COGS 118A - Final Project

THE FIFA TRANSFERMARKET PREDICTION NOTEBOOK

Group members

- Khalid Ade
- Pablo Moreno
- Sujay Srinivasan
- Daniel Vega Lojo
- Jared Chen

Abstract

Association Football (or soccer) is a worldwide sport played by over 250 million players in over 250 countries[1]. In fact, football is the world's sport, and the most popular across the globe in terms of fans as well. Football has a huge transfer market in which players are transferred across teams for up to hundreds of millions of euros. To put the amount of money that circulates in the global football market into perspective, squad values of top teams like Manchester United surpass billions of euros[2]. The market value of a player accounts for a huge role in how teams conduct their business in regards to transfers. Our goal is to help these clubs make the right investments in players they want to obtain, especially when spending huge amounts of money. More specifically, we want to accurately predict the market value of players so that clubs aren't overpaying, or underselling their valued players. Plenty of factors play a role in determining the market value of a player. The most important factors include age, performance for club and national team (measured in stats such as goals, assists, tackles etc.) for a player in that position, experience(measured by number of seasons in top leagues), marketing value (measured by social media presence), and injury vulnerability[2].

Background

The global football transfer market involves the circulation of billions of euros. Many top European clubs have spent hundreds of millions of dollars to bolster their

respective teams. For example teams like Manchester United, Manchester City, and PSG have spent almost billions of euros to sign players to help their teams' success in their respective leagues and on the European stage[6]. There is no doubt that decisions involving huge sums of such money should be carefully analyzed so that clubs can maximize success in both the business side as well as the performance side of their respective clubs. Transfermarkt is an online platform for transfers, market values, rumors, and stats. The business model consists of, in addition to sports journalistic reporting, the profiles of the players and discussion forums on the performance and market values of individual soccer players, teams and leagues[4]. Frequently being discussed in sports science and sports economics literature over the past few years, the so-called "market values" („Marktwerte“) have s to become the center of media attentioMultipleous studies have shown positive correlations between the predicted market values on Transfermarkt and the actual player income.'It's reportedly known that players who are in contract negotiations would sometimes refer to Transfermarkt values as baselines for their salary expectations[4]. The “market values” can also be used as a measure of marketability; a higher marketability helps a player secure partnerships through sponsorship contracts. The age and performance statistics on Transfermarkt are also particularly useful in that player observers can identify young players and predict the development opportunities[4].

The open forums of Transfermarkt allow users to discuss and predict individual players' market values and performance. Previous studies on collective intelligence[2] have used OLS regression models to evaluate the accuracy of predictions. It is shown that “forecasts of international soccer results based on the crowd's valuations are more accurate than those based on standard predictors.”[3] This reveals a potential possibility that distributed intelligence is a contributing factor to the accuracy of predictions. We want to know if supervised machine learning algorithms, as another form of distributed intelligence, can make accurate predictions just as humans do. More particularly, we want to use machine learning models like OLS to predict market value of players across the football world.

Problem Statement

Given the considerable number of players in football across the globe, it can get tedious to know which players have potential and are worth investing in. Do they have high performance for a player in their position? Are they playing for a renowned club or in a renowned league? Is their behavior respectable and are they marketable?

These are the kinds of questions top clubs use when considering paying the big bucks for players. The problem we are trying to tackle is predicting the market value of players (in euros) using stats that are important when investing in a player such as goals, assists, and marketability.

Data

The dataset[7] is composed of 7 different subsets, we will be using 4 of the datasets. Since each feature resides in different sets.

- `Appearances.csv`
  - Player ID, Game ID, Appearance ID, Competition ID, Player club ID, Assist, Minutes Played, Yellow cards, Red Cards
- `Clubs.csv`
  - Club ID, Name, Pretty_name, Domestic_competition_id, Total_market_value, Squad_size, Average_age, Foreigners_numbers, Foreigners_percentage, National_team_players, Stadium_name, Stadium_seats, Net_transfer_record, Coach_name, URL
- `Competitions.csv`
  - Competition_id, Name, type, country_id, country_name, domestic_league_code, confederation, URL.
- `Games.csv`
  - Game_id, Competition_code, Season, Round, Date, Home_club_id, Away_club_id, Home_club_goals, away_club_goals, Home_club_postions, Away_club_postion, Stadium, Attendance, Referee, URL
- `Leagues.csv`
  - League_id, name, Confederation
- `Player_valuations.csv`
  - Player_id, Date, Market_value
- `Players.csv`
  - Player_id, Last_season, Current_club_id, Name, Pretty_name, country_of_birth, Country_of_citizenship, Date_of_birth, Position, Sub_position, Foot, Height_in_cm, Market_value_in_gbp, Highest_market_value_in_gbp, URL

- *What an observation consists of*: We are trying to use the variables we assume to be the most important and independent from each other. We decided on
    - Club, Nationality, Minutes, Goals, Assist, Age, Conduct, Years Played, Position, Physicality.
- *What some critical variables are, how they are represented*: We want variables which have the highest co-variance with each other. The metric should handle most features as unique features.
- *Any special handling, transformations, cleaning, etc will be needed*: There will be club names, and probably inferences in our data. Such as Media Presence or Potential, these are metrics which can be objective to the person. How popular is the player that we are analyzing?

