Football Analytics - Analysing the FIFA 20 Player Ratings Dataset

Introduction

Football - or more familiarly known as "soccer" to some - is arguably the most popular sport in the world. It is a worldwide entertainment beloved by many; there's a reason why most nations have their own professional football league system, and why there's lots of national, continental, and sometimes global level football competitions every often. What else sports can match this level of popularity?

It shouldn't be surprising then that video games featuring real-world clubs and players are also very popular. One of them is the FIFA series, a football simulation video game series developed by a major video game company Electronic Arts. It's a series in that the creators release a follow-up version every year - the latest version as of now is FIFA 20.

As a rough introduction, each player in the game is assigned an overall player rating out of 100 based on his performance in the preceding year. Of course these figures are updated annually in accordance with the latest real-world data. In FIFA 20, the best player in the game - Lionel Messi - was given an overall rating of 94, with detailed specs of the following: 88 pace, 96 dribbling, 92 shooting, 39 defence, 92 passing, and 66 physical.

If you come to think about it, that's quite detailed and precise - especially considering how attributes like such are difficult to be accurately quantified. Perhaps not surprisingly, the figures often spark debates and controversies among football fans.

I wondered - how much are they relevant to the reality? How are the numbers assigned in the first place? Then again an idea struck my head - that this combined with other ideas could develop into an exciting DS project. And that's how it began for me...

With all these said, the aim of this project is to:

- Delve into questions like above using statistical methods and tools
- Extract lots of fun football facts through EDA
- Create lots of different visualisations because it's fun and images always help
- Have fun!

The Dataset

Thankfully, there's a dataset for FIFA 20, available on Kaggle, found here:

https://www.kaggle.com/stefanoleone992/notebook?select=players_20.csv (https://www.kaggle.com/stefanoleone992/notebook?select=players_20.csv)

As a side note, all the data in the dataset are based on the 2018-19 season.

This dataset consists of whopping 104 columns - description for each column can be found in the link above. Number of observations: 18278.

In [1]:

```
import pandas as pd
fifa = pd.read_csv("/Users/Jun/Desktop/fifa20.csv")
```

Toolsets

- Language: Python
- Libraries: NumPy, pandas, Matplotlib, Seaborn, Plotly
- Development Environment: Jupyter Notebook

Data Preprocessing

As with every other data-driven project, data preprocessing (i.e. data cleaning) must be preceded before any analysis. Clearly not all 104 columns will be in use, and therefore the first step is to drop unnecessary columns.

After some thought I decided to keep the following columns and drop the rest: long_name, age, height_cm, weight_kg, nationality, club, overall, value_eur, wage_eur, player_positions, preferred_foot, weak_foot, skill_moves, work_rate, pace, shooting, passing, dribbling, defending, physic, gk_diving, gk_handling, gk_kicking, gk_reflexes, gk_speed, gk_positioning

The rest have been excluded due to one or more of the following reasons:

- the feature is completely meaningless/adds literally nothing
- the feature could be useful but I simply wasn't interested at all
- the feature could be useful but dropped it for simplicity's sake

In [2]:

```
keep = ["long_name", "age", "height_cm", "weight_kg", "nationality"
fifa = fifa.loc[:, keep]
fifa.head(10)
```

Out[2]:

	long_name	age	height_cm	weight_kg	nationality	club	overall
0	Lionel Andrés Messi Cuccittini	32	170	72	Argentina	FC Barcelona	94
1	Cristiano Ronaldo dos Santos Aveiro	34	187	83	Portugal	Juventus	93
2	Neymar da Silva Santos Junior	27	175	68	Brazil	Paris Saint-Germain	92
3	Jan Oblak	26	188	87	Slovenia	Atlético Madrid	91
4	Eden Hazard	28	175	74	Belgium	Real Madrid	91
5	Kevin De Bruyne	28	181	70	Belgium	Manchester City	91
6	Marc-André ter Stegen	27	187	85	Germany	FC Barcelona	90
7	Virgil van Dijk	27	193	92	Netherlands	Liverpool	90
8	Luka Modrić	33	172	66	Croatia	Real Madrid	90
9	Mohamed Salah Ghaly	27	175	71	Egypt	Liverpool	90

10 rows × 26 columns

The column names are quite self-explanatory, but just to be clear with some of them,

- overall: player's overall rating, out of 100 (base rating, no upgrades or whatever obviously)
- value_eur: player's market value, in euros
- wage_eur: player's weekly wage, in euros
- weak_foot: player's weak foot (i.e. non-preferred foot) rating, 1-5
- skill_moves: player's skill moves rating, 1-5
- work_rate: how a player puts effort to participate in attacks and defences (i.e. (Low/High) means low effort in attack and high effort in defence)
- pace, dribbling, ..., gk_positioning: specific attributes out of 100

The next step is to ensure that no player's data have been duplicated.

In [3]:

```
sum(fifa.duplicated()) # no duplication
```

Out[3]:

```
0
```

note: sum(fifa.duplicated("long_name")) returns 60, but this isn't problematic since two different footballers can have the exact same long name (yes, this is possible, i.e. "Ben Davies" at Tottenham Hotspur & "Ben Davies" at Preston North End)

Then for each feature, ensure that there's no ridiculous value:

In [4]:

```
fifa["age"].unique()
```

Out[4]:

```
array([32, 34, 27, 26, 28, 33, 20, 25, 31, 30, 24, 29,
23, 19, 22, 37, 36,
       21, 41, 38, 35, 18, 40, 39, 17, 16, 42])
```

In [5]:

```
fifa["height_cm"].unique()
```

Out[5]:

```
array([170, 187, 175, 188, 181, 193, 172, 178, 191, 19
2, 168, 173, 184,
       182, 189, 176, 177, 199, 194, 185, 183, 180, 17
9, 169, 186, 174,
       163, 165, 195, 190, 196, 171, 197, 167, 198, 16
6, 164, 200, 161,
       158, 201, 203, 162, 157, 160, 159, 156, 202, 20
5])
```

In [6]:

```
fifa["weight_kg"].unique()
```

Out[6]:

```
array([ 72,  83,  68,  87,  74,  70,  85,  92,  66,  7
1,  73,  89,  91,
        82,  86,  80,  76,  75,  84,  69,  96,  67,  7
7,  78,  64,  81,
        61,  94,  59,  60,  90,  79,  97,  65,  95,  9
3,  63,  62,  88,
       100,  98,  58, 101, 103,  56,  99,  57, 102,  5
5,  52,  54, 104,
       107,  53, 110,  50])
```

Inspection on all the other numerical columns confirms that all values are sensible. Next up - missing values:

In [7]:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.heatmap(fifa.isnull(), cbar = False)
```

Out[7]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x116257748>
```

All the necessary data values are present. The features "pace", "shooting", "passing", "dribbling", "defending", and "physic" are in complete contrast with the goalkeeper-associated features, which is exactly what I want to see (i.e. exactly how it is in the game). No further action is required for missing values and this dataset is good to go.

The Analysis

Now for the first investigation - how well do the game statistics reflect reality? To be specific, how well, in general, do the player attribute numbers evaluate performances of players in the real world?

But what probably should come off first is, how are the numbers measured in the first place? Incidentally this was once answered in an interview by ESPN FC with Michael Mueller-Moehring, overseer of player rating calculations, back in 2016: https://www.espn.com/soccer/blog/espn-fc-united/68/post/2959703/fifa-17-player-ratings-system-blends-advanced-stats-and-subjective-scouting (https://www.espn.com/soccer/blog/espn-fc-united/68/post/2959703/fifa-17-player-ratings-system-blends-advanced-stats-and-subjective-scouting)

In short, the player ratings system takes into account two main sources of data: 1) advanced stats, such as number of goals scored/successful passes/saves, etc. 2) subjective data from a team of thousand data editors, field experts, and in-person game-goers.

Despite their best efforts, however, the final output isn't totally free from misevaluation, and this is understandable - after all, football is a complex sports where so many variables come into play. As Mueller-Moehring rightly points out, "the stats are, in most cases, not taking into account very specific circumstances" - data simply can't tell everything.

Conclusion: they have a structured and rigid approach to their work, which renders the output reliable to some extent, yet inevitably subjectivity is still there, so do not take it seriously.

With all these said I judged that it is pointless to try gauaging how "valid" the numbers are, though this was precisely what drove me in the beginning.

- Evaluating how "accurate" something requires a "test set"(i.e. the "correct/truthful" values) and in this case that "test set" is unavailable. For instance, what advanced real statistics and what combination of them can be used to accurately tell how "good" a midfielder is? What's more, football is an extremely complex sport - data alone aren't enough.
- The ratings team, some of which are experts, have already been incorporating advanced stats. This further renders what I'm trying to accomplish meaningless.

Thus I decided to take a slight detour and delve into the following question instead:

Question: How are the in-game specific attribute ratings related to overall rating?

Specifically, for each playing position, what specific attributes hold greater weight when calculating overall rating? Or to put it more regression-friendly, what would a model to explain one's overall rating in terms of other specific ratings look like?

There are many different types of positions in football. After a thorough research I decided to partition all the unique playing positions into the following.

- Forward: ST, CF
- Winger: LW, RW
- Attacking Midfielder: CAM
- Central Midfielder: CM
- Side Midfielder: LM, RM
- Defensive Midfielder: CDM
- Centre Back: CB
- Full Back: LWB, RWB, LB, RB
- Goalkeeper: GK

Note: I initially thought of separating strikers from centre forwards, but the lack of "main" centre forwards in this dataset and the similarity between the two roles eventually led me to group them together.

Now a player can play different positions. It seems like, for each player, player_positions tend to list positions in an order of decreasing familiarity, so for those with multiple playing positions I chose the first one.

In [8]:

```
main_pos = fifa["player_positions"].str.split(",").str[0]
```

For forward (ST/CF):

```
In [9]:
```

```
fw = (main_pos == "ST") | (main_pos == "CF")
forward = fifa[fw].head(2500)
forward = forward[["long_name", "overall", "weak_foot", "skill_moves
forward.head(10)
```

```
Out[9]:
```

	long_name	overall	weak_foot	skill_moves	work_rate	pace	shoc
1	Cristiano Ronaldo dos Santos Aveiro	93	4	5	High/Low	90.0	
10	Kylian Mbappé	89	4	5	High/Low	96.0	
12	Harry Kane	89	4	3	High/High	70.0	
17	Sergio Leonel Agüero del Castillo	89	4	4	High/Medium	80.0	
19	Luis Alberto Suárez Díaz	89	4	3	High/Medium	73.0	
20	Robert Lewandowski	89	4	4	High/Medium	77.0	
22	Antoine Griezmann	89	3	4	High/High	81.0	
34	Edinson Roberto Cavani Gómez	88	4	3	High/High	75.0	
38	Pierre-Emerick Aubameyang	88	4	4	Medium/Low	94.0	
43	Son Heung-min	87	5	4	High/High	88.0	

Because work_rate is a categorical variable, it has to be replaced with dummy variables.

In [10]:

```
import pandas as pd
dummies = pd.get_dummies(forward.work_rate)
forward = pd.concat([forward, dummies], axis = 1)
```

Of the 2000 sample players chosen, 1500 were randomly chosen to feed the model. The remaining 500 will be used as the test set.

In [11]:

```
X = forward.drop(["long_name", "overall", "work_rate"], axis = 1)
Y = forward["overall"]
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size
```

Now for fitting a regression model. For variable selection I chose backward elimination.

In [12]:

```
# Backward Elimination

import statsmodels.api as sm
X_train = sm.add_constant(X_train)
cols = list(X_train.columns)

while len(cols) > 0:
    X_train = X_train[cols]
    model = sm.OLS(Y_train, X_train).fit()
    pvals = pd.Series(model.pvalues.values, index = cols)
    fmaxp = pvals.idxmax()
    if (pvals[fmaxp] > 0.05) & (fmaxp != "const"):
        cols.remove(fmaxp)
    else:
        break

print(cols)
```

```
['const', 'skill_moves', 'pace', 'shooting', 'passing'
, 'dribbling', 'physic']

//anaconda3/lib/python3.7/site-packages/numpy/core/fro
mnumeric.py:2389: FutureWarning: Method .ptp is deprec
ated and will be removed in a future version. Use nump
y.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Backward elimination has chosen variables above. In the end I chose to drop 'defending' and 'Medium/Low' as well because they hardly made any difference (in terms of BIC). Thus the final model:

In [13]:

```
opt = ["const", "pace", "shooting", "dribbling", "physic"]
X_opt_train = X_train[opt]

model = sm.OLS(Y_train, X_opt_train).fit()
model.summary()
```

Out[13]:

OLS Regression Results

```
     Dep. Variable:          overall        R-squared:      0.957
            Model:               OLS   Adj. R-squared:      0.956
           Method:     Least Squares      F-statistic:  1.028e+04
             Date:  Mon, 13 Jul 2020 Prob (F-statistic):      0.00
             Time:          16:07:15   Log-Likelihood:    -3125.9
No. Observations:              1875              AIC:      6262.
     Df Residuals:              1870              BIC:      6289.
         Df Model:                 4
  Covariance Type:         nonrobust

            coef  std err       t   P>|t|  [0.025  0.975]
    const  0.8473   0.368    2.300  0.022   0.125   1.570
     pace  0.0275   0.003    7.906  0.000   0.021   0.034
 shooting  0.5914   0.008   73.368  0.000   0.576   0.607
dribbling  0.2655   0.008   33.627  0.000   0.250   0.281
   physic  0.1255   0.004   32.613  0.000   0.118   0.133

