



Analysis on Soccer Player Attributes

By: Sau Kha
April 2018



Executive Summary

The purpose of this project is to study player attributes of a European soccer database from Kaggle.com. The database consists of eight tables, sqlite_sequence, Player_Attributes, Player, Match, League, Country, Team and Team Attributes. The database was populated with more than 25,0000 matches from 2008 to 2016 seasons, 10,000 players, and 11 European countries with their lead championship. Players' and teams' attributes are sourced from EA Sports' FIFA video game series (<https://sofifa.com/>).

Research Questions:

The research questions for this project are:

1. What attributes set the players apart?
2. Which player attribute contributes most to a player's overall rating?

Data Cleaning:

Data analysis was conducted in Python using various libraries/modules such as, sqlite3, pandas, principal component analysis from scikit-learn, matplotlib, and seaborn. Data from the Player and the Player_Attributes tables were queried with an inner join query on the player identification number from the FIFA application programming interface (API). Queried data comprises of 183929 rows and 50 columns. Data was then cleaned so that records with missing or null data in any column was removed. Duplicated data, if any, was removed. Final dataset consists of 10898 rows and 50 columns. 38 columns of attributes with numeric data were identified for principal component and further analysis.

Data Exploration and Analysis:

Principal component analysis was conducted. Results shows that the first three principal components out of 38 dimensions, PC1, PC2 and PC3, explain 43.9%, 15.7% and 8.9%, respectively, which only explained a total of 68.5% of the variance in the dataset. Loading scores of all player attributes in each of the three principal components show that no one single player attribute contributes significantly more than any other attributes to these three principal components. Various charts indicate that there are two distinct (large and small) subgroups of players in the dataset. Created scatter plots, distribution plots, joinplots and correlation coefficient analysis explained the clusters and substantiated the answers to the research questions.

Conclusions:

1. The following attributes set players apart into subgroups: gk_diving, gk_reflexes, gk_handling, gk_positioning, and ball control. The first four attributes pertain to goalkeeper position while ball control relate to more to non_goalkeepers. Goalkeepers score relatively high in the goalkeeping attributes, but low in ball control, while the rest of players score in the opposite way. Thus, players were set into one small subgroup of goalkeepers and one large group of the rest of the players. Gk_kicking, a goalkeeping attribute further divided the larger subgroup into two: some players in the back field had high scores in gk_kicking that had a moderate positive linear correlation to their ball control scores.
2. Considering all players, reactions attribute had a strong positive linear correlation with overall rating, thus contributes most to the rating. The higher the reactions score, the higher the overall rating of the player.
3. However, at the subgroup level, for the goalkeeper subgroup, gk_diving, gk_handling, gk_positioning and gk_reflexes attributes had very strong positive linear correlation with overall rating and contribute most to the players' rating. For the rest of the players, reactions attribute had a strong correlation and contributed most to their overall rating.

Appendix A shows information and sources of data. Appendix B presents all the Python scripts used for data queries, cleaning, exploration and analysis and their outputs.

Table of Contents

Executive Summary	i
Introduction.....	1
Research questions.....	1
Intended Audience	1
Reference	1
Methodology and Analysis Approach	2
Findings.....	3
Principal Component Analysis.....	3
Visualization	4
Research Question: What attributes set the players apart?	4
Research Question: Which player attribute contributes most to a player's overall rating?	12
Conclusions and Answers to Research Question.....	16
Research Question: What attributes set the players apart?	16
Research Question: Which player attribute contributes most to a player's overall rating?	17
Limitation and Future Study	17
APPENDIX A: Source of Data.....	
APPENDIX B: Jupyter Notebook - Python Codes and Outputs	

List of Tables

Table 1. Loading Scores of Player Attributes in the First Principal Component, PC1.....	4
Table 2. Correlation Coefficient between Overall Rating and All Numeric Attributes	12
Table 3. Correlation Coefficient between Overall Rating and All Numeric Attributes - Goalkeeper Subgroup	13
Table 4. Correlation Coefficient between Overall Rating and All Numeric Attributes - Non_Goalkeeper Subgroup	14