We are still going to be in search of more databases that might have different descriptive data that we might like to see how organizations search for talent. We can use what they might describe as their most sought out characteristics.

For simplicity we can also assume that all players have no contracts for their evaluation and are based solely on performance and the other variables mentioned.

```
import sys
```

```
import re
_r = re.escape
def _re_replace(s : str, to_replace : dict):
    for p, r in to_replace.items():
        s = re.compile(p).sub(r, s)
    return s
```

```
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
%config InlineBackend.figure_formats = ['svg']
```

```
!{sys.executable} -m pip install --quiet pandas
import pandas as pd
```

```
'c:\Users\DanDan' is not recognized as an internal or external command,
operable program or batch file.
```

```
!{sys.executable} -m pip install --quiet seaborn
import seaborn as sns
```

```
'c:\Users\DanDan' is not recognized as an internal or external command,
operable program or batch file.
```

```
# OLS using statsmodels
!{sys.executable} -m pip install --quiet statsmodels numpy
import statsmodels.api as sm
import numpy as np
```

```
'c:\Users\DanDan' is not recognized as an internal or external command,
operable program or batch file.
```

```
!{sys.executable} -m pip install --quiet sklearn
!{sys.executable} -m pip install --quiet patsy
import sklearn as skl

import sklearn.linear_model

from sklearn.compose import ColumnTransformer
from sklearn.datasets import fetch_openml
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import Lasso
from sklearn.linear_model import ElasticNet
from sklearn.model_selection import KFold
import scipy.stats as stats
import patsy
```

```
'c:\Users\DanDan' is not recognized as an internal or external command,
operable program or batch file.
'c:\Users\DanDan' is not recognized as an internal or external command,
operable program or batch file.
```

```
_data_ = {
    name: pd.read_csv(
        file,
        engine = 'c',
        low_memory = True,
        memory_map = False, # set `False` to load into memory
        **kwargs
    ) for name, file, kwargs in [
        ('appearances', 'data/appearances.csv', {
            'dtype': {
                'player_id': 'object',
                'game_id': 'object',
                'appearance_id': 'object',
                'competition_id': 'object',
                'player_club_id': 'object'
            }
        }),
        ('clubs', 'data/clubs.csv', {
            'dtype': {
```

```
                'club_id': 'object'
            }
        }),
        #('competitions', 'data/competitions.csv', {}),
        ('games', 'data/games.csv', {
            'dtype': {
                'game_id': 'object'
            }
        }),
        #('leagues', 'data/leagues.csv', {}),
        ('players', 'data/players.csv', {
            'parse_dates': ['date_of_birth'],
            'dtype': {
                'player_id': 'object',
                'country_of_birth': 'category',
                'country_of_citizenship': 'category',
                'position': 'category',
                'sub_position': 'category'
            }
        }),
        ('player_valuations', 'data/player_valuations.csv', {
            'parse_dates': ['date'],
            'dtype': {
                'player_id': 'object'
            }
        })
    ]
}
```

```
data = {}
```

```
# clubs
data['clubs'] = _data_['clubs'].copy()

data['clubs'] = data['clubs'][[
    'club_id',
    'pretty_name'
]]
data['clubs'].rename(
    columns = {'pretty_name': 'club_name'},
    inplace = True
)
data['clubs'].set_index('club_id', inplace = True)

data['clubs']
```

club_id	club_name
1032	Fc Reading
2323	Orduspor
1387	Acn Siena 1904
3592	Kryvbas Kryvyi Rig
1071	Wigan Athletic
...	...
1269	Pec Zwolle
200	Fc Utrecht
317	Fc Twente Enschede
3948	Royale Union Saint Gilloise
1304	Heracles Almelo

801 rows × 1 columns

```
# games
data['games'] = _data_['games'].copy()

data['games'] = data['games'][[
    'season',
    'game_id'
]]
data['games'].set_index('game_id', inplace = True)

data['games']
```

	season
game_id	
2244388	2012
2219794	2011
2244389	2012
2271112	2012
2229332	2012
...	...
3646190	2021
3646188	2021
3655616	2021
3655629	2021
3646191	2021

56028 rows × 1 columns

```
# appearances
data['appearances'] = _data_['appearances'].copy()

data['appearances'] = data['appearances'].loc[
    :, ~data['appearances'].columns.isin([
        'appearance_id',
        'competition_id'
    ])
]
data['appearances'].rename(
    columns = {'player_club_id': 'club_id'},
    inplace = True
)

data['appearances'] = (
    data['appearances']
        .merge(
            data['games'],
            on = 'game_id',
            copy = False
        ).drop(columns = 'game_id')
        .merge(
            data['clubs'],
            on = 'club_id',
            copy = False
        ).drop(columns = 'club_id')
)
```

```
data['appearances'] = (
    data['appearances']
        .groupby(['player_id', 'season'])
        .agg({
            **{
                c: 'sum' for c in [
                    'goals',
                    'assists',
                    'minutes_played',
                    'yellow_cards',
                    'red_cards'
                ]
            },
            'club_name': 'last'
        })
        .reset_index('season')
)

data['appearances']
```

player_id	season	goals	assists	minutes_played	yellow_cards	red_cards	club_name
10	2014	32	18	4578	12	0	Lazio Rom
10	2015	16	14	3428	6	0	Lazio Rom
100009	2014	0	0	5576	8	0	Kuban Krasnodar
100009	2015	2	2	4512	12	0	Kuban Krasnodar
100009	2016	0	0	1260	6	0	Anzhi Makhachkala
...	...	...	...	...	...	...	...
99923	2014	0	2	832	4	0	Cagliari Calcio
99924	2016	0	2	1824	6	0	Ca Osasuna
99977	2014	0	0	194	0	0	Rcd Mallorca
99977	2015	10	6	3046	2	0	Royal Excel Mouscron
99977	2019	0	0	716	0	0	Caykur Rizespor