       Omnibus:  47.997    Durbin-Watson:     2.081
Prob(Omnibus):   0.000  Jarque-Bera (JB):    74.327
          Skew:  0.243          Prob(JB):  7.25e-17
      Kurtosis:  3.846         Cond. No.  1.67e+03
```

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.67e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

Prob(Omnibus) indicates that the residuals do not follow Normal distribution - which basically means errors are not normally distributed. This, however, poses no problem especially since sample size is large. (If that wasn't helpful at all read here: https://thestatsgeek.com/2013/08/07/assumptions-for-linear-regression/ (https://thestatsgeek.com/2013/08/07/assumptions-for-linear-regression/))

Multicollinearity (i.e. two or more explanatory variables related to each other) seems to be a thing, as evinced in the large condition number. In general severe multicollinearity undermines regression coefficient estimates by inflating their variances (i.e. here std err). But even so in this case the standard errors remain relatively small, with all of them being less than 0.01. In other words, regression coefficients vary little from one trial to another (in fact the confidence intervals are telling). With this considered I concluded that multicollinearity does not, for the most part, temper with interpreting regression coefficients.

The adj. R-squared value (i.e. goodness-of-fit) is very high, at about 0.955. How will this model perform on the test set though?

In [14]:

```
X_opt_train = X_opt_train.drop(columns = "const")

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_opt_train, Y_train)
```

Out[14]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jo
bs=None, normalize=False)
```

In [15]:

```
X_opt_test = X_test[["pace", "shooting", "dribbling", "physic"]]
regressor.score(X_opt_test, Y_test)
```

Out[15]:

```
0.9639036936229834
```

Seems like the final model works fantastic on the test set. So there we have it:

Expected overall rating for a forward (i.e. main position is either ST or CF) = 0.8473 + 0.0275 * pace + 0.5914 * shooting + 0.2655 * dribbling + 0.1255 * physic

Test set score for this model: 0.96

Train/Test size: 1875/625

I applied the same procedure for the remaining eight playing position categories. The results are shown below:

In [16]:

```
models = ["E(Y) = 0.8473 + 0.0275*pace + 0.5914*shooting + 0.2655*d
test = [0.96, 0.985, 0.97, 0.96, 0.97, 0.975, 0.98, 0.97, 0.99]
tt_size = ["1875/625", "525/175", "750/250", "1500/500", "1500/500"
top3 = ["shooting, dribbling, physic", "dribbling, passing, shootin
index = ["Forward(ST/CF)", "Winger(LW/RW)", "Attacking Midfielder(C
```

In [17]:

```
d = {"position category": index, "final model": models, "test set s
table = pd.DataFrame(data = d)
table.set_index("position category", inplace = True)
pd.set_option('display.max_colwidth', 150)
table
```

Out[17]:

	final model	test set score (approx)	train/test size	top 3 attributes
position category				
Forward(ST/CF)	E(Y) = 0.8473 + 0.0275*pace + 0.5914*shooting + 0.2655*dribbling + 0.1255*physic	0.960	1875/625	shooting, dribbling, physic
[illegible]	E(Y) = -3.039 + 0.1301*pace + 0.1965*shooting +	[illegible]	[illegible]	dribbling,

Winger(LW/RW)	0.2676*passing + 0.4199*dribbling + 0.0339*physic	0.985	525/175	passing, shooting
Attacking Midfielder(CAM)	E(Y) = 1.4226 + 0.0458*pace + 0.1848*shooting + 0.3425*passing + 0.3914*dribbling + 0.0383*physic	0.970	750/250	dribbling, passing, shooting
Central Midfielder(CM)	E(Y) = 1.019 + 0.0622*shooting + 0.3936*passing + 0.3356*dribbling + 0.1658*defending + 0.0684*physic	0.960	1500/500	passing, dribbling, defending
Side Midfielder(LM/RM)	E(Y) = -0.5254 + 0.1233*pace + 0.0839*shooting + 0.3808*passing + 0.3701*dribbling + 0.0656*physic	0.970	1500/500	passing, dribbling, pace
Defensive Midfielder(CDM)	E(Y) = 1.5289 + 0.2279*passing + 0.1272*dribbling + 0.5359*defending + 0.1306*physic	0.975	1050/350	defending, passing, physic
Centre Back(CB)	E(Y) = 1.4127 + 0.0390*passing + 0.0452*dribbling + 0.7300*defending + 0.1852*physic	0.980	2250/750	defending, physic, dribbling
Full Back(LWB, RWB, LB, RB)	E(Y) = 0.5984 + 0.1364*pace + 0.1502*passing + 0.1090*dribbling + 0.5709*defending + 0.0593*physic	0.970	1950/650	defending, passing, pace
Goalkeeper(GK)	E(Y) = -0.9113 + 0.2430*gk_diving + 0.2154*gk_handling + 0.0543*gk_kicking + 0.2497*gk_reflexes + 0.2570*gk_positioning	0.990	1500/500	gk_positioning, gk_reflexes, gk_diving

And there you have it. Some side notes:

1) As a reminder, for players with multiple playing positions I only considered their "main one" (i.e. the position that appears on their card)

2) The explanatory variables I considered are the following: weak_foot, skill_moves, work_rate, pace, shooting, passing, dribbling, defending, physic, gk_diving, gk_handling, gk_kicking, gk_reflexes, gk_speed, gk_positioning. Note that none of the first three variables were included in any of the final models.

3) Although "top 3 attributes" lists three of the most contributive attributes in the order of decreasing significance, the order isn't always meaningful. This is especially the case since multicollinearity is present (that being said interpreting regression coefficients is valid for the most part). It's just that, for instance, we can't say "gk_positioning" is more important than "gk_reflexes" when the two coefficients differ by less than 0.01.

Empirical data suggests that, roughly speaking, if two regression coefficients differ by less than 0.04, there's no significant evidence that the seemingly larger one truly holds greater weight than the other one (i.e. comparison is unmeaningful).

4) The models only explain how, for each main playing position, overall rating is associated with the main 12 specific attributes. This is NOT the way the game makers calculate overall ratings.

Try them by yourself and see how they work!

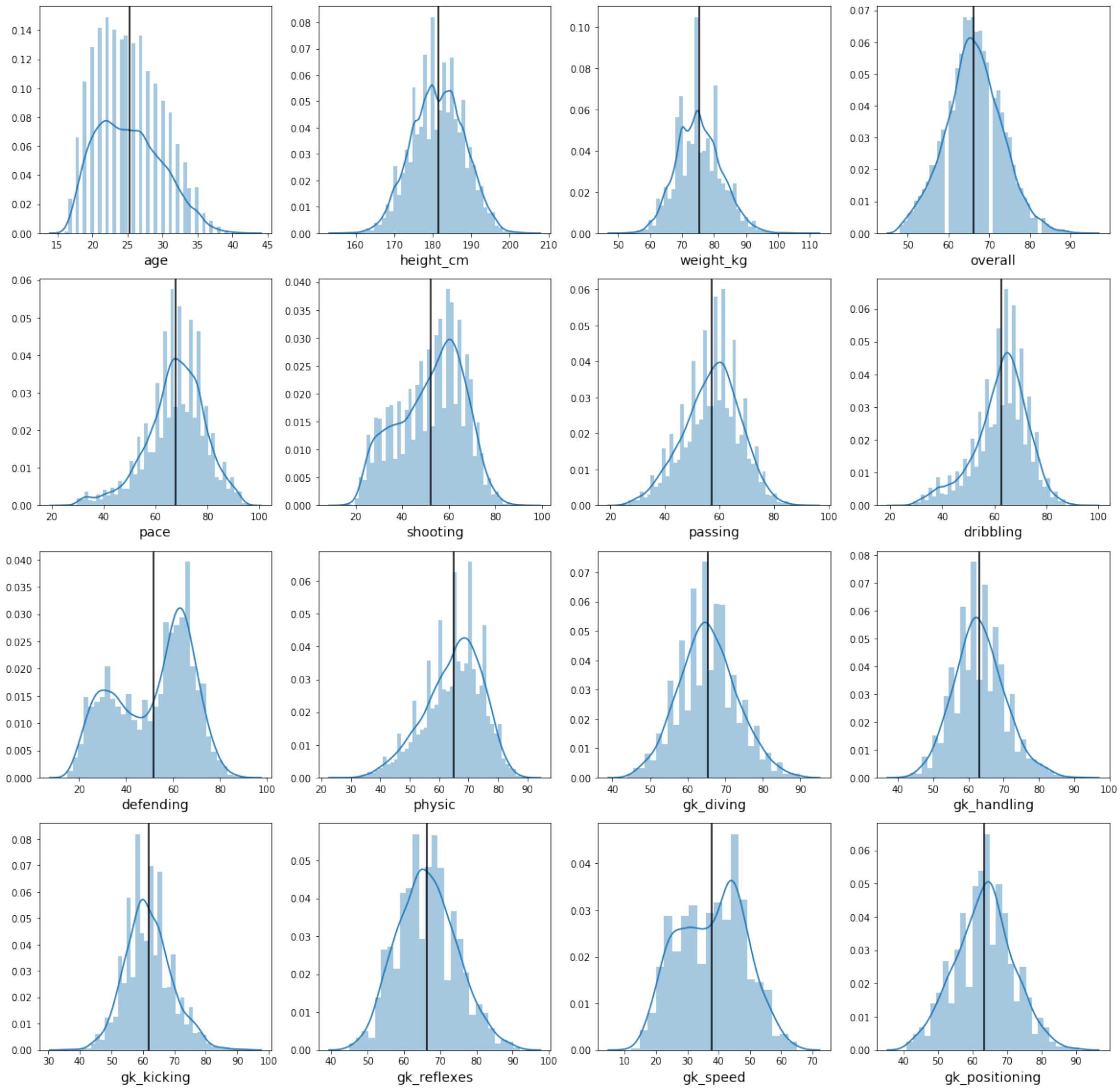
Next - exploratory data analysis.

Basic distributional information

Starting with the basics. Below are distribution for many of the feature columns I chose to explore.

In [18]:

```
fig = plt.figure(figsize = (20, 20))
hist_col = ["age", "height_cm", "weight_kg", "overall", "pace", "sho

d = {}
for i in range(16):
    d["ax{0}".format(i)] = fig.add_subplot(4, 4, i+1)
    d["ax{0}".format(i)] = sns.distplot(fifa[hist_col[i]].dropna())
    d["ax{0}".format(i)].axvline(fifa[hist_col[i]].mean(), color =
    d["ax{0}".format(i)].set_xlabel(hist_col[i], fontsize = 14)
```



Many of them are more or less normally distributed, though some are slightly skewed. It's interesting how all (except for gk_speed) goalkeeper attributes are normally distributed while the other six counterparts (pace, shooting, ..., physic) are not.

Note:

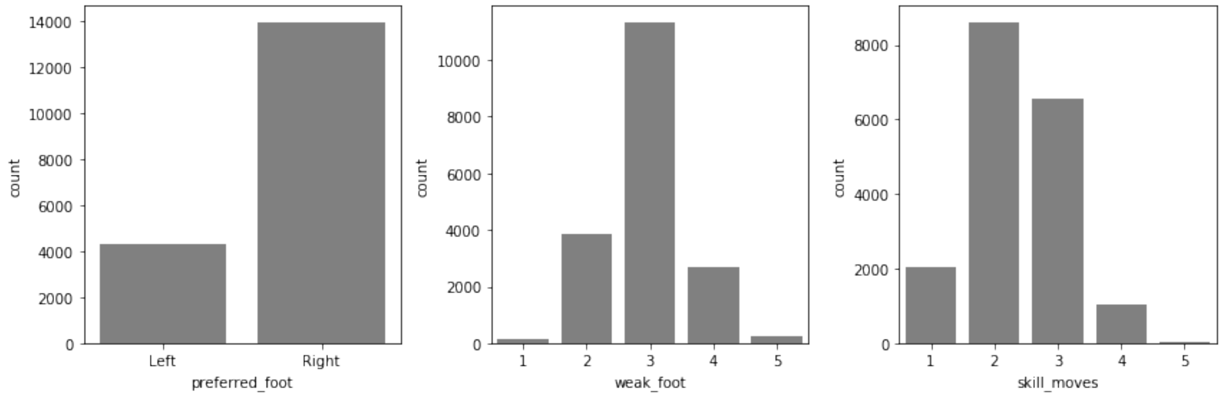
1) The black vertical lines indicate their respective mean values.

2) The y-values are to the y-values probability density plots.

Bar graphs:

In [19]:

```
fig = plt.figure(figsize = (12, 4))
bar_col = ["preferred_foot", "weak_foot", "skill_moves"]

d = {}
for i in range(3):
    d["ax{0}".format(i)] = fig.add_subplot(1, 3, i+1)
    d["ax{0}".format(i)] = sns.countplot(fifa[bar_col[i]], color = 

plt.tight_layout()
```