List of Figures

Figure 1. Scree Plot: Percentage of Explained Variance	3
Figure 2. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_diving	5
Figure 3. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_handling	5
Figure 4. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_positioning	5
Figure 5. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_reflexes.....	6
Figure 6. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_kicking	6
Figure 7. PC1 Versus PC2 and PC1 Versus PC3 – Colored by Ball Control.....	6
Figure 8. Distribution Plots of All Player Attributes	7
Figure 9. Distribution Plots of Soccer Players by Goalkeeping Attributes: Diving, Handling, Kicking, Positioning and Reflexes.....	8
Figure 10. Distribution Plots of Soccer Players by Ball Control Attribute.....	8
Figure 11. Jointplot: Ball Control Attribute Versus gk_Diving Attribute	9
Figure 12. Jointplot: Ball Control Attribute Versus gk_Handling Attribute	9
Figure 13. Jointplot: Ball Control Attribute Versus gk_Positioning Attribute	10
Figure 14. Jointplot: Ball Control Attribute Versus gk_Reflexes Attribute	10
Figure 15. Jointplot: Ball Control Attribute Versus gk_Kicking Attribute	11
Figure 20. Scatter Plot of Reactions Attribute Against Overall Rating	12
Figure 21. Scatter Plot of Reactions Attribute Against Overall Rating with Hue by Defensive Work Rate and Attacking Work Rate	13
Figure 22. Non_Goalkeepers Subgroup: Scatter Plot of Reactions Attribute Against Overall Rating ..	14
Figure 26. Players and Team Formation of Soccer Team, Liverpool, for 2018	18
Figure 27. Attributes of a Player from the Soccer Team, Liverpool and Similar Players with Overall Rating and Potential	18
Figure 28. Examples of Model Soccer Team Formations	19

Introduction

This purpose of this project is to study player attributes of a European soccer database from Kaggle.com. The database consists of eight tables, sqlite_sequence, Player_Attributes, Player, Match, League, Country, Team and Team Attributes. The database was populated with more than 25,0000 matches from 2008 to 2016 seasons, 10,000 players, and 11 European countries with their lead championship. Players' and teams' attributes are sourced from EA Sports' FIFA video game series (<https://sofifa.com/>).

To be attain great team performance, individual soccer player typically undertakes intensive drills to master specific skills for his dedicated position(s). In addition to basic skills, specialized training may focus in attack, defensive, mentality, movement, power or goalkeeping attributes. When highly skillful soccer players work together as a team in a strategic team formation, the likelihood to win a game may increase significantly.

In the dataset, each soccer player was given an overall rating and potential score, along with scores for the following numeric attributes: crossing, finishing, heading accuracy, short passing, volleys, dribbling, curve, free kick accuracy, long passing, ball control, acceleration, sprint speed, agility, reactions, balance, shot power, jumping, stamina, strength, long shots, aggression, interceptions, positioning, vision, penalties, marking, standing tackle, sliding tackle, gk_diving, gk_handling, gk_kicking, gk_positioning and gk_reflexes.

Research questions

The research questions for this project are:

1. What attributes set players apart?
2. Which player's attribute contributes most to a player's overall rating?

Intended Audience

The intended audience for this project are: soccer fans and FIFA video game players.

Reference

1. Database source: <https://www.kaggle.com/hugomathien/soccer/data>
2. Original data sources:
 - a. <http://football-data.mx-api.enetscores.com/> : scores, lineup, team formation and events
 - b. <http://sofifa.com/> : players and teams attributes from EA Sports FIFA games. *FIFA series and all FIFA assets property of EA Sports*
3. Merriam-Webster – Visual Dictionary Online: “player positions”
<http://www.visualdictionaryonline.com/sports-games/ball-sports/soccer/player-positions.php>
4. Video from Bing.com: “Soccer Goalie Diving Drills”:
<https://www.bing.com/videos/search?q=soccer+goalie+diving+drills&view=detail&mid=95069C368876787EAEBE95069C368876787EAEBE&FORM=VIRE>
5. Wikipedia.org: “Association Football Positions”
https://en.wikipedia.org/wiki/Association_football_positions
6. YouTube video series: “StatQuest: Principal Component Analysis (PCA), Step-by-Step”
<https://www.youtube.com/playlist?list=PLblh5JKOoLUIcdIgu78MnlATeyx4cEVeR>
7. YouTube video: “Top 10 BEST Longest Goals by Goalkeepers”
<https://www.youtube.com/watch?v=QUp6OyrHoMc>

8. footy4Kids: “Goalkeeper coaching for U12 and up”
<http://www.footy4kids.co.uk/soccer-drills/goalkeeping/goalkeeper-coaching-for-u12-and-up/>