54216 rows × 7 columns

```
# player valuations
```

```
data['player_valuations'] = _data_['player_valuations'].copy()

data['player_valuations']['season'] = (
    pd.DatetimeIndex(data['player_valuations']['date']).year
)
data['player_valuations'].drop(columns = 'date', inplace = True)

data['player_valuations'] = (
    data['player_valuations']
        .groupby(['player_id', 'season'])
        .agg({'market_value': 'mean'})
        .reset_index('season')
)
data['player_valuations'].rename(
    columns = {'market_value_in_gbp': 'market_value'},
    inplace = True
)

data['player_valuations']
```

player_id	season	market_value
10	2004	6300000.0
10	2005	10800000.0
10	2006	22500000.0
10	2007	20700000.0
10	2008	18000000.0
...	...	...
99977	2018	990000.0
99977	2019	720000.0
99977	2020	562500.0
99977	2021	495000.0
99977	2022	540000.0

181182 rows × 2 columns

```
# players
data['players'] = _data_['players'].copy()

data['players'] = data['players'].loc[
    :, ~data['players'].columns.isin([
        'last_season',
        'name',
```

```
        'current_club_id',
        'market_value_in_gbp',
        'highest_market_value_in_gbp',
        'country_of_birth',
        'url',
        'foot'
    ])
]
data['players'].rename(
    columns = {
        'pretty_name': 'name',
        'height_in_cm': 'height',
        'country_of_citizenship': 'nationality'
    },
    inplace = True
)

data['players']['sub_position'] = (
    data['players']['sub_position'].cat
        .rename_categories(
            lambda s: (
                _re_replace(s, {
                    fr'''^(.*){_r(' - ')}(.*)$''': r'\2'
                })
                .title()
            )
        )
)

data['players'].set_index('player_id', inplace = True)

data['players']
```

player_id	name	nationality	date_of_birth	position	sub_position	height
254016	Arthur Delalande	France	1992-05-18	Midfield	Central Midfield	186
51053	Daniel Davari	Iran	1988-01-06	Goalkeeper	Goalkeeper	192
31451	Torsten Oehrl	Germany	1986-01-07	Attack	Centre-Forward	192
44622	Vladimir Kisenkov	Russia	1981-10-08	Defender	Right-Back	182
30802	Oscar Diaz	Spain	1984-04-24	Attack	Centre-Forward	183
...	...	...	...	...	...	...
462285	Fabian De Keijzer	Netherlands	2000-05-10	Goalkeeper	Goalkeeper	193
368612	Merveille Bokadi	DR Congo	1996-05-21	Defender	Centre-Back	186
408574	Joey Veerman	Netherlands	1998-11-19	Midfield	Central Midfield	185
364245	Jordan Teze	Netherlands	1999-09-30	Defender	Centre-Back	183
575367	Richard Ledezma	United States	2000-09-06	Attack	Attacking Midfield	174

23682 rows × 6 columns

```
# final dataset
data['all'] = data['players'].merge(
    data['player_valuations'].merge(
        data['appearances'],
        on = ['player_id', 'season'],
        copy = False
    ),
    on = 'player_id',
    copy = False
)

data['all']['age'] = (
    pd.to_datetime(data['all']['season'], format = '%Y', utc = True)
        - pd.to_datetime(data['all']['date_of_birth'], utc = True)
).astype('timedelta64[Y]')
data['all'].drop(columns = 'date_of_birth', inplace = True)

data['all'].dropna(axis = 'index', inplace = True)

data['all']
```

player_id	name	nationality	position	sub_position	height	season	market_value
9800	Artem Milevskyi	Ukraine	Attack	Centre-Forward	189	2020	90000.0
43084	Gaetano Berardi	Switzerland	Defender	Right-Back	179	2020	360000.0
230826	Gennaro Acampora	Italy	Midfield	Central Midfield	174	2020	360000.0
198087	Matteo Ricci	Italy	Midfield	Defensive Midfield	176	2020	1530000.0
110689	Deniz Mehmet	Turkey	Goalkeeper	Goalkeeper	192	2020	68000.0
...	...	...	...	...	...	...	..
364245	Jordan Teze	Netherlands	Defender	Centre-Back	183	2019	420000.0
364245	Jordan Teze	Netherlands	Defender	Centre-Back	183	2020	1102500.0
364245	Jordan Teze	Netherlands	Defender	Centre-Back	183	2021	5400000.0
575367	Richard Ledezma	United States	Attack	Attacking Midfield	174	2020	658250.0
575367	Richard Ledezma	United States	Attack	Attacking Midfield	174	2021	765000.0