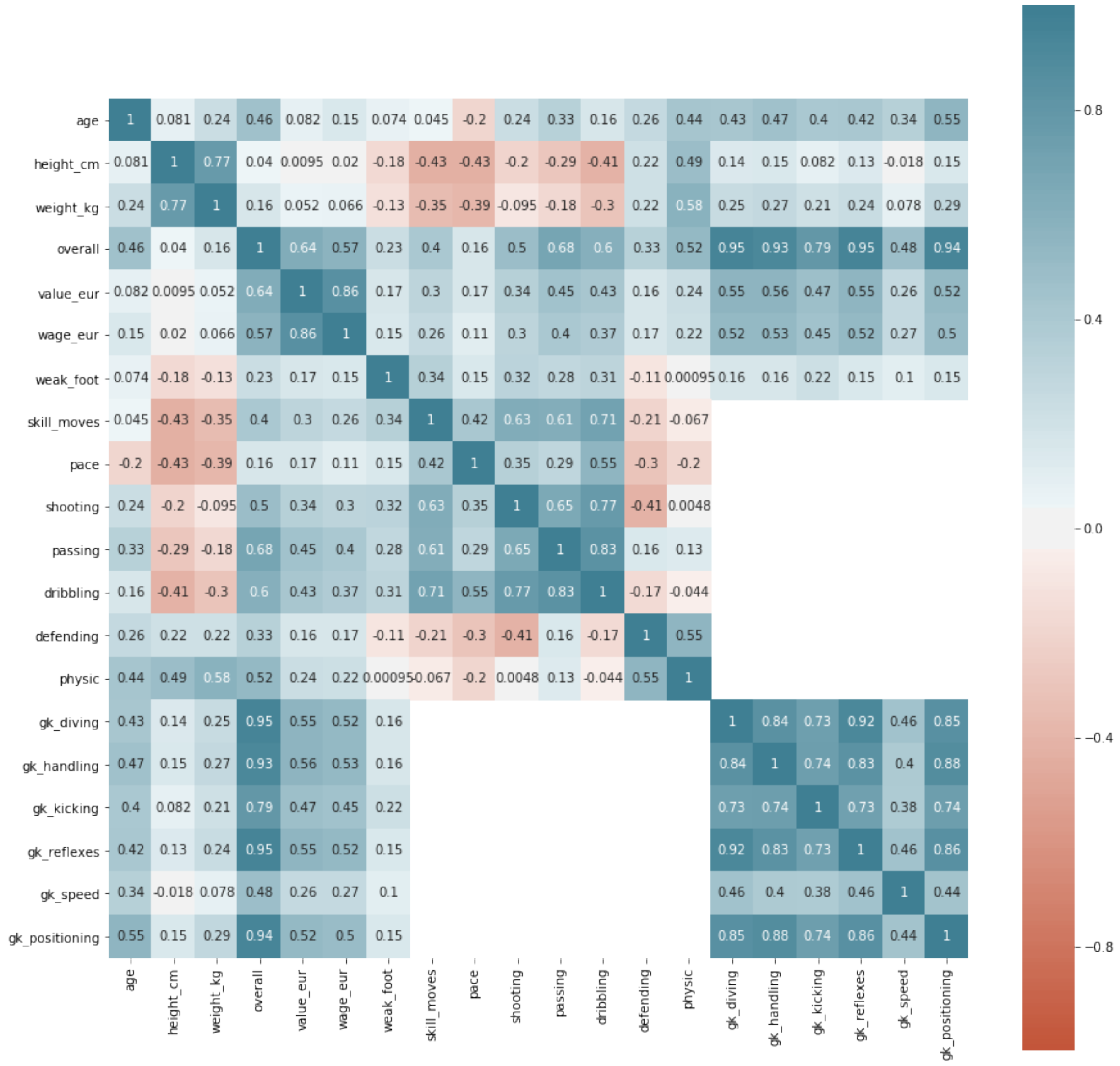


The correlation matrix plot

What two features are related to each other? The correlation matrix plot below has all the answers...

In [20]:

```
fig = plt.figure(figsize = (15, 15))
cor = fifa.drop(columns = ["long_name", "nationality", "club", "play
ax = sns.heatmap(cor, vmin = -1, vmax = 1, center = 0, cmap = sns.d
```



The numbers inside the square boxes are Pearson correlation coefficients (i.e. a value close to -1 indicates strong negative linear relationship between two variables and a value close to 1 strong positive linear relationship).

Note that many of the goalkeeper attributes are highly correlated with 'overall'. This backs the result found earlier, where it was concluded that the most important features for explaining player overall rating are: gk_diving, gk_handling, gk_reflexes, and gk_positioning. Here all of these have $r > 0.9$.

Strong ($r > 0.7$) correlation is seen in some of the other pairs, but many of them are not surprising. For instance, 'wage_eur' and 'value_eur' are strongly correlated, and this is not surprising at all.

Aside from these two pairs strike my attention - 'passing'&'dribbling' and 'shooting'&'dribbling'. The r values for these are 0.83 and 0.77, respectively. For some reason 'dribbling' has lots to do with 'passing'&'shooting', more so than 'pace' ($r = 0.55$), which is quite interesting and personally for me, counter-intuitive.

Perhaps the results table earlier may shed some light to this. According to the table, 'passing'&'dribbling' are more or less equally important for midfielders (CAM, CM, LM, RM), and these players make up a bulk of the original dataset. For 'pace'&'dribbling', more less so - in fact, none of the final models seem to treat the two equally.

The road to world-class

Becoming a top-of-the-world footballer is tough. To be one you've got to have skills and talents that set you apart big time. But just how much of different level are they?

For each of the main positions I defined above, I picked the top 20 players in the world in terms of overall rating (if 20 sounds too much, that's just over the number of starting goalkeepers that make it into the top 16 of the UEFA Champions League in one season) and compared them to their respective 75th percentile-ish and median-ish level players (so groups of 20 for each level). The results are below:

In [21]:

```
map_d = {'RW': 'Winger(LW, RW)', 'ST': 'Forward(ST, CF)', 'LW': 'Wi
main_pos_c = main_pos.map(map_d)
fifa["position_category"] = main_pos_c
```

In [22]:

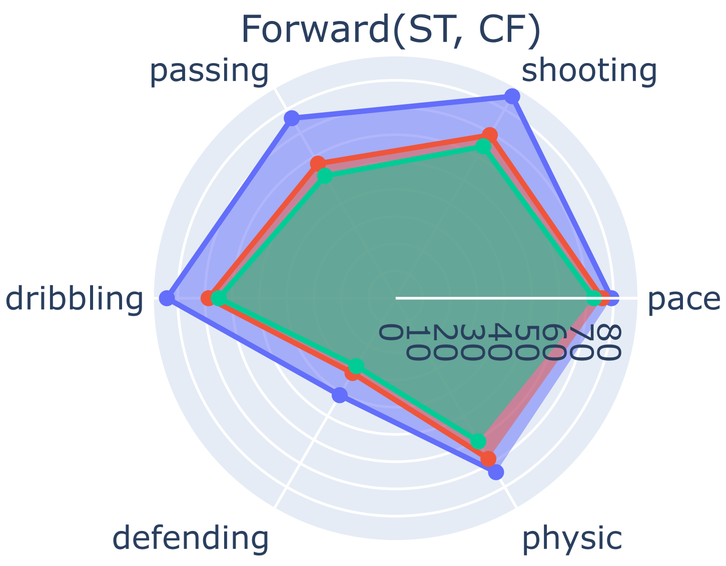
```
# top 20
import numpy as np
world_c = fifa.groupby("position_category").head(20)
wcavg = world_c.pivot_table(values = ['pace', 'shooting', 'passing'

# 75th percentile
def third_q(df):
    x = round(df.shape[0]*0.25)
    return df.iloc[(x-10):(x+10), ]

top_quantile = fifa.groupby("position_category").apply(third_q)
top_quantile = top_quantile.droplevel("position_category")
topqavg = top_quantile.pivot_table(values = ['pace', 'shooting', 'pa

# median
def median(df):
    x = round(df.shape[0]*0.5)
    return df.iloc[(x-10):(x+10), ]

mid_quantile = fifa.groupby("position_category").apply(median)
mid_quantile = mid_quantile.droplevel("position_category")
midqavg = mid_quantile.pivot_table(values = ['pace', 'shooting', 'pa
```

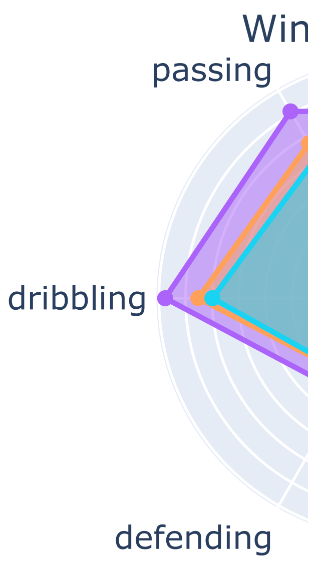
In [23]:

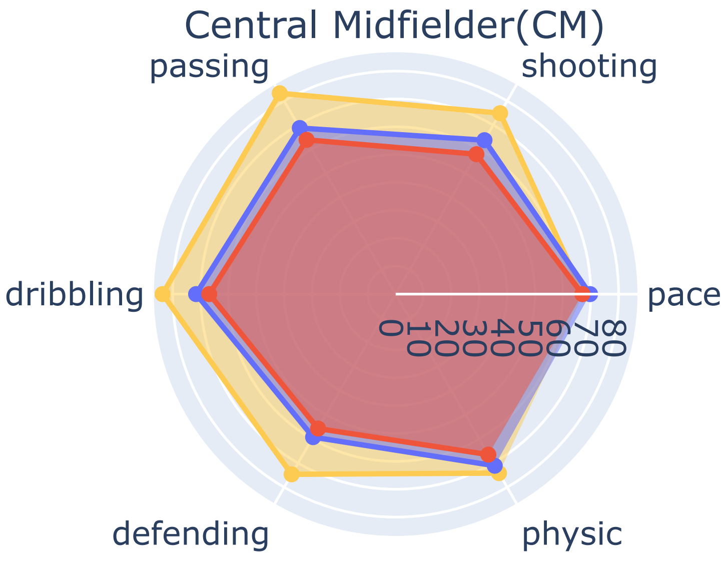
```
# Yeah I know this sucks, but Plotly lacks lots of support, includi

import plotly.graph_objects as go
from plotly.subplots import make_subplots

pos = ['Forward(ST, CF)', 'Winger(LW, RW)', 'Central Attacking Midf
       'Central Defensive Midfielder(CDM)', 'Centre Back(CB)', 'Full

spec = [[{'type':'polar'}, {'type':'polar'}, {'type':'polar'}],
         [{'type':'polar'}, {'type':'polar'}, {'type':'polar'}],
         [{'type':'polar'}, {'type':'polar'}, {'type':'polar'}]]

fig = make_subplots(rows = 3, cols = 3, start_cell = "top-left", sha

atts = ["pace", "shooting", "passing", "dribbling", "defending", "ph
gatts = ["gk_diving", "gk_handling", "gk_kicking", "gk_reflexes", "g
```

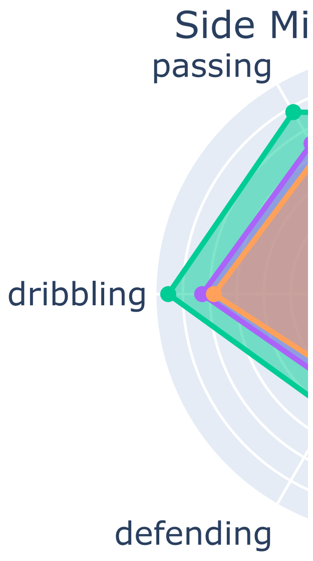
```
for i in fig['layout']['annotations']:
    i['font'] = dict(size = 14)

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[0], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[0], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[0], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[1], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[1], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[1], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[2], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[2], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[2], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[3], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[3], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[3], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[4], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[4], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[4], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[5], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[5], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[5], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[6], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[6], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[6], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[7], atts]), the
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[7], atts]), t
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[7], atts]), t

fig.add_trace(go.Scatterpolar(r = list(wcavg.loc[pos[8], gatts]), th
fig.add_trace(go.Scatterpolar(r = list(topqavg.loc[pos[8], gatts]), 
fig.add_trace(go.Scatterpolar(r = list(midqavg.loc[pos[8], gatts]), 

fig.update_layout(height = 1000, width = 1000, title_text = "World-
```

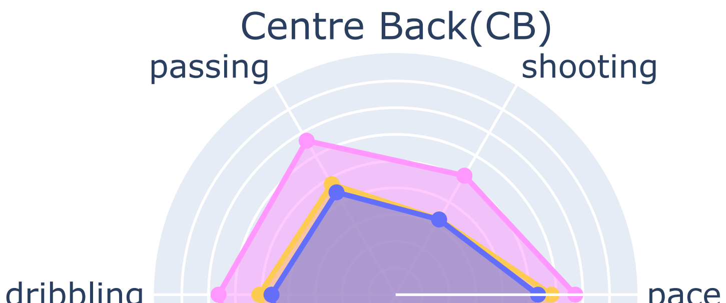
World-Class v. Top Quarter v. Median

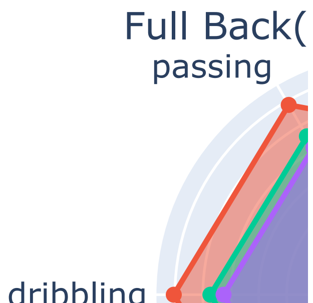

Forward(ST, CF)
passing
shooting
dribbling
pace
0
10
20
30
40
50
60
70
80
defending
physic

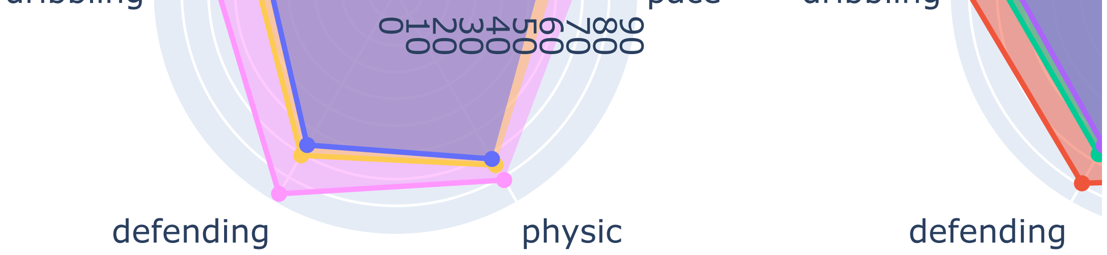

Win
passing
dribbling
defending


Central Midfielder(CM)
passing
shooting
dribbling
pace
0
10
20
30
40
50
60
70
80
defending
physic


Side Mi
passing
dribbling
defending


Centre Back(CB)
passing
shooting
dribbling
pace


Full Back(
passing
dribbling



Two things that stick out:

- Across all positions, the gap between world-class and 75th-percentile is much larger than that of between the 75th-percentile and median.
- Attributes 'pace' and 'physic' do not see much improvement for all positions.

The big 5

Ask anyone with little to no interest or knowledge in football to name one player or a club that they have heard of and chances are, he/it is very likely affilitated to one of the big 5 leagues of Europe - the English Premier League of England, La Liga of Spain, Bundesliga of Germany, Serie A of Italy, and Ligue 1 of France. Well there's other leagues that deserve a shoutout (i.e. Eredivisie) and sometimes it's the underdog clubs that stand at the top but the point is that these leagues attract media, fans, money, and most importantly the greatest figures in the football world.

This time I selected players that were part of the European big five during the 2018-19 season and made some comparisons. Here's the trick - for each club, I only selected the top 15 players and disregarded the rest. This is because a club usually has around 30 or more players and only a portion of them (i.e. the best players) get to take part in games regularly.