Methodology and Analysis Approach

The methodology for this study basically is first to explore, clean up, analyze, visualize the data and then draw conclusions and attempt to find answers to the research questions. This study was conducted in Python using various libraries/modules such as, sqlite3, pandas, principal component analysis from scikit-learn, matplotlib, and seaborn. The steps below were taken to reach the findings and conclusions:

1. Database was downloaded from <https://www.kaggle.com/hugomathien/soccer/data>
2. Table structures and field datatypes were explored. Tables from the downloaded sqlite database were exported and previewed.
3. Using IO tools from pandas library, read_sql and to_csv, the Player and the Player_Attributes tables were queried with an inner join SQL on the player identification number from the FIFA application programming interface (API). Queried data was saved to a csv file.
4. Queried data comprises of 183929 rows and 50 columns. Data was then cleaned so that records with missing or null data in any column was removed. Duplicated data, if any, was removed. Final dataset consists of 10898 rows and 50 columns.

Lessons Learned:

- a. I was surprised to find that a huge number of rows (players) were dropped till I read from the data source webpage and noted that the administrator of the database was not able to source quite some players' attributes from FIFA and have those fields blank.
- b. Data does not lie.
5. Identified thirty-eight (38) columns of players' attributes in numeric format for principal component analysis (PCA). Ran PCA to identify players' attributes that explained the most variance, to help reduce dimensions before further analysis. .

Lessons Learned:

- a. Loading scores from PCA did not support dimension reduction. At this time, a decision had to be made to how to proceed. Additional data exploration was needed to decide where to zoom in for further analysis. Since it did not go as what I expected, I turned to other tools to use and approach to take.
- b. I created a series of charts using a programming loop, which allowed exploration systematically on each attribute.
- c. Additionally, adding color in the scatter plots for each point added a third dimension and presented a third attribute in the charts. The added information was easier on human eyes when compared to adding a z-axis for the third attribute.
- d. Joint plots provided distribution of player variables that were presented in a scatter plot. These techniques provided valuable information as how to proceed with data analysis to get to the bottom to find answers to the research questions.
- e. Scatter plot matrix contains all the pairwise scatter plots of the variables on a single page in a matrix format. This was very useful to help roughly determine whether there was a linear correlation between multiple variables.
6. Conducted further data exploration and visualization with scatter plots matrix, scatter plots, distribution plots and joint plots.
7. Identified clusters with further analysis.

8. Calculated correlation coefficient matrix for all attributes. Located attributes with strong correlation coefficient with overall rating. This was repeated for the goalkeeper subgroup and for the rest of the players. Correlated attributes were identified. Created scatter plots for the correlated attributes for visual confirmation.
9. Draw conclusions based on findings.

Findings

Principal Component Analysis

Figure 1 presents the Scree Plot of all the principal components from the 38 numeric attributes of the final, cleanup dataset of soccer players. PCA results shows that the first three principal components, PC1, PC2 and P32, out of 38 dimensions explain 43.9%, 15.7% and 8.9% of variance, respectively. This is a total of 68.5% of all variance.