50781 rows × 14 columns

Evaluation

```
data['all'][data['all'].isna().any(axis = 1)]
```

player_id	name	nationality	position	sub_position	height	season	market_value	goals

```
data['all'].dtypes
```

```
name                 object
nationality        category
position           category
sub_position       category
height                int64
season                int64
market_value        float64
goals                 int64
assists               int64
minutes_played        int64
yellow_cards          int64
red_cards             int64
club_name            object
age                 float64
dtype: object
```

```
data['all'].describe()
```

	height	season	market_value	goals	assists	minutes
count	50781.000000	50781.000000	5.078100e+04	50781.000000	50781.000000	50781
mean	180.794628	2017.380063	3.630890e+06	3.880546	2.949883	2805
std	17.703409	2.318805	8.274637e+06	7.352176	4.793814	2103
min	0.000000	2013.000000	9.000000e+03	0.000000	0.000000	2
25%	178.000000	2015.000000	3.600000e+05	0.000000	0.000000	884
50%	182.000000	2017.000000	9.000000e+05	0.000000	2.000000	2566
75%	187.000000	2019.000000	3.150000e+06	4.000000	4.000000	4410
max	206.000000	2021.000000	1.800000e+08	122.000000	62.000000	10122

```
pd.DataFrame(data['all']['sub_position'].unique())
```

	0
0	Centre-Forward
1	Right-Back
2	Central Midfield
3	Defensive Midfield
4	Goalkeeper
5	Centre-Back
6	Attacking Midfield
7	Right Winger
8	Left Winger
9	Left-Back
10	Left Midfield
11	Midfield
12	Second Striker
13	Right Midfield
14	Attack
15	Defender

```
data['all'][data['all']['name'] == 'Cristiano Ronaldo']
```

player_id	name	nationality	position	sub_position	height	season	market_value	g
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2014	96000000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2015	105000000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2016	99000000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2017	90000000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2018	96000000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2019	74250000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2020	54000000.0	
8198	Cristiano Ronaldo	Portugal	Attack	Centre-Forward	187	2021	39000000.0	

One hot encoding

```
# one hot encode categorical features
data['all_onehot'] = pd.get_dummies(data['all'], columns = [
    'position',
    'sub_position',
    'nationality',
    'club_name'
])

data['all_onehot']
```

player_id	name	height	season	market_value	goals	assists	minutes_played	yell
9800	Artem Milevskyi	189	2020	90000.0	0	0	720	
43084	Gaetano Berardi	179	2020	360000.0	0	0	228	
230826	Gennaro Acampora	174	2020	360000.0	2	4	1248	
198087	Matteo Ricci	176	2020	1530000.0	0	6	4880	
110689	Deniz Mehmet	192	2020	68000.0	0	0	1080	
...	...	...	...	...	...	...	...	
364245	Jordan Teze	183	2019	420000.0	0	0	360	
364245	Jordan Teze	183	2020	1102500.0	0	2	7494	
364245	Jordan Teze	183	2021	5400000.0	2	8	5260	
575367	Richard Ledezma	174	2020	658250.0	0	2	234	
575367	Richard Ledezma	174	2021	765000.0	2	0	88	

50781 rows × 588 columns

```
data['all_onehot'].dtypes
```

```
name                                object
height                               int64
season                               int64
market_value                       float64
goals                                int64
                                    ...
club_name_Yeni Malatyaspor           uint8
club_name_Zenit St Petersburg        uint8
club_name_Zirka Kropyvnytskyi        uint8
club_name_Zorya Lugansk              uint8
club_name_Zska Moskau                uint8
Length: 588, dtype: object
```

```
data['all_onehot'].describe()
```

	height	season	market_value	goals	assists	minutes
count	50781.000000	50781.000000	5.078100e+04	50781.000000	50781.000000	50781
mean	180.794628	2017.380063	3.630890e+06	3.880546	2.949883	2805
std	17.703409	2.318805	8.274637e+06	7.352176	4.793814	2103
min	0.000000	2013.000000	9.000000e+03	0.000000	0.000000	2
25%	178.000000	2015.000000	3.600000e+05	0.000000	0.000000	884
50%	182.000000	2017.000000	9.000000e+05	0.000000	2.000000	2566
75%	187.000000	2019.000000	3.150000e+06	4.000000	4.000000	4410
max	206.000000	2021.000000	1.800000e+08	122.000000	62.000000	10122

8 rows × 587 columns

```
data['all_onehot'][data['all_onehot']['name'] == 'Lionel Messi']
```

	name	height	season	market_value	goals	assists	minutes_played	yellow_c
player_id								
28003	Lionel Messi	169	2014	108000000.0	116	62	10122	
28003	Lionel Messi	169	2015	108000000.0	82	48	8458	
28003	Lionel Messi	169	2016	108000000.0	108	40	8904	
28003	Lionel Messi	169	2017	108000000.0	90	40	8936	
28003	Lionel Messi	169	2018	156000000.0	102	44	8048	
28003	Lionel Messi	169	2019	130500000.0	60	50	7262	
28003	Lionel Messi	169	2020	95400000.0	78	30	8746	
28003	Lionel Messi	169	2021	66000000.0	22	26	5384	