In [24]:

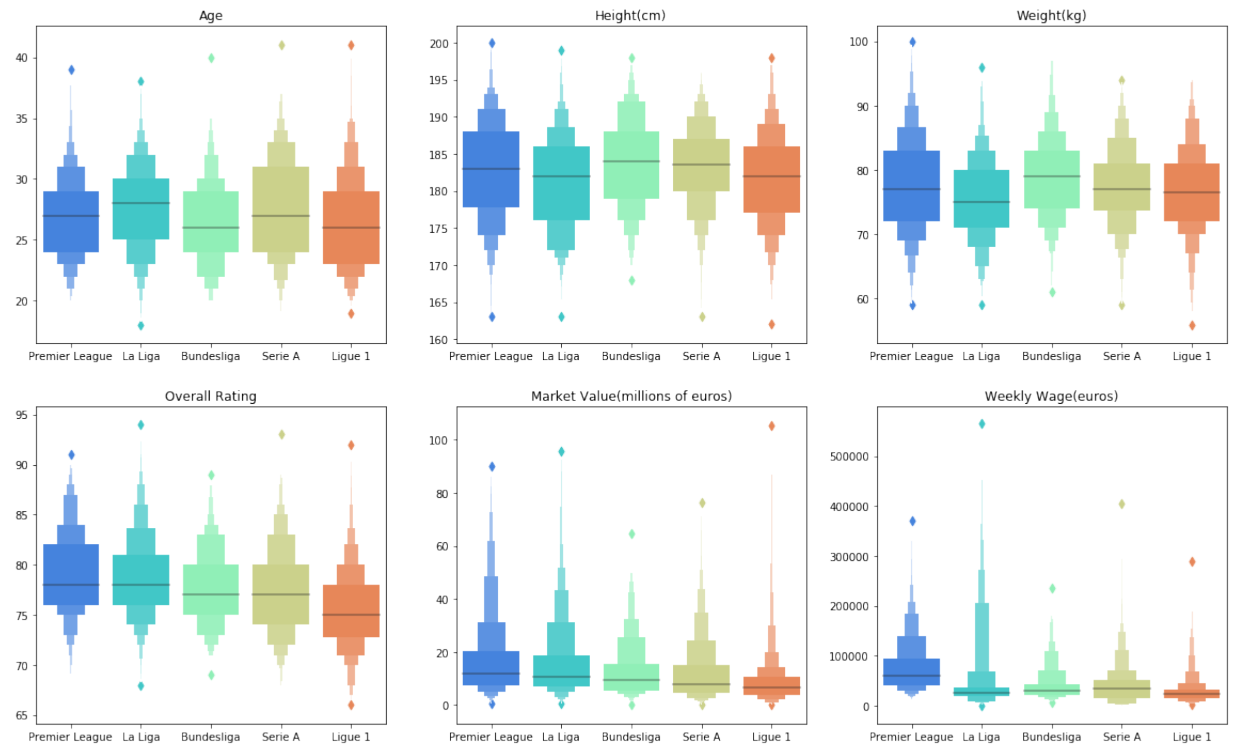
```
epl_teams = ["Manchester City", "Manchester United", "Tottenham Hots
laliga_teams = ["FC Barcelona", "Atlético Madrid", "Real Madrid", "V
bundesliga_teams = ["FC Bayern München", "FC Schalke 04", "TSG 1899
serieA_teams = ["Juventus", "Napoli", "Roma", "Inter", "Lazio", "Mi
ligue1_teams = ["Paris Saint-Germain", "AS Monaco", "Olympique Lyonn
```

In [25]:

```
fifa["value_eur"] = fifa["value_eur"]/1000000
fifa.rename(columns = {"value_eur": "value_eur_mil"}, inplace = True

epl = fifa[fifa["club"].apply(lambda x: True if (x in epl_teams) els
laliga = fifa[fifa["club"].apply(lambda x: True if (x in laliga_tean
bundesliga = fifa[fifa["club"].apply(lambda x: True if (x in bundesl
serieA = fifa[fifa["club"].apply(lambda x: True if (x in serieA_tean
ligue1 = fifa[fifa["club"].apply(lambda x: True if (x in ligue1_tean
```

In [26]:

```
fig = plt.figure(figsize = (20, 12))
cat = ["age", "height_cm", "weight_kg", "overall", "value_eur_mil",
title = ["Age", "Height(cm)", "Weight(kg)", "Overall Rating", "Marke

d = {}
for i in range(6):
    d["ax{0}".format(i)] = fig.add_subplot(2, 3, i+1)
    d["ax{0}".format(i)] = sns.boxenplot(data = [epl[cat[i]], laliga
    d["ax{0}".format(i)].set_title(title[i])
    d["ax{0}".format(i)].set_xticklabels(["Premier League", "La Liga
```



Note: In case you've never seen these types of plots before, these are basically upgraded boxplots. Just like regular boxplots the largest blocks in the middle refer to mid-50% (i.e. 25th percentile ~ 75th percentile), with the black lines indicating median values. Right above them are boxes indicating the next 12.5 percentage worth quantiles (i.e. 75th ~ 87.5th), and this pattern continues (so the next ones above refer to 87.5th ~ 93.75th).

Just a reminder that all the results are based on the 2018-19 season. Going by each,

1) The Premier League, in terms of middle 75% range, has the edge over others when it comes to overall ratings. The Spanish La Liga is a strong competitor though - in fact it actually beats the Premier League at the very top, although by a small margin. One thing

is clear - the French Ligue 1 is at the bottom among the five. Overall the order goes as: EPL, La Liga, Bundesliga, Serie A, Ligue 1.

Interestingly enough, this order matches the UEFA country coefficients ranking for 2018-19: England(22.642), Spain(19.571), Germany(15.214), Italy(12.642), France(10.583). https://www.uefa.com/memberassociations/uefarankings/country/#/yr/2019 (https://www.uefa.com/memberassociations/uefarankings/country/#/yr/2019) Roughly speaking, these country coefficients measure how well did the clubs in their respective nations in European competitions like the Champions League in one particular season.

2) As for age, comparing the middle 75% ranges, Serie A players are the oldest in general. The youngest - Bundesliga.

3) Bundesliga players in general are the tallest. Statistically speaking, among UK, Spain, Germany, France, and Italy, Germany has the highest average height for men, and it seems like that is reflected here. Shortest - La Liga.

4) As tall as they are, the Budesligans are the heaviest. The lightest: La Liga.

5) All in all, Premier League players worth more than any other leaguers. The ranking order is the same as for overall rating - Premier League, La Liga, Bundesliga, Serie A, Ligue 1.

6) Overall, players in the Premier League are paid the highest, but this calls for consideration that they are paid in British pounds, not euros. The greatest wage gap can be seen in the Spanish La Liga, with standard deviation at around 77,334 euros! The least, Ligue 1, at around 29324 euros.

A league of their own

At the top of all football affairs in Europe stands the UEFA Champions League, an annual football competition whose prestige and rigor are second to none. Every year 32 European football clubs that had finished top of their respective leagues vy for the coveted Champions League throphy - and lifting it means more than words for players and clubs, for it signifies reaching the pinnacle of European football, which in turn basically means reaching the very top of the whole world.

Like with any other field the public is mostly interested in some of the best players and clubs. Here I bring you some interesting comparisons concerning the 50 of the most valuable, strongest, and high-paying clubs in the world, based on the 2018-19 season.

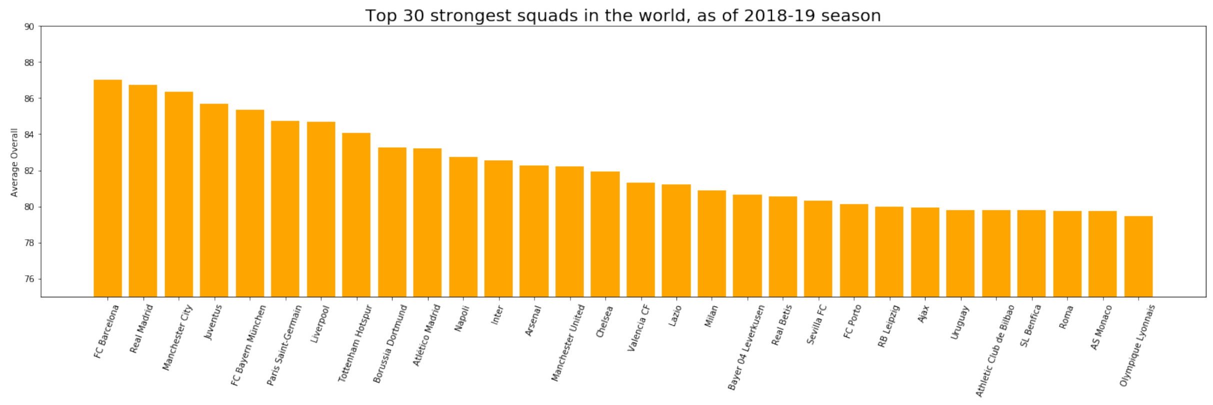
In [27]:

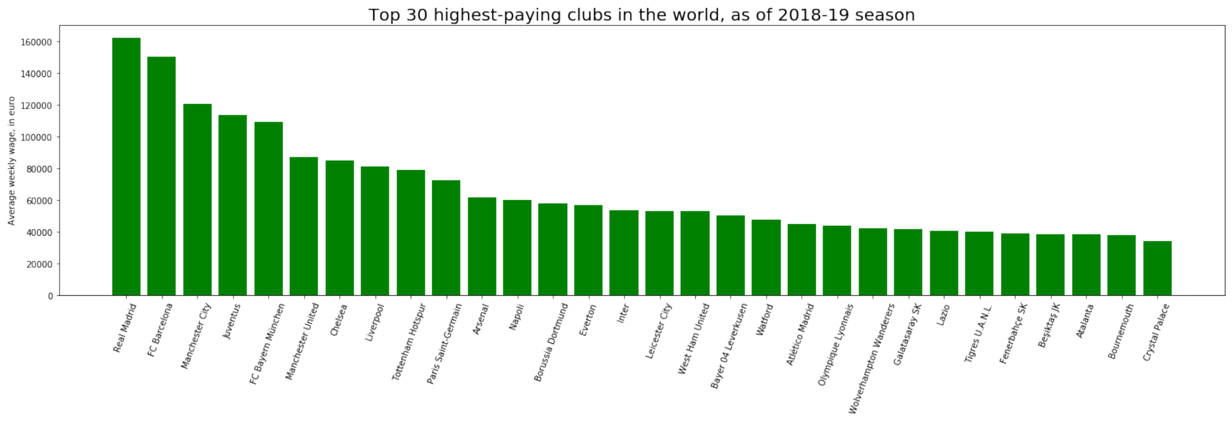
```
t15 = fifa.groupby("club").head(15)
t15a = t15.groupby("club")["overall"].agg(np.mean).sort_values(asce
```

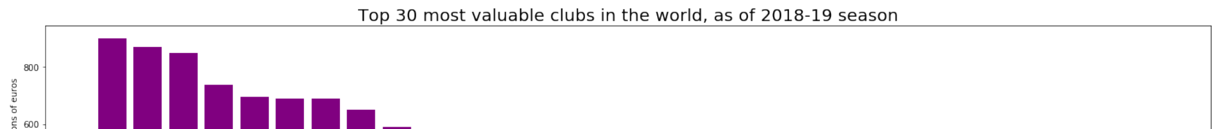
In [28]:

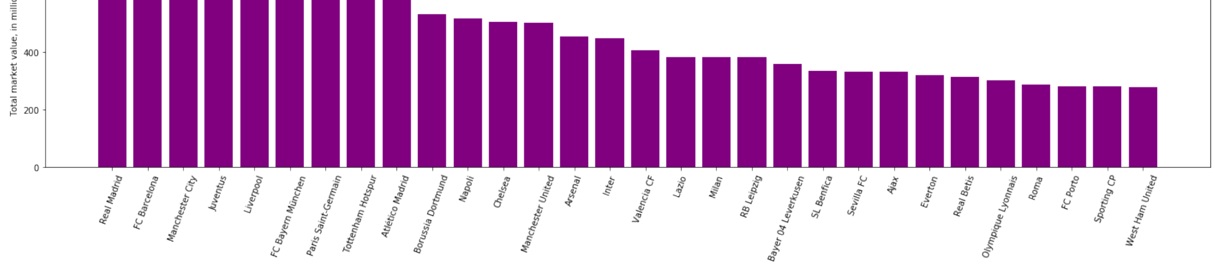
```
fig = plt.figure(figsize = (20, 20))
```

```
ax1 = fig.add_subplot(3, 1, 1)
ax1.bar(t15a.index, t15a.values, color = "orange")
ax1 = plt.title("Top 30 strongest squads in the world, as of 2018-19
ax1 = plt.xticks(rotation = 70)
ax1 = plt.ylabel("Average Overall")
ax1 = plt.axis([None, None, 75, 90])

hpcs = fifa.groupby("club")["wage_eur"].agg(np.mean).sort_values(asc
ax2 = fig.add_subplot(3, 1, 2)
ax2.bar(hpcs.index, hpcs.values, color = "green")
ax2 = plt.title("Top 30 highest-paying clubs in the world, as of 20
ax2 = plt.xticks(rotation = 70)
ax2 = plt.ylabel("Average weekly wage, in euro")

mvcs = fifa.groupby("club")["value_eur_mil"].sum().sort_values(asce
ax3 = fig.add_subplot(3, 1, 3)
ax3.bar(mvcs.index, mvcs.values, color = "purple")
ax3 = plt.title("Top 30 most valuable clubs in the world, as of 201
ax3 = plt.xticks(rotation = 70)
ax3 = plt.ylabel("Total market value, in millions of euros")

plt.tight_layout()
```









Note: The boxplots have been sorted by mean in a decreasing order.

One thing seems clear - money does talk, although it certainly isn't everything.

