~ ., data = pca_data_1,
                     method = "xgbTree",
                     trControl = tc,
                     verbose = FALSE,
                     ## Only a single model can be passed to the
                     ## function when no resampling is used:
                     tuneGrid = xgbGrid_3,
                     metric = "RMSE")
```

Model 4

```
xgbGrid_4 <- expand.grid(nrounds = 1000,
                         max_depth =  4,
                         eta = .1,
                         gamma = 0,
                         colsample_bytree = .7,
                         min_child_weight = 1,
                         subsample = c(.5,.8,1))


xgbFit4_pca <- train(Value ~ ., data = pca_data_1,
                     method = "xgbTree",
                     trControl = tc,
                     verbose = FALSE,
                     ## Only a single model can be passed to the
                     ## function when no resampling is used:
                     tuneGrid = xgbGrid_4,
                     metric = "RMSE")
```

The best model out of the four XGB models resulted the best model overall, which was the fourth model. This model was extremely heavy in computation (It had 1000 nrounds but its performance was great as seen below:

subsample	RMSE	Rsquared	MAE
0.5	0.7251331	0.9821827	0.3476573
0.8	0.7329982	0.9830806	0.3383368
1.0	0.7217564	0.9835732	0.3410680

The best RMSE is 0.7217. This means that the model has an average error of $ 721,700 USD when estimating market value. This is a good model since some players can be worth up to $ 100 million USD and a mean of $ 5 million USD.

Two random Random Forest, models and one Support Vector Machines model were made but none of them achieved better performances. SVM proved to be very computationally expensive but achieved and RMSE of over 1.0. Below are the architecture of the three models mentioned.

First RF model

```
rfFit_pca1 <- train(Value ~ .,
                    data = pca_data_1,
                    method = 'ranger',
                    # should be set high at least p/3
                    tuneLength = 10,
                    trControl = tc,
                    ## parameters passed onto the ranger function
                    # the bigger the better.
                    num.trees = 15,
                    importance = "permutation")
```

Second RF model

```
rfFit_pca2 <- train(Value ~ .,
                        data = pca_data_1,
                        method = 'ranger',
                        # should be set high at least p/3
                        tuneLength = 10,
                        trControl = tc,
                        ## parameters passed onto the ranger function
                        # the bigger the better.
                        num.trees = 150,
                        importance = "permutation")
```

Support Vector Machines

```
svm_pca1 <- train(Value~., data=pca_data_1, method = "svmLinear", trControl = tc)
```

What could have been better

- Imputation: I decided to do a simple imputation method (mean) because only 0.001133037% of the values were null. It can be argued that more sophisticated algorithms could have been applied to fill the null values but I invested that time in experimenting with the modelling phase.

- Categorical columns: I got rid of two categorical columns which could have been very indicative but were not easy to dummify or treat (Club and Nationality). I did this because I thought I already had a lot of meaningful columns and I could spend more time experimenting with modelling. I am looking forward to treat categorical variables in a better way in next projects though.

- Understanding tunegrids: the way I achieved better performance in the models was by doing a baseline model and checked what tendencies the parameters showed without really understanding what the parameters meant. I am looking forward to study more in depth those parameters.

- Overall I wanted to make a project that was complete in the sense that I could show I have a **data scientist toolset**. This project has data inspections and cleaning, visualizations, data mining, data preprocessing, and machine learning. I am satisfied with the results of the models and I have some ideas as to how the results could improve (which I mention in the bullets above).