8 rows × 588 columns

Exploratory Data Analysis

```
data['all_eda'] = data['all'].copy()

data['all_eda']['log_market_value'] = np.log(data['all_eda']['market_valu
```

```
df_highest_market_value_players = data['all_eda'].nlargest(n = 1, columns

df_highest_market_value_players
```

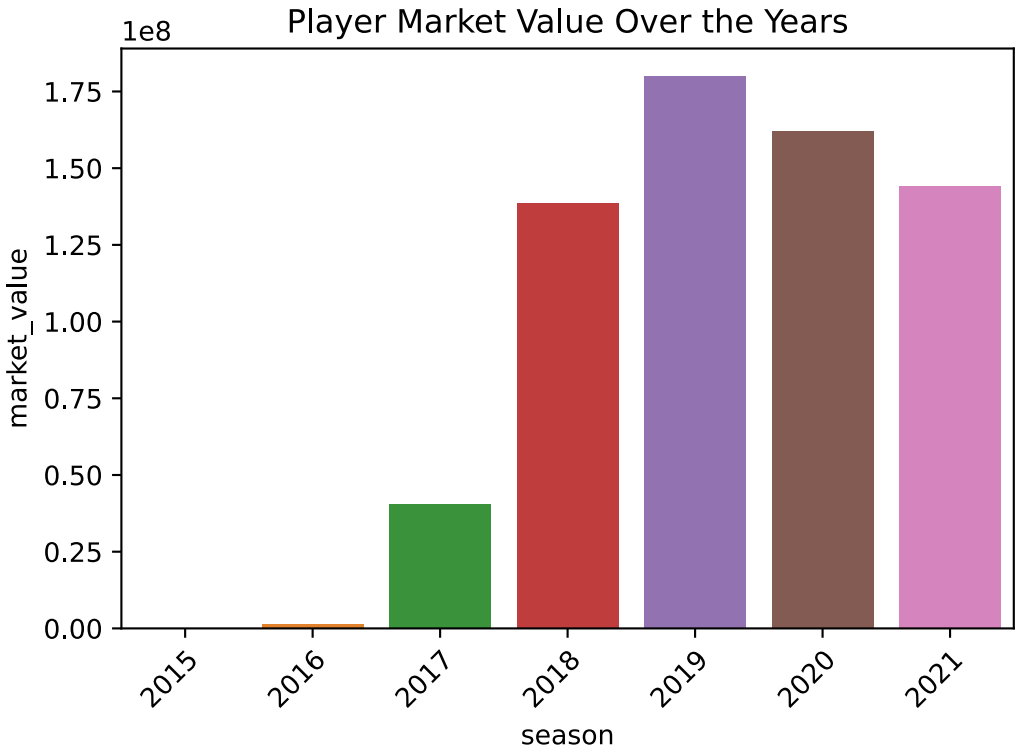
player_id	name	nationality	position	sub_position	height	season	market_value	go
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2019	180000000.0	

```
df_highest_market_value = data['all_eda'].loc[data['all_eda']['name'].isi

df_highest_market_value
```

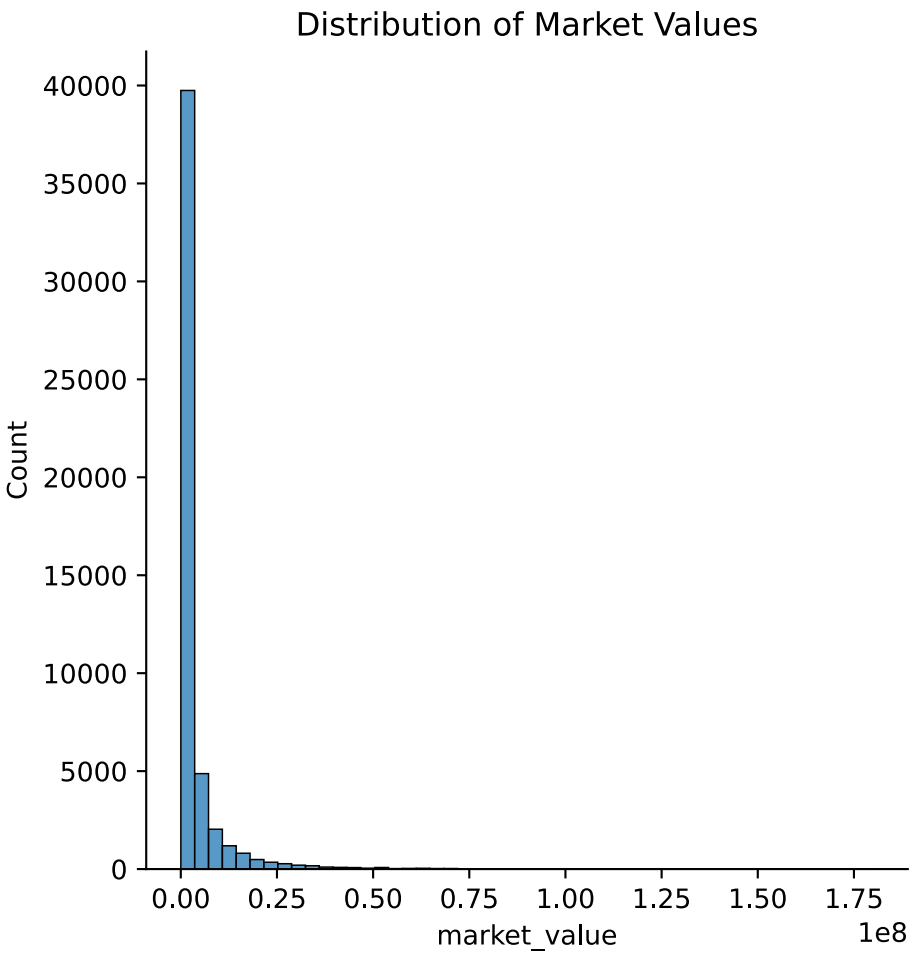
player_id	name	nationality	position	sub_position	height	season	market_value	go
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2015	45000.0	
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2016	1518750.0	
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2017	40500000.0	
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2018	138600000.0	
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2019	180000000.0	
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2020	162000000.0	
342229	Kylian Mbappe	France	Attack	Centre-Forward	178	2021	144000000.0	

```
_ = sns.barplot(
    data = df_highest_market_value,
    x = 'season', y = 'market_value'
).set(title = 'Player Market Value Over the Years')
plt.xticks(rotation = 45, ha = 'right', rotation_mode = 'anchor')
```