The world is the stage

It wouldn't be a good use of this dataset if I were to disregard the "nationality" column - so last but not least I did a global scale comparison and see how various football statistics vary across different nations. Perhaps you'll find the below the topmost choropleth map of interesting - as the name suggests it roughly shows which nations have the strongest football teams. Basically for each nation I took the average of top 15 overall ratings - and around 15 is the right number in my opinion, for it guarantees that all nations have the necessary players to form their respective functioning starting XI.

Note: Separate statistics for the four constituents of the UK aren't available in the maps.

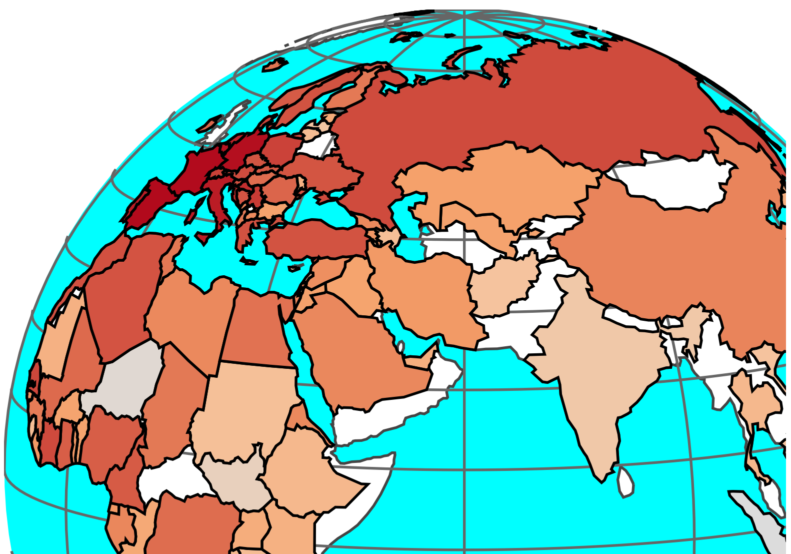
In [29]:

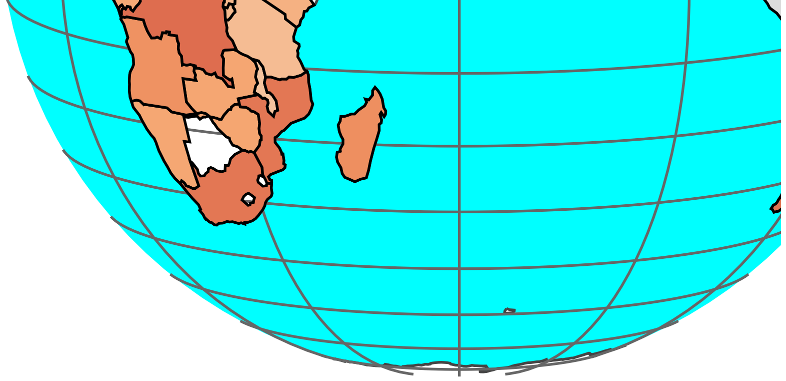
```
import plotly.offline as py
%matplotlib inline

boole = fifa["nationality"].value_counts() >= 15
nations = boole[boole].keys()

def selected(x):
    if x in nations:
        return True
    else:
        return False

qnations = fifa[fifa["nationality"].apply(selected)]
```

In [30]:

```
top15 = fifa.groupby("nationality").head(15)
top15a = top15.groupby("nationality")["overall"].agg(np.mean).sort_

data = [dict(
        type = 'choropleth',
        locations = top15a.index,
        z = top15a,
        locationmode = 'country names',
        text = "Overall",
        marker = dict(
            line = dict(color = 'rgb(0,0,0)', width = 1)),
            colorbar = dict(autotick = True, tickprefix = '',
            title = 'Overall')
            )
        ]
        ]

layout = dict(
    title = 'National team strengths around the world, based on FIFA
    geo = dict(
        showframe = False,
        showocean = True,
        oceancolor = 'rgb(0,255,255)',
        projection = dict(
        type = 'orthographic',
            rotation = dict(
                    lon = 60,
                    lat = 10),
        ),
        lonaxis =  dict(
                showgrid = True,
                gridcolor = 'rgb(102, 102, 102)'
            ),
        lataxis = dict(
                showgrid = True,
                gridcolor = 'rgb(102, 102, 102)'
                )
            ),
        )

fig = dict(data = data, layout = layout)
py.iplot(fig, validate = False)
```

National team strengths around the world, b



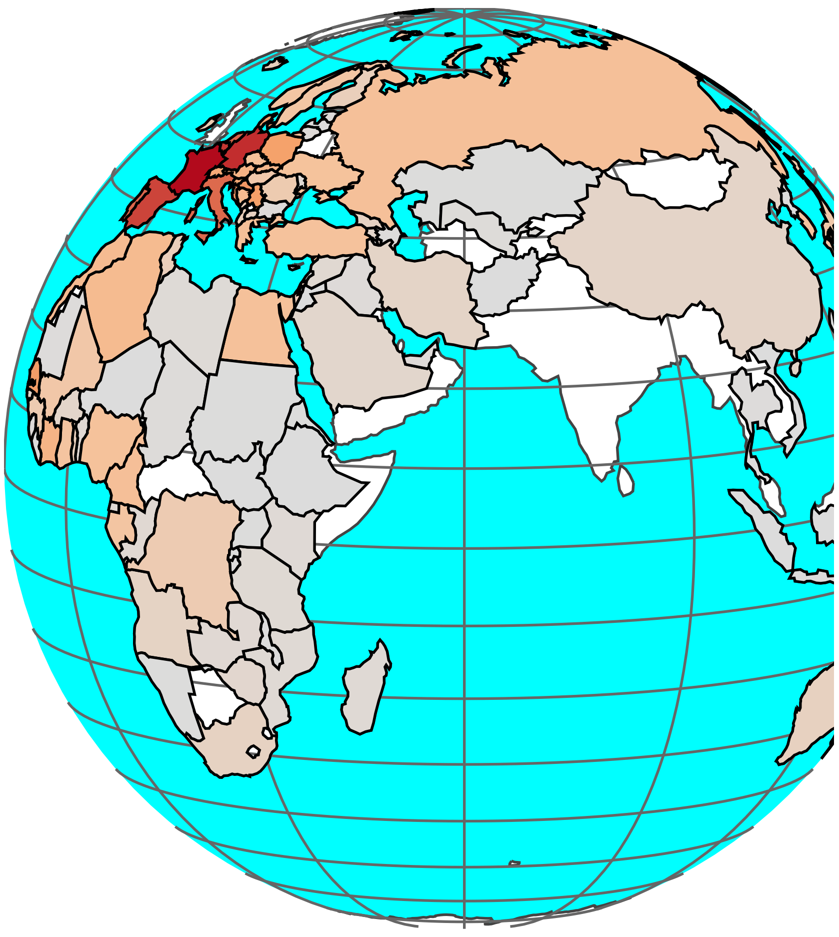


England: 83.47 || Scotland: 76.8 || Wales: 75.67 || Northern Ireland: 71.87

In [31]:

```
top10 = fifa.groupby("nationality").head(10)
top10s = top10.groupby("nationality")["value_eur_mil"].agg(sum)
top10s = top10s[top10s > 0]
countries = top10s.index

data = [dict(
        type = 'choropleth',
        locations = countries,
        z = top10s,
        locationmode = 'country names',
        text = "M €",
        marker = dict(
            line = dict(color = 'rgb(0,0,0)', width = 1)),
            colorbar = dict(autotick = True, tickprefix = '',
            title = 'Euro(Millions)')
            )
       ]


layout = dict(
    title = 'Total market value of top 10 players for each nation',
    geo = dict(
        showframe = False,
        showocean = True,
        oceancolor = 'rgb(0,255,255)',
        projection = dict(
          type = 'orthographic',
              rotation = dict(
                      lon = 60,
                      lat = 10),
          ),
          lonaxis =  dict(
                  showgrid = True,
                  gridcolor = 'rgb(102, 102, 102)'
              ),
          lataxis = dict(
                  showgrid = True,
                  gridcolor = 'rgb(102, 102, 102)'
                  )
              ),
          )

fig = dict(data = data, layout = layout)
py.iplot(fig, validate = False)
```

Total market value of top 10 players for



England: 411M || Scotland: 138.5M || Wales: 128.1M || Northern Ireland: 40.875M

Conclusion

Summary

Below I have gathered some interesting facts and discoveries concerning FIFA 20 and real-world football:

1) The in-game player statistics of the FIFA series are output created by a team of thousands of data gatherers, reviewers, and experts and they blend advanced statistics with opinions and suggestions of game-goers and pundits. Understandably they aren't free of bias but their work speaks and have reputation for that.

2) All the overall ratings can be explained almost entirely with the associated "key" attributes, even though the equations found do not reflect the game maker's approach at all. See the table above for details

3) The average footballer in FIFA 20 is of 25 years of age, stands just above 180cm, weighs around 75kg, and has a overall rating of 66-ish. He is right-footed, has a weak foot rating of 3, and skill moves rating of 2.

4) "shooting"&"dribbling" are very closely related to each other - "passing"&"dribbling" too. The goalkeeper attributes are very closely related to each other and many of them are almost directly related to overall rating.

5) In FIFA 20, the gap between the very top players and 75th percentile is much greater than that of between 75th percentile and median-ish players, at least in terms of numbers. Attributes "pace" and "physic" do not vary much though.

6) Based on the 2018-19 season, EPL has the edge over all the other major European leagues when it comes to overall ratings. The weakest among the major five is Ligue 1. At the very pinnacle, however, the Spanish La Liga has the edge, and that is the case for market value and weekly wage. Overall the Bundesligans are the youngest and the tallest. The greatest weekly wage gap is seen in La Liga - the least, Ligue 1. For more refer to above.

7) Based on the 2018-19 season, the top 10 world's most valuable clubs were, in the following order: Real Madrid, Barcelona, Manchester City, Juventus, Liverpool, Bayern Munich, Paris Saint-Germain, Tottenham, Atletico Madrid, Dortmund.

The top 10 highest-paying clubs were the following: Real Madrid, Barcelona, Manchester City, Juventus, Bayern Munich, Manchester United, Chelsea, Liverpool, Tottenham, Paris Saint-Germain.

Lastly, the top 10 strongest clubs (based solely on average overall rating of top 15 players for each) in the same season have been the following: Barcelona, Real Madrid, Manchester City, Juventus, Bayern Munich, Paris Saint-Germain, Liverpool, Tottenham, Dortmund, Atletico Madrid.

8) The top 5 strongest national teams come from Spain, Brazil, France, Germany, and Belgium. The top 5 most valuable national teams come from France, Brazil, Germany, Belgium, and Argentina.

Moving Forward

Although I did not quite follow my initial curiosity-driving question I believe I still ended up with some interesting and meaningful insights, most of which requires some wrangling with this particular type of dataset. I think I am most satistied with the linear models I found in the beginning, as they are incredibly well-performing and offer something that cannot be found anywhere else. It's been a fruitful project overall and I had fun exploring it.

There's room for improvement and I would be happy to develop even further in the future, maybe for FIFA 21. A wonderful dashboard and maybe some other advanced ML algorithms could bring the project to another level...

The biggest challenge for me was figuring out what to ask and how to go about approaching what I'd like to know. With limited knowledge I was forced to search the web many times and educate myself with some knowledge concering the dataset and football in general. What's more, football is such a complex sport and there's so many variables that come into play, and this makes coming up with "meaningful" and "accurate" insights and findings to be rather demanding. Now I definitely understand why "domain knowledge" and "business insights" are deemed critical in data science...

Anyways, this has been an amateur football fan's journey of storytelling football with data, and if you've been reading the whole thing, thank you very much! I hope it had something interesting to offer.

Football Analytics - Analysing the FIFA 20 Player Ratings Dataset

Introduction

Football - or more familiarly known as "soccer" to some - is arguably the most popular sport in the world. It is a worldwide entertainment beloved by many; there's a reason why most nations have their own professional football league system, and why there's lots of national, continental, and sometimes global level football competitions every often. What else sports can match this level of popularity?

It shouldn't be surprising then that video games featuring real-world clubs and players are also very popular. One of them is the FIFA series, a football simulation video game series developed by a major video game company Electronic Arts. It's a series in that the creators release a follow-up version every year - the latest version as of now is FIFA 20.

As a rough introduction, each player in the game is assigned an overall player rating out of 100 based on his performance in the preceding year. Of course these figures are updated annually in accordance with the latest real-world data. In FIFA 20, the best player in the game - Lionel Messi - was given an overall rating of 94, with detailed specs of the following: 88 pace, 96 dribbling, 92 shooting, 39 defence, 92 passing, and 66 physical.

If you come to think about it, that's quite detailed and precise - especially considering how attributes like such are difficult to be accurately quantified. Perhaps not surprisingly, the figures often spark debates and controversies among football fans.

I wondered - how much are they relevant to the reality? How are the numbers assigned in the first place? Then again an idea struck my head - that this combined with other ideas could develop into an exciting DS project. And that's how it began for me...

With all these said, the aim of this project is to:

- Delve into questions like above using statistical methods and tools
- Extract lots of fun football facts through EDA
- Create lots of different visualisations because it's fun and images always help
- Have fun!

The Dataset

Thankfully, there's a dataset for FIFA 20, available on Kaggle, found here:

https://www.kaggle.com/stefanoleone992/notebook?select=players_20.csv
(https://www.kaggle.com/stefanoleone992/notebook?select=players_20.csv)

As a side note, all the data in the dataset are based on the 2018-19 season.

This dataset consists of whopping 104 columns - description for each column can be found in the link above. Number of observations: 18278.

In [1]:

```
```

Toolsets

- Language: Python
- Libraries: NumPy, pandas, Matplotlib, Seaborn, Plotly
- Development Environment: Jupyter Notebook

Data Preprocessing

As with every other data-driven project, data preprocessing (i.e. data cleaning) must be preceded before any analysis. Clearly not all 104 columns will be in use, and therefore the first step is to drop unnecessary columns.