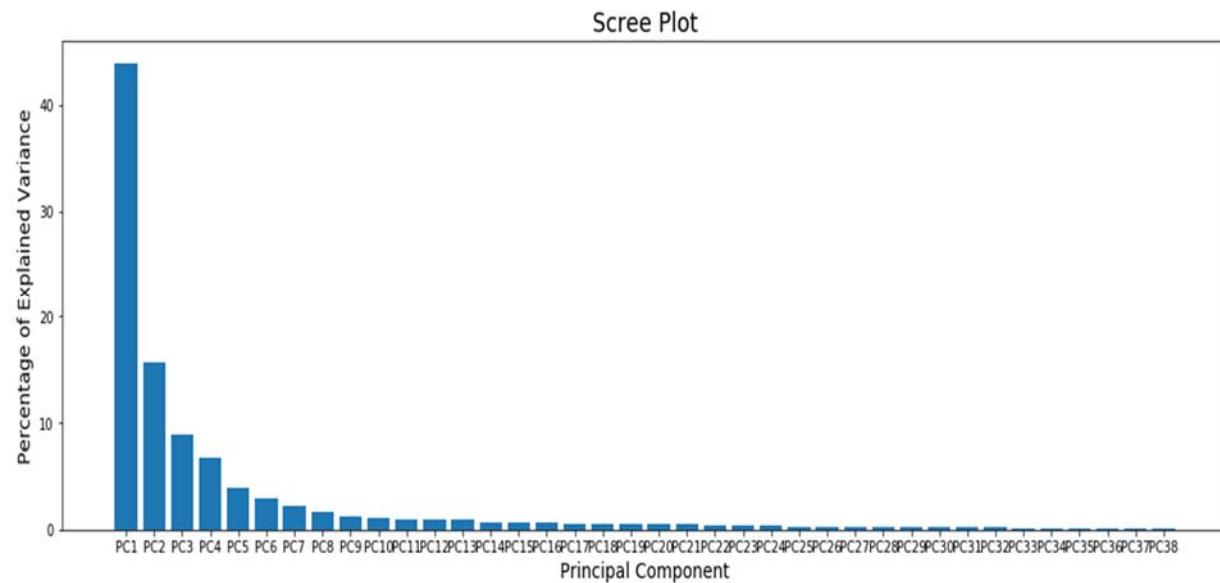


Figure 1. Scree Plot: Percentage of Explained Variance

Table 1 below lists the loading scores of player attributes in the first principal component. Loading scores of all player attributes in the first three principal components show that no one single player attribute contributes significantly more than any other attributes to the variance. None the less, the ball Control attribute was the leading attribute that explained the most variance. Considering the players as one group, overall rating has a low loading score in the PC1 dimension.

Table 1. Loading Scores of Player Attributes in the First Principal Component, PC1

Player Attributes	Loading Scores	Player Attributes	Loading Scores	Player Attributes	Loading Scores
<u>ball_control</u>	<u>0.233895</u>	gk_positioning	0.193662	gk_kicking	0.121982
dribbling	0.226609	volleys	0.193077	reactions	0.105162
short_passing	0.220168	finishing	0.191773	aggression	0.087644
crossing	0.213529	acceleration	0.185480	<u>overall_rating</u>	<u>0.083033</u>
Curve	0.211442	penalties	0.183370	potential	0.080174
long_shots	0.211257	sprint_speed	0.182038	interceptions	0.071469
positioning	0.204851	long_passing	0.177636	standing_tackle	0.065230
shot_power	0.199655	agility	0.168734	sliding_tackle	0.059237
vision	0.197614	stamina	0.158548	marking	0.052163
gk_diving	0.196600	balance	0.153811	strength	0.028138
free_kick_accuracy	0.196366	height	0.134852	age	0.005573
gk_reflexes	0.195234	heading_accuracy	0.132158	jumping	0.001561
gk_handling	0.195011	weight	0.125473		

Visualization

Research Question: What attributes set the players apart?

From the findings above, it was decided not to reduce any dimension for further analysis. Scatter plots were created using data from the first three principal components: PC1 versus PC2 and PC1 versus PC3. Two distinct clusters were noted. The largest variation is seen in the first principal component dimension, PC1.

To find out what player attribute(s) plays a part in distinguishing the players into the subgroups, color was added in the plots for one player attribute at a time for all 38 attributes (lighter hue indicates higher scores). Scatter plots in Figures 2 through 6 indicate that goalkeeper attributes play a part in separating the players into the two clusters. It is reasonable to believe that the smaller subgroup shown on the right of the scatter plots consist of goalkeepers, who score higher in goalkeepers attributes (lighter hue). There are two reasons: a) Each team usually designates only one player as the goalkeeper, hence makes goalkeepers a small subgroup in the dataset. b) Goalkeepers, who focus their training in goalkeeping skills, should score higher in these attributes.