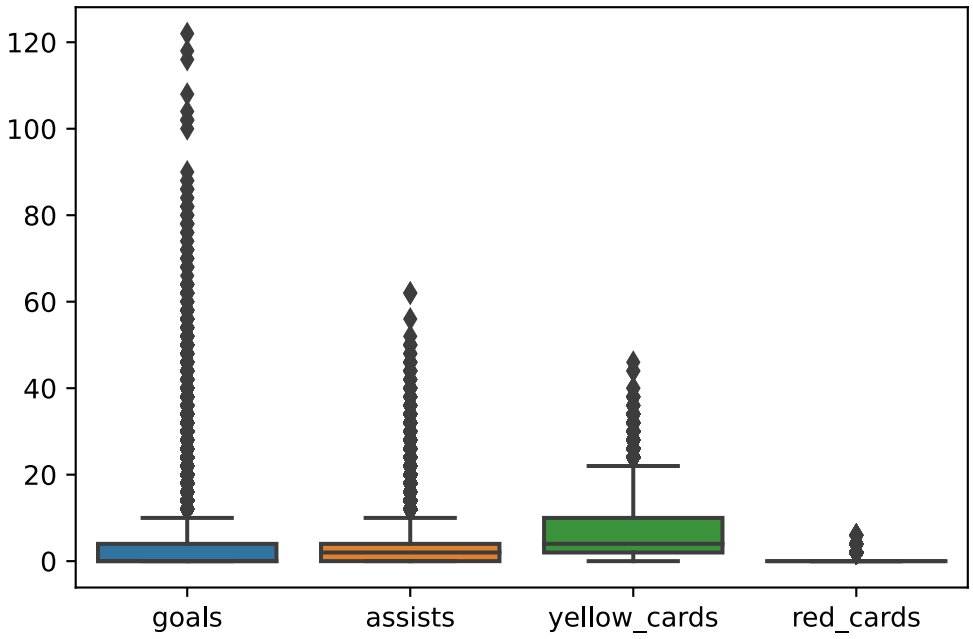
```
plt.show()
```



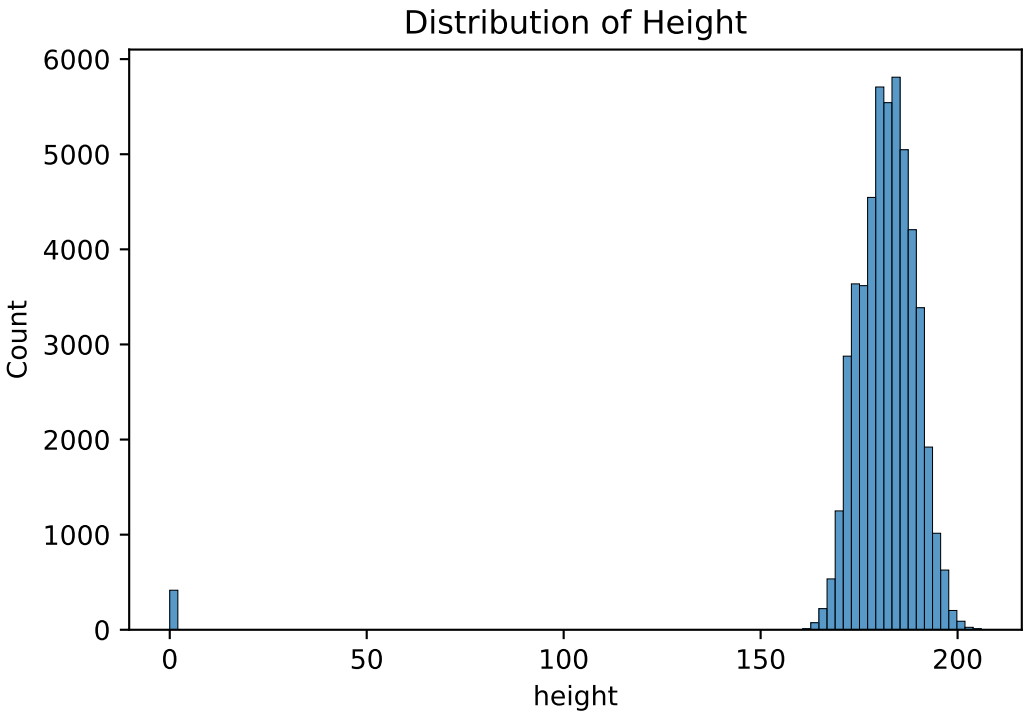

```
_ = sns.displot(
    data = data['all_eda'].reset_index(),
    x = 'market_value',
    bins = 50
).set(title = 'Distribution of Market Values')

plt.show()
```




```
_ = sns.boxplot(data = data['all_eda'][[
    'goals',
    'assists',
    'yellow_cards',
    'red_cards'
]]).set(title = '')
```




```
_ = sns.histplot(
    data = data['all_eda'].reset_index(), x = 'height',
    bins = 100
).set(title = 'Distribution of Height')
```