After some thought I decided to keep the following columns and drop the rest: long_name, age, height_cm, weight_kg, nationality, club, overall, value_eur, wage_eur, player_positions, preferred_foot, weak_foot, skill_moves, work_rate, pace, shooting, passing, dribbling, defending, physic, gk_diving, gk_handling, gk_kicking, gk_reflexes, gk_speed, gk_positioning

The rest have been excluded due to one or more of the following reasons:

- the feature is completely meaningless/adds literally nothing
- the feature could be useful but I simply wasn't interested at all
- the feature could be useful but dropped it for simplicity's sake

```
In [2]:
```

```
Out[2]:
```

	long_name	age	height_cm	weight_kg	nationality	club	overall
0	Lionel Andrés Messi Cuccittini	32	170	72	Argentina	FC Barcelona	94
1	Cristiano Ronaldo dos Santos Aveiro	34	187	83	Portugal	Juventus	93
2	Neymar da Silva Santos Junior	27	175	68	Brazil	Paris Saint-Germain	92
3	Jan Oblak	26	188	87	Slovenia	Atlético Madrid	91
4	Eden Hazard	28	175	74	Belgium	Real Madrid	91
5	Kevin De Bruyne	28	181	70	Belgium	Manchester City	91
6	Marc-André ter Stegen	27	187	85	Germany	FC Barcelona	90
7	Virgil van Dijk	27	193	92	Netherlands	Liverpool	90
8	Luka Modrić	33	172	66	Croatia	Real Madrid	90
9	Mohamed Salah Ghaly	27	175	71	Egypt	Liverpool	90

10 rows × 26 columns

The column names are quite self-explanatory, but just to be clear with some of them,

- overall: player's overall rating, out of 100 (base rating, no upgrades or whatever obviously)
- value_eur: player's market value, in euros
- wage_eur: player's weekly wage, in euros
- weak_foot: player's weak foot (i.e. non-preferred foot) rating, 1-5
- skill_moves: player's skill moves rating, 1-5
- work_rate: how a player puts effort to participate in attacks and defences (i.e. (Low/High) means low effort in attack and high effort in defence)
- pace, dribbling, ..., gk_positioning: specific attributes out of 100

The next step is to ensure that no player's data have been duplicated.

In [3]:

```
```

Out[3]:

```
0
```

note: sum(fifa.duplicated("long_name")) returns 60, but this isn't problematic since two different footballers can have the exact same long name (yes, this is possible, i.e. "Ben Davies" at Tottenham Hotspur & "Ben Davies" at Preston North End)

Then for each feature, ensure that there's no ridiculous value:

In [4]:

```

```

Out[4]:

```
array([32, 34, 27, 26, 28, 33, 20, 25, 31, 30, 24, 29,
23, 19, 22, 37, 36,
       21, 41, 38, 35, 18, 40, 39, 17, 16, 42])
```

In [5]:

```
```

Out[5]:

```
array([170, 187, 175, 188, 181, 193, 172, 178, 191, 19
2, 168, 173, 184,
       182, 189, 176, 177, 199, 194, 185, 183, 180, 17
9, 169, 186, 174,
       163, 165, 195, 190, 196, 171, 197, 167, 198, 16
6, 164, 200, 161,
       158, 201, 203, 162, 157, 160, 159, 156, 202, 20
5])
```

In [6]:

Out[6]:

```
array([ 72,  83,  68,  87,  74,  70,  85,  92,  66,  7
1,  73,  89,  91,
        82,  86,  80,  76,  75,  84,  69,  96,  67,  7
7,  78,  64,  81,
        61,  94,  59,  60,  90,  79,  97,  65,  95,  9
3,  63,  62,  88,
       100,  98,  58, 101, 103,  56,  99,  57, 102,  5
5,  52,  54, 104,
       107,  53, 110,  50])
```

Inspection on all the other numerical columns confirms that all values are sensible. Next up - missing values:

In [7]:

Out[7]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x116257748>
```

All the necessary data values are present. The features "pace", "shooting", "passing", "dribbling", "defending", and "physic" are in complete contrast with the goalkeeper-associated features, which is exactly what I want to see (i.e. exactly how it is in the game). No further action is required for missing values and this dataset is good to go.

The Analysis

Now for the first investigation - how well do the game statistics reflect reality? To be specific, how well, in general, do the player attribute numbers evaluate performances of players in the real world?

But what probably should come off first is, how are the numbers measured in the first place? Incidentally this was once answered in an interview by ESPN FC with Michael Mueller-Moehring, overseer of player rating calculations, back in 2016: https://www.espn.com/soccer/blog/espn-fc-united/68/post/2959703/fifa-17-player-ratings-system-blends-advanced-stats-and-subjective-scouting (https://www.espn.com/soccer/blog/espn-fc-united/68/post/2959703/fifa-17-player-ratings-system-blends-advanced-stats-and-subjective-scouting)

In short, the player ratings system takes into account two main sources of data: 1) advanced stats, such as number of goals scored/successful passes/saves, etc. 2) subjective data from a team of thousand data editors, field experts, and in-person game-goers.

Despite their best efforts, however, the final output isn't totally free from misevaluation, and this is understandable - after all, football is a complex sports where so many variables come into play. As Mueller-Moehring rightly points out, "the stats are, in most cases, not taking into account very specific circumstances" - data simply can't tell everything.

Conclusion: they have a structured and rigid approach to their work, which renders the output reliable to some extent, yet inevitably subjectivity is still there, so do not take it seriously.

With all these said I judged that it is pointless to try gauaging how "valid" the numbers are, though this was precisely what drove me in the beginning.

- Evaluating how "accurate" something requires a "test set"(i.e. the "correct/truthful" values) and in this case that "test set" is unavailable. For instance, what advanced real statistics and what combination of them can be used to accurately tell how "good" a midfielder is? What's more, football is an extremely complex sport - data alone aren't enough.
- The ratings team, some of which are experts, have already been incorporating advanced stats. This further renders what I'm trying to accomplish meaningless.

Thus I decided to take a slight detour and delve into the following question instead:

Question: How are the in-game specific attribute ratings related to overall rating?

Specifically, for each playing position, what specific attributes hold greater weight when calculating overall rating? Or to put it more regression-friendly, what would a model to explain one's overall rating in terms of other specific ratings look like?

There are many different types of positions in football. After a thorough research I decided to partition all the unique playing positions into the following.

- Forward: ST, CF
- Winger: LW, RW
- Attacking Midfielder: CAM
- Central Midfielder: CM
- Side Midfielder: LM, RM
- Defensive Midfielder: CDM
- Centre Back: CB
- Full Back: LWB, RWB, LB, RB
- Goalkeeper: GK

Note: I initially thought of separating strikers from centre forwards, but the lack of "main" centre forwards in this dataset and the similarity between the two roles eventually led me to group them together.

Now a player can play different positions. It seems like, for each player, player_positions tend to list positions in an order of decreasing familiarity, so for those with multiple playing positions I chose the first one.

In [8]:

```

```

For forward (ST/CF):

In [9]:

```
```

Out[9]:

	long_name	overall	weak_foot	skill_moves	work_rate	pace	shooting	passing	d
1	Cristiano Ronaldo dos Santos Aveiro	93	4	5	High/Low	90.0	93.0	82.0	
10	Kylian Mbappé	89	4	5	High/Low	96.0	84.0	78.0	
12	Harry Kane	89	4	3	High/High	70.0	91.0	79.0	
17	Sergio Leonel Agüero del Castillo	89	4	4	High/Medium	80.0	90.0	77.0	
19	Luis Alberto Suárez Díaz	89	4	3	High/Medium	73.0	89.0	80.0	
[illegible]	Robert	89	4	4	High/Medium	77.0	87.0	74.0	

Because work_rate is a categorical variable, it has to be replaced with dummy variables.

In [10]:

```
```

Of the 2000 sample players chosen, 1500 were randomly chosen to feed the model. The remaining 500 will be used as the test set.

In [11]:

```
```

Now for fitting a regression model. For variable selection I chose backward elimination.

In [12]:

```
['const', 'skill_moves', 'pace', 'shooting', 'passing'
, 'dribbling', 'physic']
```

```
//anaconda3/lib/python3.7/site-packages/numpy/core/fro
mnumeric.py:2389: FutureWarning: Method .ptp is deprec
ated and will be removed in a future version. Use nump
y.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Backward elimination has chosen variables above. In the end I chose to drop 'defending' and 'Medium/Low' as well because they hardly made any difference (in terms of BIC). Thus the final model:

In [13]:

Out[13]:

OLS Regression Results

Dep. Variable:	overall	**R-squared:**	0.957
Model:	OLS	**Adj. R-squared:**	0.956
Method:	Least Squares	**F-statistic:**	1.028e+04
Date:	Mon, 13 Jul 2020	**Prob (F-statistic):**	0.00
Time:	16:07:15	**Log-Likelihood:**	-3125.9
No. Observations:	1875	**AIC:**	6262.

```
    Df Residuals:          1870              BIC:      6289.
        Df Model:             4
 Covariance Type:     nonrobust

             coef  std err       t   P>|t|  [0.025  0.975]
    const  0.8473    0.368   2.300   0.022   0.125   1.570
     pace  0.0275    0.003   7.906   0.000   0.021   0.034
 shooting  0.5914    0.008  73.368   0.000   0.576   0.607
dribbling  0.2655    0.008  33.627   0.000   0.250   0.281
   physic  0.1255    0.004  32.613   0.000   0.118   0.133

      Omnibus:  47.997     Durbin-Watson:     2.081
Prob(Omnibus):   0.000  Jarque-Bera (JB):    74.327
         Skew:   0.243          Prob(JB):  7.25e-17
     Kurtosis:   3.846          Cond. No.  1.67e+03
```

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.67e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

Prob(Omnibus) indicates that the residuals do not follow Normal distribution - which basically means errors are not normally distributed. This, however, poses no problem especially since sample size is large. (If that wasn't helpful at all read here: https://thestatsgeek.com/2013/08/07/assumptions-for-linear-regression/ (https://thestatsgeek.com/2013/08/07/assumptions-for-linear-regression/))

Multicollinearity (i.e. two or more explanatory variables related to each other) seems to be a thing, as evinced in the large condition number. In general severe multicollinearity undermines regression coefficient estimates by inflating their variances (i.e. here std err). But even so in this case the standard errors remain relatively small, with all of them being less than 0.01. In other words, regression coefficients vary little from one trial to another (in fact the confidence intervals are telling). With this considered I concluded that multicollinearity does not, for the most part, temper with interpreting regression coefficients.

The adj. R-squared value (i.e. goodness-of-fit) is very high, at about 0.955. How will this model perform on the test set though?

In [14]:

```
```

Out[14]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jo
bs=None, normalize=False)
```

In [15]:

```
```

Out[15]:

```
0.9639036936229834
```

Seems like the final model works fantastic on the test set. So there we have it:

Expected overall rating for a forward (i.e. main position is either ST or CF) = 0.8473 + 0.0275 * pace + 0.5914 * shooting + 0.2655 * dribbling + 0.1255 * physic

Test set score for this model: 0.96

Train/Test size: 1875/625

I applied the same procedure for the remaining eight playing position categories. The results are shown below:

```
In [16]:
```

```
In [17]:
```

```
Out[17]:
```

	final model	test set score (approx)	train/test size	top 3 attributes
position category				
Forward(ST/CF)	E(Y) = 0.8473 + 0.0275*pace + 0.5914*shooting + 0.2655*dribbling + 0.1255*physic	0.960	1875/625	shooting, dribbling, physic
Winger(LW/RW)	E(Y) = -3.039 + 0.1301*pace + 0.1965*shooting + 0.2676*passing + 0.4199*dribbling + 0.0339*physic	0.985	525/175	dribbling, passing, shooting
Attacking Midfielder(CAM)	E(Y) = 1.4226 + 0.0458*pace + 0.1848*shooting + 0.3425*passing + 0.3914*dribbling + 0.0383*physic	0.970	750/250	dribbling, passing, shooting
Central Midfielder(CM)	E(Y) = 1.019 + 0.0622*shooting + 0.3936*passing + 0.3356*dribbling + 0.1658*defending + 0.0684*physic	0.960	1500/500	passing, dribbling, defending
Side Midfielder(LM/RM)	E(Y) = -0.5254 + 0.1233*pace + 0.0839*shooting + 0.3808*passing + 0.3701*dribbling + 0.0656*physic	0.970	1500/500	passing, dribbling, pace

Defensive Midfielder(CDM)	E(Y) = 1.5289 + 0.2279*passing + 0.1272*dribbling + 0.5359*defending + 0.1306*physic	0.975	1050/350	defending, passing, physic
Centre Back(CB)	E(Y) = 1.4127 + 0.0390*passing + 0.0452*dribbling + 0.7300*defending + 0.1852*physic	0.980	2250/750	defending, physic, dribbling
Full Back(LWB, RWB, LB, RB)	E(Y) = 0.5984 + 0.1364*pace + 0.1502*passing + 0.1090*dribbling + 0.5709*defending + 0.0593*physic	0.970	1950/650	defending, passing, pace
Goalkeeper(GK)	E(Y) = -0.9113 + 0.2430*gk_diving + 0.2154*gk_handling + 0.0543*gk_kicking + 0.2497*gk_reflexes + 0.2570*gk_positioning	0.990	1500/500	gk_positioning, gk_reflexes, gk_diving

And there you have it. Some side notes:

1) As a reminder, for players with multiple playing positions I only considered their "main one" (i.e. the position that appears on their card)

2) The explanatory variables I considered are the following: weak_foot, skill_moves, work_rate, pace, shooting, passing, dribbling, defending, physic, gk_diving, gk_handling, gk_kicking, gk_reflexes, gk_speed, gk_positioning. Note that none of the first three variables were included in any of the final models.

3) Although "top 3 attributes" lists three of the most contributive attributes in the order of decreasing significance, the order isn't always meaningful. This is especially the case since multicollinearity is present (that being said interpreting regression coefficients is valid for the most part). It's just that, for instance, we can't say "gk_positioning" is more important than "gk_reflexes" when the two coefficients differ by less than 0.01.

Empirical data suggests that, roughly speaking, if two regression coefficients differ by less than 0.04, there's no significant evidence that the seemingly larger one truly holds greater weight than the other one (i.e. comparison is unmeaningful).

4) The models only explain how, for each main playing position, overall rating is associated with the main 12 specific attributes. This is NOT the way the game makers calculate overall ratings.

Try them by yourself and see how they work!

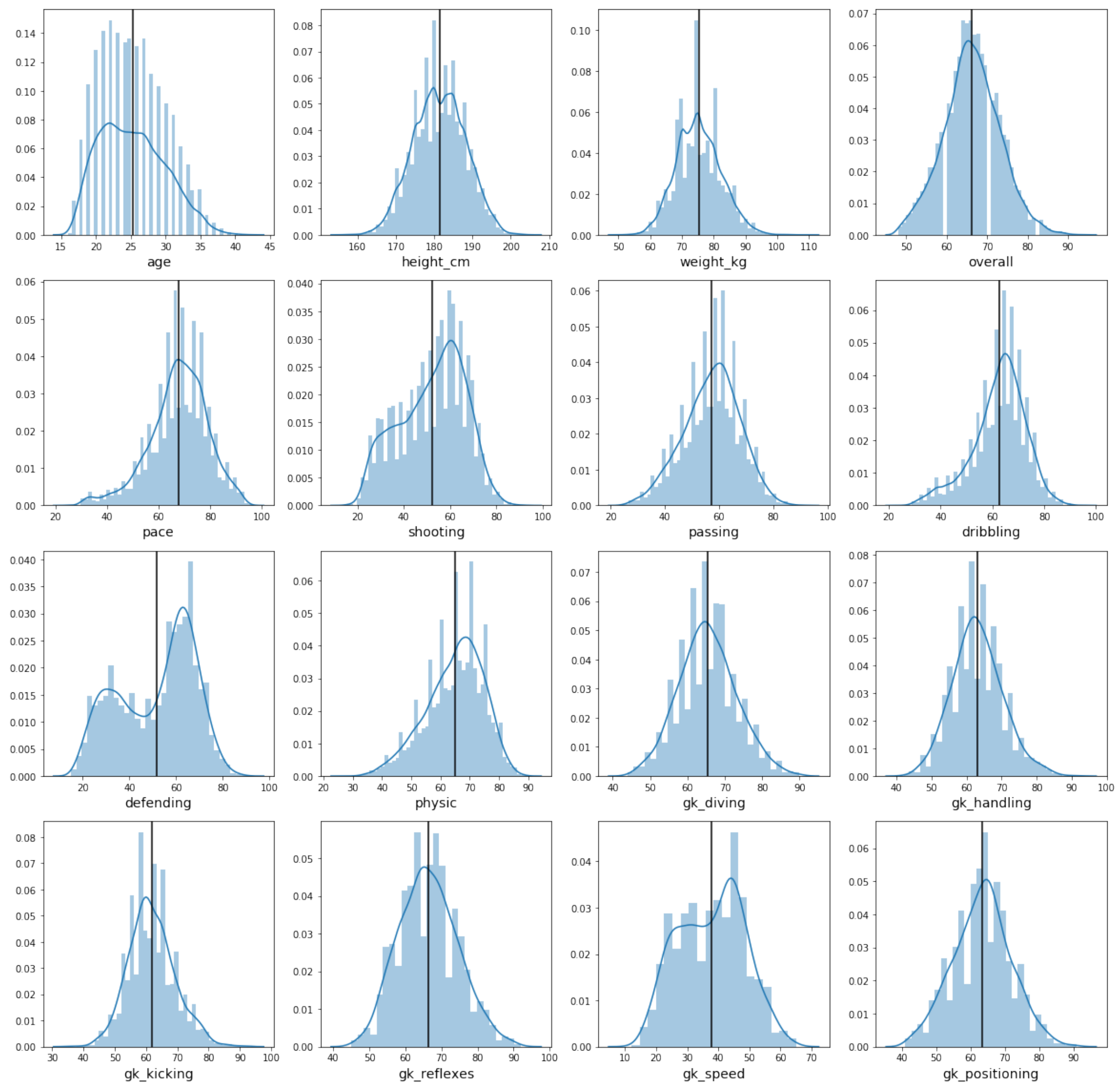
Next - exploratory data analysis.

Basic distributional information

Starting with the basics. Below are distribution for many of the feature columns I chose to explore.

In [18]:

```

```



Many of them are more or less normally distributed, though some are slightly skewed. It's interesting how all (except for gk_speed) goalkeeper attributes are normally distributed while the other six counterparts (pace, shooting, ..., physic) are not.

Note:

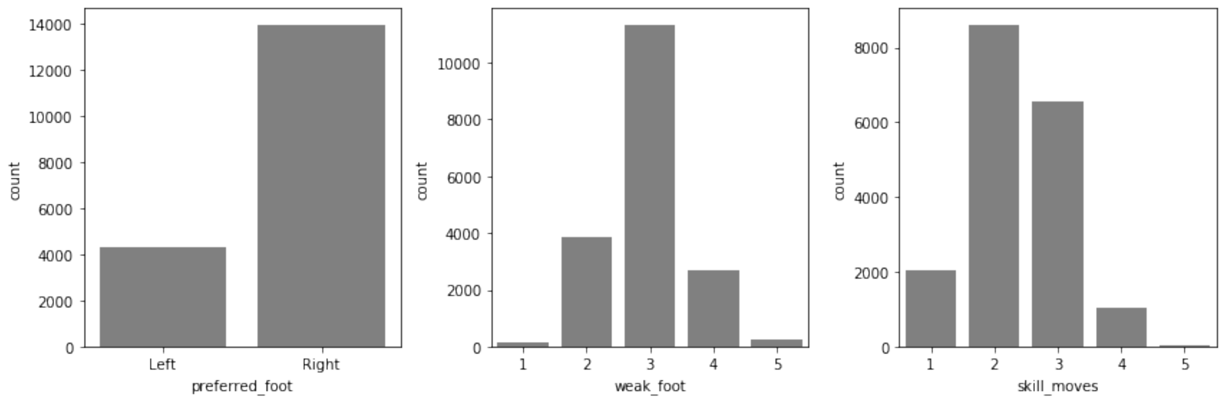
1) The black vertical lines indicate their respective mean values.

2) The y-values are to the y-values probability density plots.

Bar graphs:

In [19]:

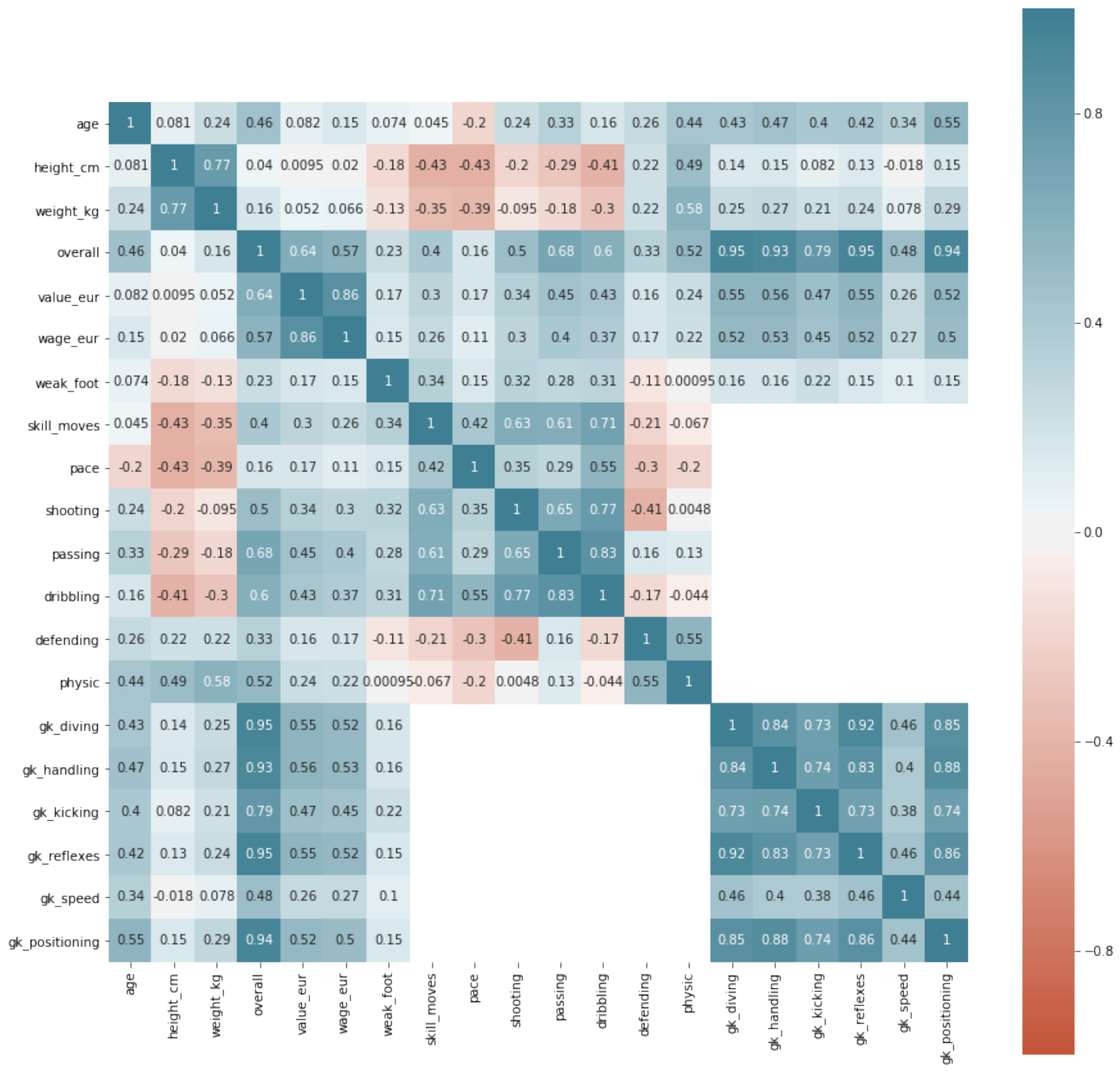
```
```



The correlation matrix plot

What two features are related to each other? The correlation matrix plot below has all the answers...

In [20]:


age
height_cm
weight_kg
overall
value_eur
wage_eur
weak_foot
skill_moves
pace
shooting
passing
dribbling
defending
physic
gk_diving
gk_handling
gk_kicking
gk_reflexes
gk_speed
gk_positioning
0.8
0.4
0.0
−0.4
−0.8

The numbers inside the square boxes are Pearson correlation coefficients (i.e. a value close to -1 indicates strong negative linear relationship between two variables and a value close to 1 strong positive linear relationship).

Note that many of the goalkeeper attributes are highly correlated with 'overall'. This backs the result found earlier, where it was concluded that the most important features for explaining player overall rating are: gk_diving, gk_handling, gk_reflexes, and gk_positioning. Here all of these have $r > 0.9$.

Strong ($r > 0.7$) correlation is seen in some of the other pairs, but many of them are not surprising. For instance, 'wage_eur' and 'value_eur' are strongly correlated, and this is not surprising at all.

Aside from these two pairs strike my attention - 'passing'&'dribbling' and 'shooting'&'dribbling'. The r values for these are 0.83 and 0.77, respectively. For some reason 'dribbling' has lots to do with 'passing'&'shooting', more so than 'pace' ($r = 0.55$), which is quite interesting and personally for me, counter-intuitive.

Perhaps the results table earlier may shed some light to this. According to the table, 'passing'&'dribbling' are more or less equally important for midfielders (CAM, CM, LM, RM), and these players make up a bulk of the original dataset. For 'pace'&'dribbling', more less so - in fact, none of the final models seem to treat the two equally.

The road to world-class

Becoming a top-of-the-world footballer is tough. To be one you've got to have skills and talents that set you apart big time. But just how much of different level are they?

For each of the main positions I defined above, I picked the top 20 players in the world in terms of overall rating (if 20 sounds too much, that's just over the number of starting goalkeepers that make it into the top 16 of the UEFA Champions League in one season) and compared them to their respective 75th percentile-ish and median-ish level players (so groups of 20 for each level). The results are below:

```
In [21]:
```

```
In [22]:
```

```
In [23]:
```

Two things that stick out:

- Across all positions, the gap between world-class and 75th-percentile is much larger than that of between the 75th-percentile and median.
- Attributes 'pace' and 'physic' do not see much improvement for all positions.

The big 5

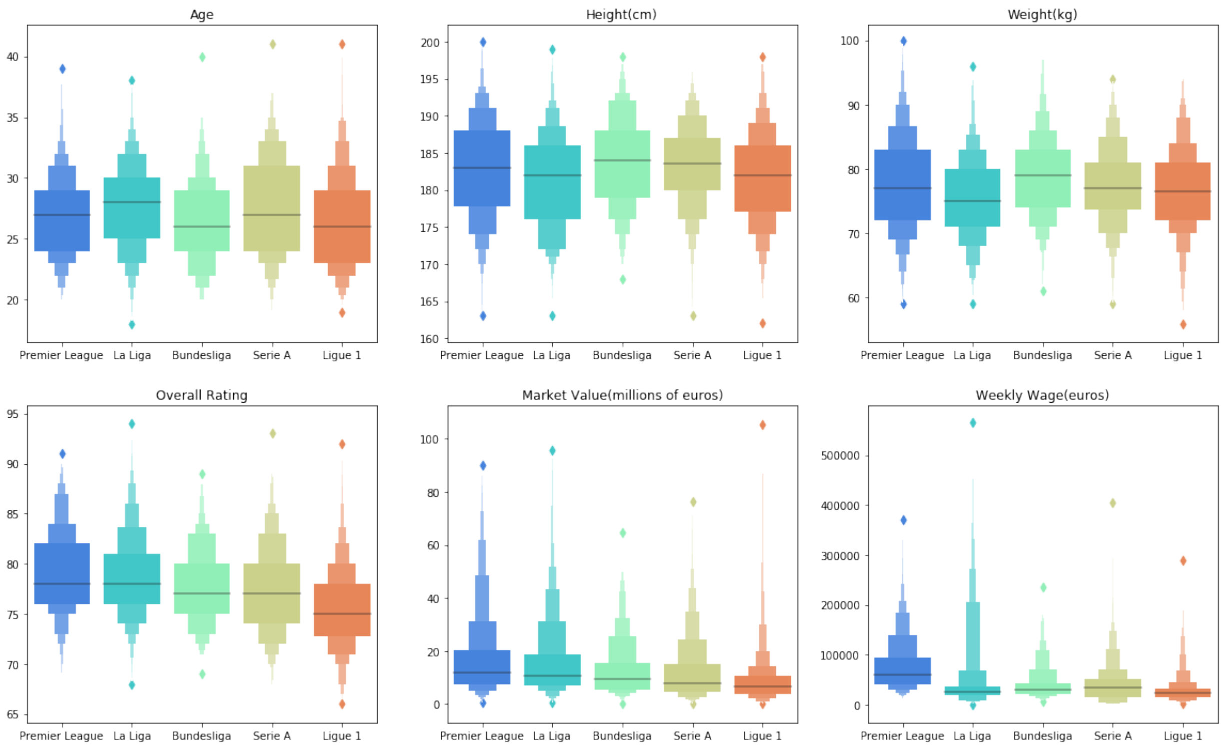
Ask anyone with little to no interest or knowledge in football to name one player or a club that they have heard of and chances are, he/it is very likely affilitated to one of the big 5 leagues of Europe - the English Premier League of England, La Liga of Spain, Bundesliga of Germany, Serie A of Italy, and Ligue 1 of France. Well there's other leagues that deserve a shoutout (i.e. Eredivisie) and sometimes it's the underdog clubs that stand at the top but the point is that these leagues attract media, fans, money, and most importantly the greatest figures in the football world.

This time I selected players that were part of the European big five during the 2018-19 season and made some comparisons. Here's the trick - for each club, I only selected the top 15 players and disregarded the rest. This is because a club usually has around 30 or more players and only a portion of them (i.e. the best players) get to take part in games regularly.

In [24]:

In [25]:

In [26]:



Note: In case you've never seen these types of plots before, these are basically upgraded boxplots. Just like regular boxplots the largest blocks in the middle refer to mid-50% (i.e. 25th percentile ~ 75th percentile), with the black lines indicating median values. Right above them are boxes indicating the next 12.5 percentage worth quantiles (i.e. 75th ~ 87.5th), and this pattern continues (so the next ones above refer to 87.5th ~ 93.75th).

Just a reminder that all the results are based on the 2018-19 season. Going by each,

1) The Premier League, in terms of middle 75% range, has the edge over others when it comes to overall ratings. The Spanish La Liga is a strong competitor though - in fact it actually beats the Premier League at the very top, although by a small margin. One thing is clear - the French Ligue 1 is at the bottom among the five. Overall the order goes as: EPL, La Liga, Bundesliga, Serie A, Ligue 1.

Interestingly enough, this order matches the UEFA country coefficients ranking for 2018-19: England(22.642), Spain(19.571), Germany(15.214), Italy(12.642), France(10.583). https://www.uefa.com/memberassociations/uefarankings/country/#/yr/2019 (https://www.uefa.com/memberassociations/uefarankings/country/#/yr/2019) Roughly speaking, these country coefficients measure how well did the clubs in their respective nations in European competitions like the Champions League in one particular season.

2) As for age, comparing the middle 75% ranges, Serie A players are the oldest in general. The youngest - Bundesliga.

3) Bundesliga players in general are the tallest. Statistically speaking, among UK, Spain, Germany, France, and Italy, Germany has the highest average height for men, and it seems like that is reflected here. Shortest - La Liga.

4) As tall as they are, the Budesligans are the heaviest. The lightest: La Liga.

5) All in all, Premier League players worth more than any other leaguers. The ranking order is the same as for overall rating - Premier League, La Liga, Bundesliga, Serie A, Ligue 1.

6) Overall, players in the Premier League are paid the highest, but this calls for consideration that they are paid in British pounds, not euros. The greatest wage gap can be seen in the Spanish La Liga, with standard deviation at around 77,334 euros! The least, Ligue 1, at around 29324 euros.

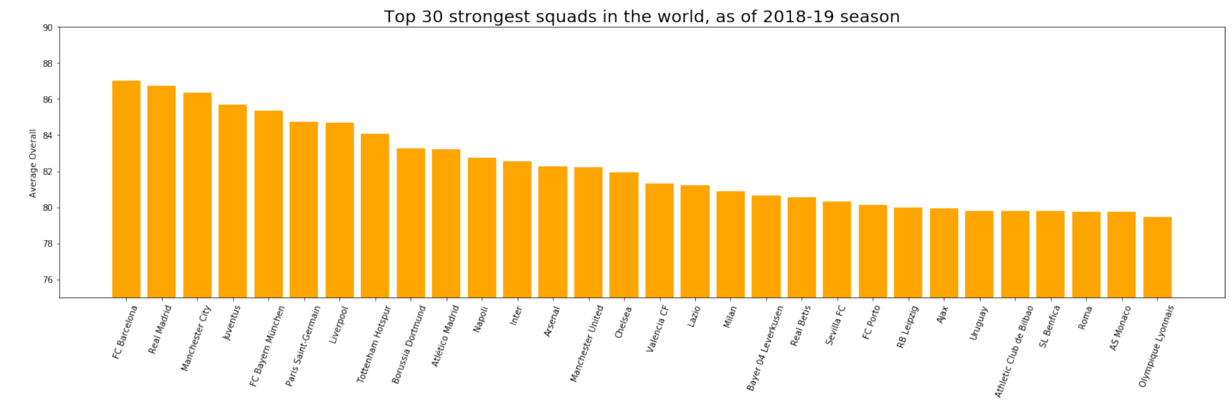
A league of their own

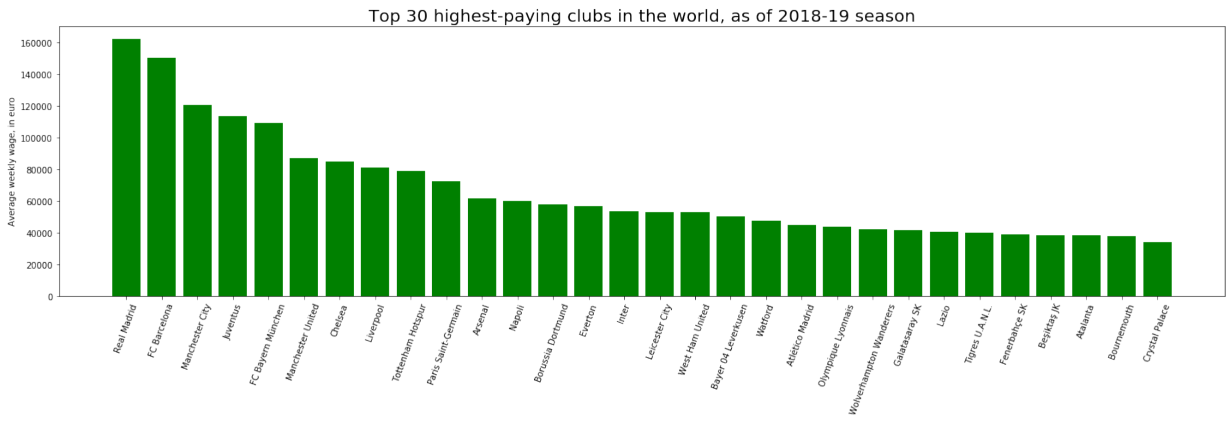
At the top of all football affairs in Europe stands the UEFA Champions League, an annual football competition whose prestige and rigor are second to none. Every year 32 European football clubs that had finished top of their respective leagues vy for the coveted Champions League throphy - and lifting it means more than words for players and clubs, for it signifies reaching the pinnacle of European football, which in turn basically means reaching the very top of the whole world.

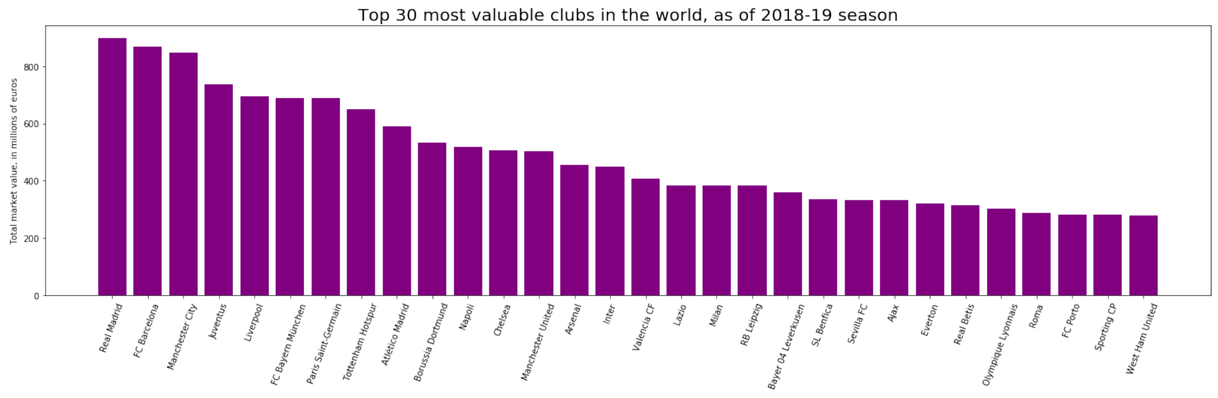
Like with any other field the public is mostly interested in some of the best players and clubs. Here I bring you some interesting comparisons concerning the 50 of the most valuable, strongest, and high-paying clubs in the world, based on the 2018-19 season.

```
In [27]:
```

In [28]:







Note: The boxplots have been sorted by mean in a decreasing order.

One thing seems clear - money does talk, although it certainly isn't everything.

The world is the stage

It wouldn't be a good use of this dataset if I were to disregard the "nationality" column - so last but not least I did a global scale comparison and see how various football statistics vary across different nations. Perhaps you'll find the below the topmost choropleth map of interesting - as the name suggests it roughly shows which nations have the strongest football teams. Basically for each nation I took the average of top 15 overall ratings - and around 15 is the right number in my opinion, for it guarantees that all nations have the necessary players to form their respective functioning starting XI.

Note: Separate statistics for the four constituents of the UK aren't available in the maps.

```
In [29]:
```

```
In [30]:
```

```
England: 83.47 || Scotland: 76.8 || Wales: 75.67 || Northern Ireland: 71.87
```

```
In [31]:
```

```
England: 411M || Scotland: 138.5M || Wales: 128.1M || Northern Ireland: 40.875M
```

Conclusion

Summary

Below I have gathered some interesting facts and discoveries concerning FIFA 20 and real-world football:

1) The in-game player statistics of the FIFA series are output created by a team of thousands of data gatherers, reviewers, and experts and they blend advanced statistics with opinions and suggestions of game-goers and pundits. Understandably they aren't free of bias but their work speaks and have reputation for that.

2) All the overall ratings can be explained almost entirely with the associated "key" attributes, even though the equations found do not reflect the game maker's approach at all. See the table above for details

3) The average footballer in FIFA 20 is of 25 years of age, stands just above 180cm, weighs around 75kg, and has a overall rating of 66-ish. He is right-footed, has a weak foot rating of 3, and skill moves rating of 2.

4) "shooting"&"dribbling" are very closely related to each other - "passing"&"dribbling" too. The goalkeeper attributes are very closely related to each other and many of them are almost directly related to overall rating.

5) In FIFA 20, the gap between the very top players and 75th percentile is much greater than that of between 75th percentile and median-ish players, at least in terms of numbers. Attributes "pace" and "physic" do not vary much though.

6) Based on the 2018-19 season, EPL has the edge over all the other major European leagues when it comes to overall ratings. The weakest among the major five is Ligue 1. At the very pinnacle, however, the Spanish La Liga has the edge, and that is the case for market value and weekly wage. Overall the Bundesligans are the youngest and the tallest. The greatest weekly wage gap is seen in La Liga - the least, Ligue 1. For more refer to above.

7) Based on the 2018-19 season, the top 10 world's most valuable clubs were, in the following order: Real Madrid, Barcelona, Manchester City, Juventus, Liverpool, Bayern Munich, Paris Saint-Germain, Tottenham, Atletico Madrid, Dortmund.

The top 10 highest-paying clubs were the following: Real Madrid, Barcelona, Manchester City, Juventus, Bayern Munich, Manchester United, Chelsea, Liverpool, Tottenham, Paris Saint-Germain.

Lastly, the top 10 strongest clubs (based solely on average overall rating of top 15 players for each) in the same season have been the following: Barcelona, Real Madrid, Manchester City, Juventus, Bayern Munich, Paris Saint-Germain, Liverpool, Tottenham, Dortmund, Atletico Madrid.

8) The top 5 strongest national teams come from Spain, Brazil, France, Germany, and Belgium. The top 5 most valuable national teams come from France, Brazil, Germany, Belgium, and Argentina.

Moving Forward

Although I did not quite follow my initial curiosity-driving question I believe I still ended up with some interesting and meaningful insights, most of which requires some wrangling with this particular type of dataset. I think I am most satistied with the linear models I found in the beginning, as they are incredibly well-performing and offer something that cannot be found anywhere else. It's been a fruitful project overall and I had fun exploring it.

There's room for improvement and I would be happy to develop even further in the future, maybe for FIFA 21. A wonderful dashboard and maybe some other advanced ML algorithms could bring the project to another level...

The biggest challenge for me was figuring out what to ask and how to go about approaching what I'd like to know. With limited knowledge I was forced to search the web many times and educate myself with some knowledge concering the dataset and football in general. What's more, football is such a complex sport and there's so many variables that come into play, and this makes coming up with "meaningful" and "accurate" insights and findings to be rather demanding. Now I definitely understand why "domain knowledge" and "business insights" are deemed critical in data science...

Anyways, this has been an amateur football fan's journey of storytelling football with data, and if you've been reading the whole thing, thank you very much! I hope it had something interesting to offer.