Proposed Solution

Packages:

- `sklearn` (scikit-learn)
- `StatsModel`
- `Seaborn`
- Ordinary Least square regression: `statsmodels.api`

We believe that we can take the characteristics football clubs may regard to be the most important in the dataset and use those features to evaluate players. Those can be our core information in order to use regression analysis. If able to determine a certain cluster for the data set depending on the position and attributes. We can compare the players in the data set with the new data. We will be using OLS (Ordinary Least Squares).

We have to start with the most important step which is data cleaning and EDA analysis in order to get more accurate results, Matrix transformation with the numpy library. In order to increase covariance with variables, we can use dimension reduction techniques. Next we can normalize the data sets in order to not get skewed by one particular feature. Dealing with values missing or if we need to have numeric values for non-numeric data (i.e popularity, health, position).

Our comparison of errors will be coming from the Transfermrket.com website as it is updated everyday to evaluate different players. We can get the player's information to get a percent error or a total error for our evaluations. If we can, we will use the RMSE (Root mean square error) or Mean Absolute Values of our model prediction.

Evaluation Metrics

We will be using an OLS regression model and the evaluation techniques we are considering are RMSE and Euclidean distance. A possible evaluation metric we will use is RMSE or Mean Absolute Value of Errors. It is derived by calculating the difference between the estimated and actual value, square those results, then calculate the mean of those results. The formula for RMSE is

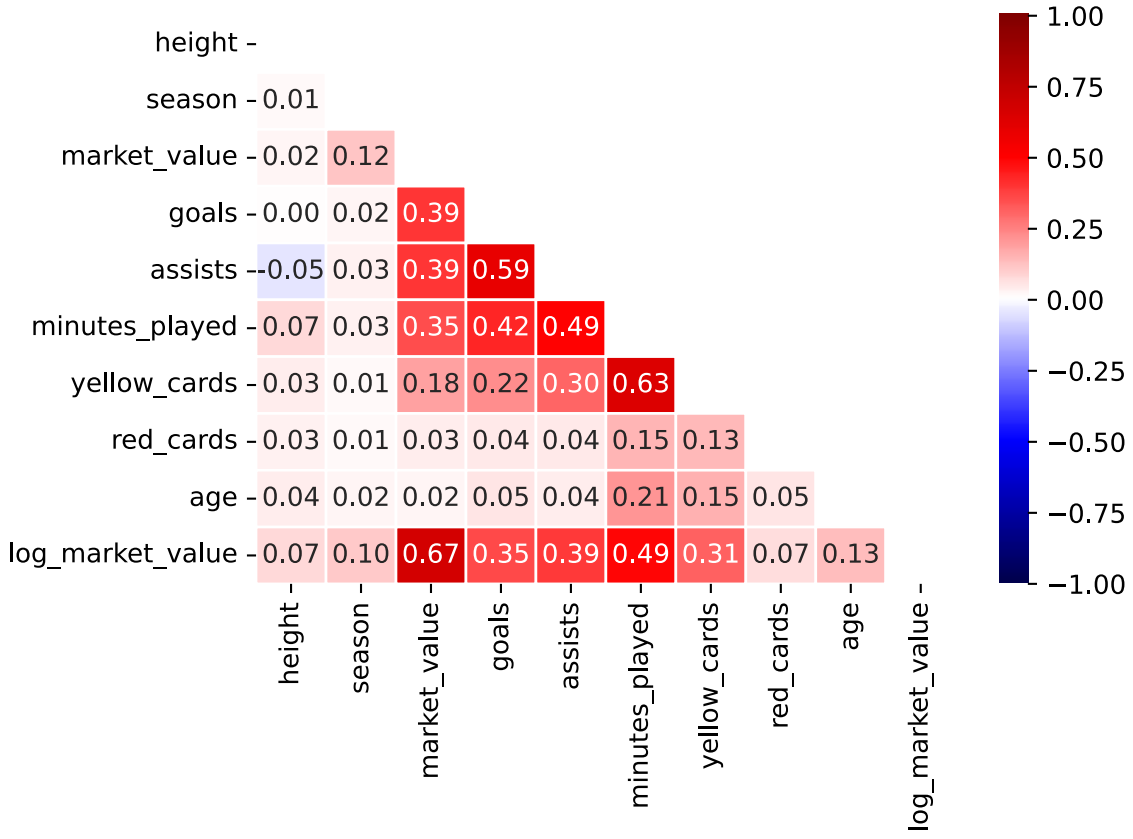
$$ \text{RMSE} = \sqrt{\frac{\sum_{i = 1}^{N}\left(\text{Predicted}_{i} - \text{Actual}_{i}\right)^{2}}{N}} $$

Results

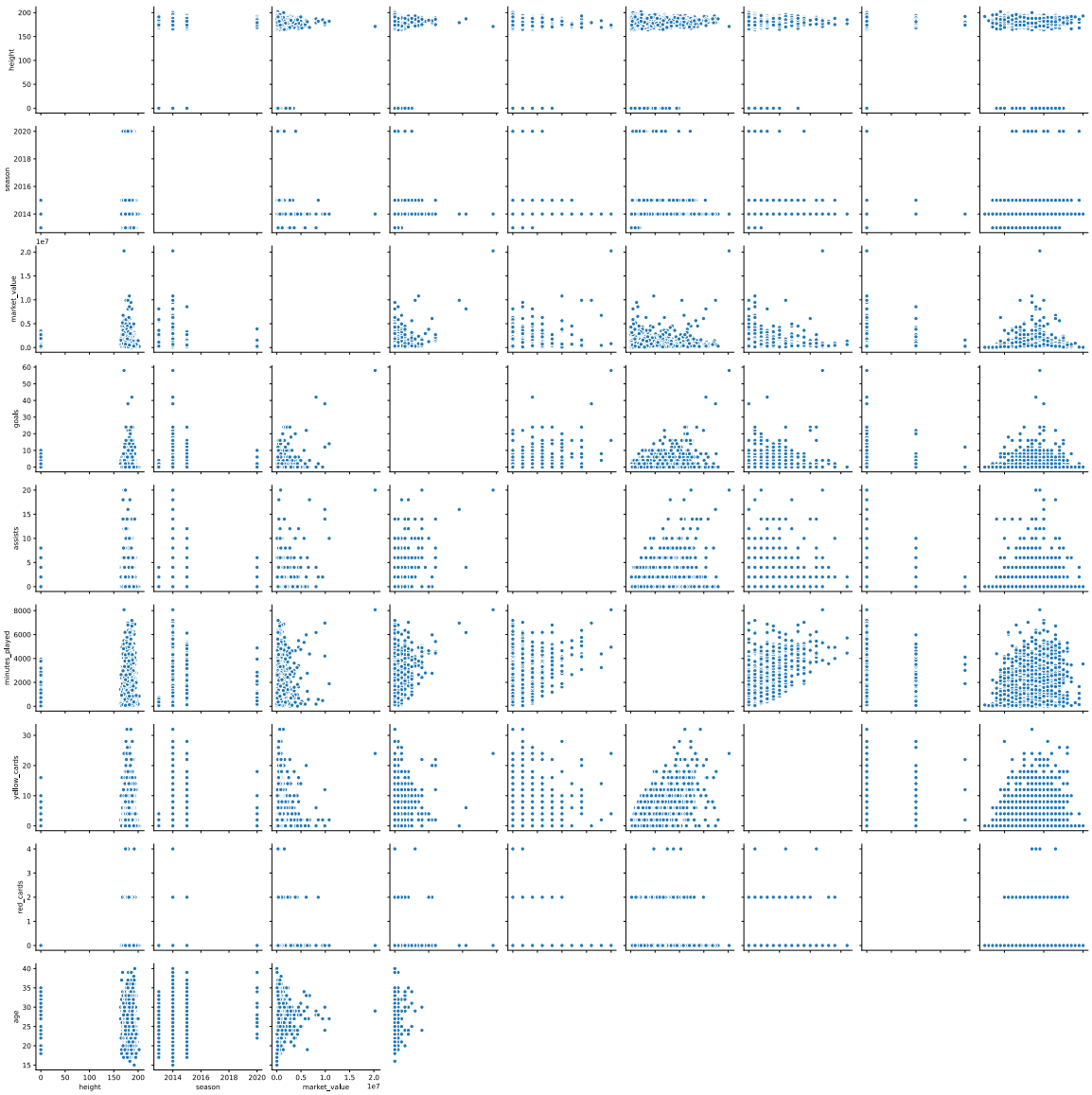
Subsection 1

We wanted to start by analyzing which data variables are important and correlate with each other using a heat map. The heat map will allow us to determine which are important to keep and also the pair plot will show the correlation between the variables.

```
corr = data['all_eda'].corr()
_ = sns.heatmap(corr,
    cmap = 'seismic',
    linewidth = 1, linecolor = 'white',
    vmax = 1, vmin = -1,
    mask = np.triu(np.ones_like(corr, dtype = bool)),
    annot = True,
    fmt = '0.2f'
)
```




```
_ = sns.pairplot(data['all'][:1500].reset_index())
```


height
season
market_value
goals
assists
minutes_played
yellow_cards
red_cards
age
1e7