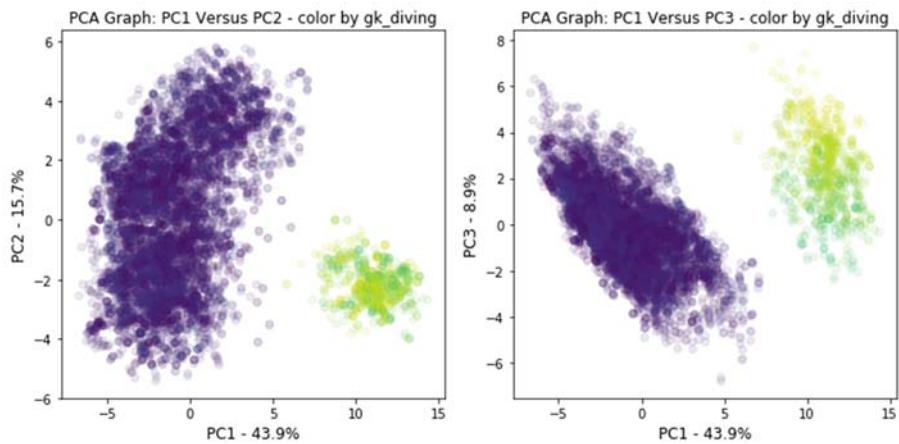


Figure 2. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_diving

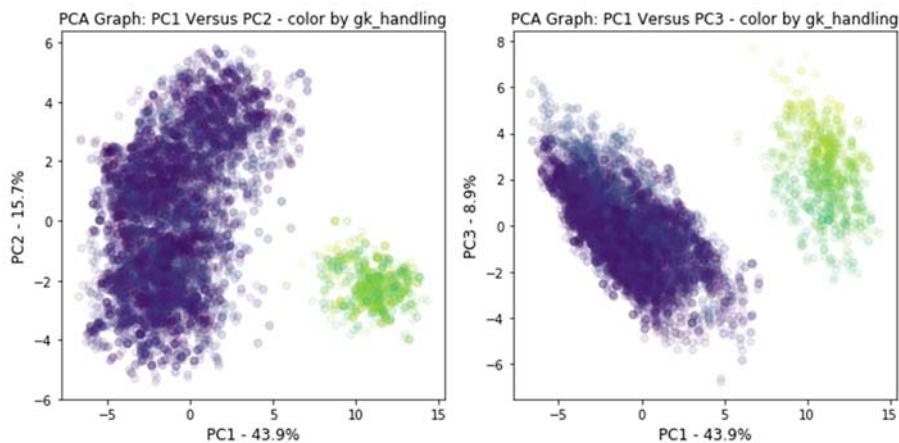


Figure 3. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_handling

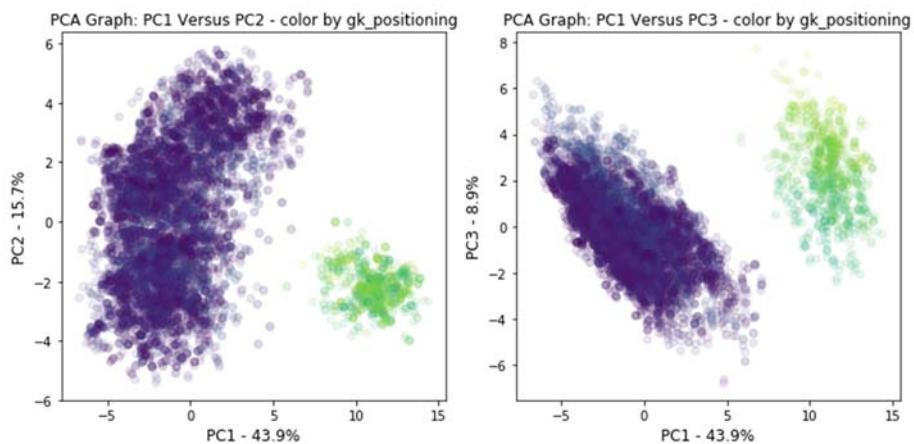


Figure 4. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_positioning

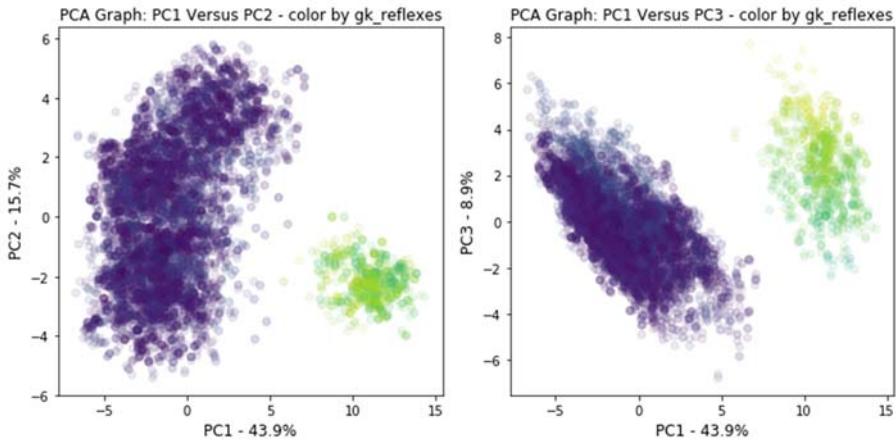


Figure 5. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_reflexes

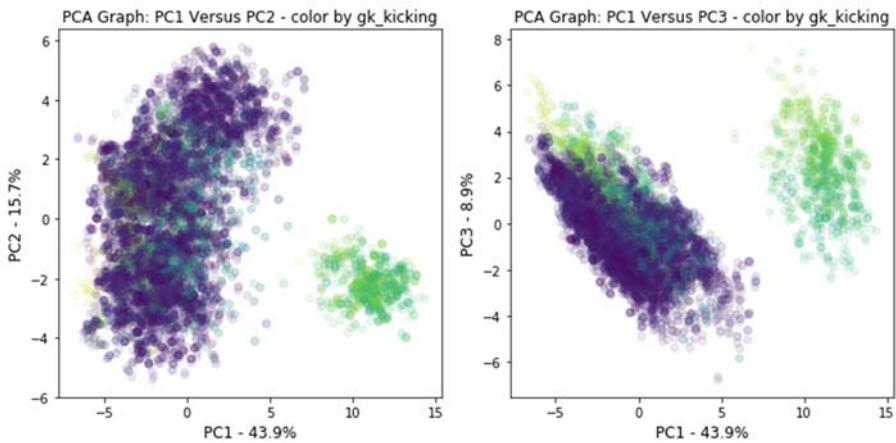


Figure 6. Scatter Plots of PC1 Versus PC2 and PC1 Versus PC3 – Colored by gk_kicking

Ball control attribute, with the highest loading score of 0.2339 in the first principal component, explains the most variance in the x-axis, PC1 dimension. Figure 7 below indicates that the small subgroup of goalkeepers on the right scores lower (darker hue) in the ball control attribute than the rest of the players (larger subgroup on the left). This attribute also set the players apart.

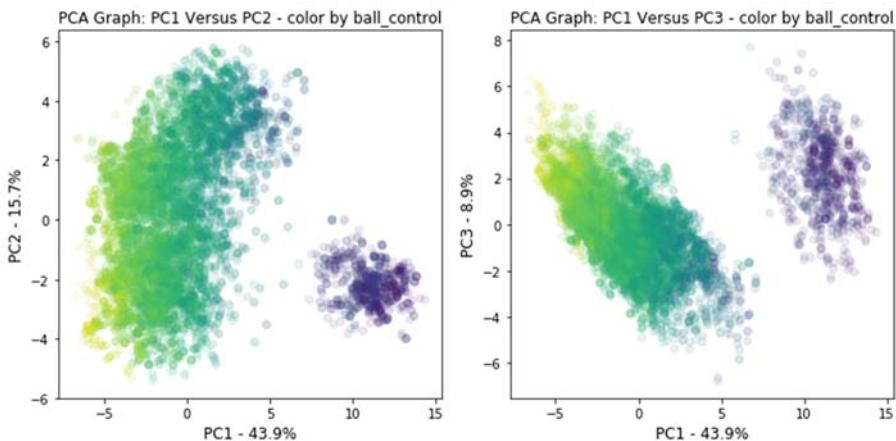


Figure 7. PC1 Versus PC2 and PC1 Versus PC3 – Colored by Ball Control

Figure 8 shows the distribution plots of all players attributes: age, height, weight, overall_rating, potential, crossing, finishing, heading_accuracy, short_passing, volleys, dribbling, curve, free_kick_accuracy, long_passing, ball_control, acceleration, sprint_speed, agility, reactions, balance, shot_power, jumping, stamina, strength, long_shots, aggression, interceptions, positioning, vision, penalties, marking, standing_tackle, sliding_tackle, gk_diving, gk_handling, gk_kicking, gk_positioning, gk_reflexes. It presents further evidence of separation of the players into subgroups by a few player attributes.

The distribution plots below show a small subgroup of players with relatively higher scores in gk_diving, gk_handling, gk_kicking, gk_positioning and gk_reflexes goalkeeping attributes (yellow highlight below and Figure 9) but with relatively low scores in ball control attribute (red highlight below and Figure 10). This reaffirms that goalkeeping and ball control attributes set the soccer players (goalkeepers) apart from the rest of players. This conclusion agrees with findings from the scatter plots in PC1 versus PC2 and PC1 versus PC3, in which a smaller subgroup is associated with lighter hue for relatively higher scores in goalkeeping attributes (Figures 2 through 6) and with darker hue for lower scores in ball control attribute (Figure 7).

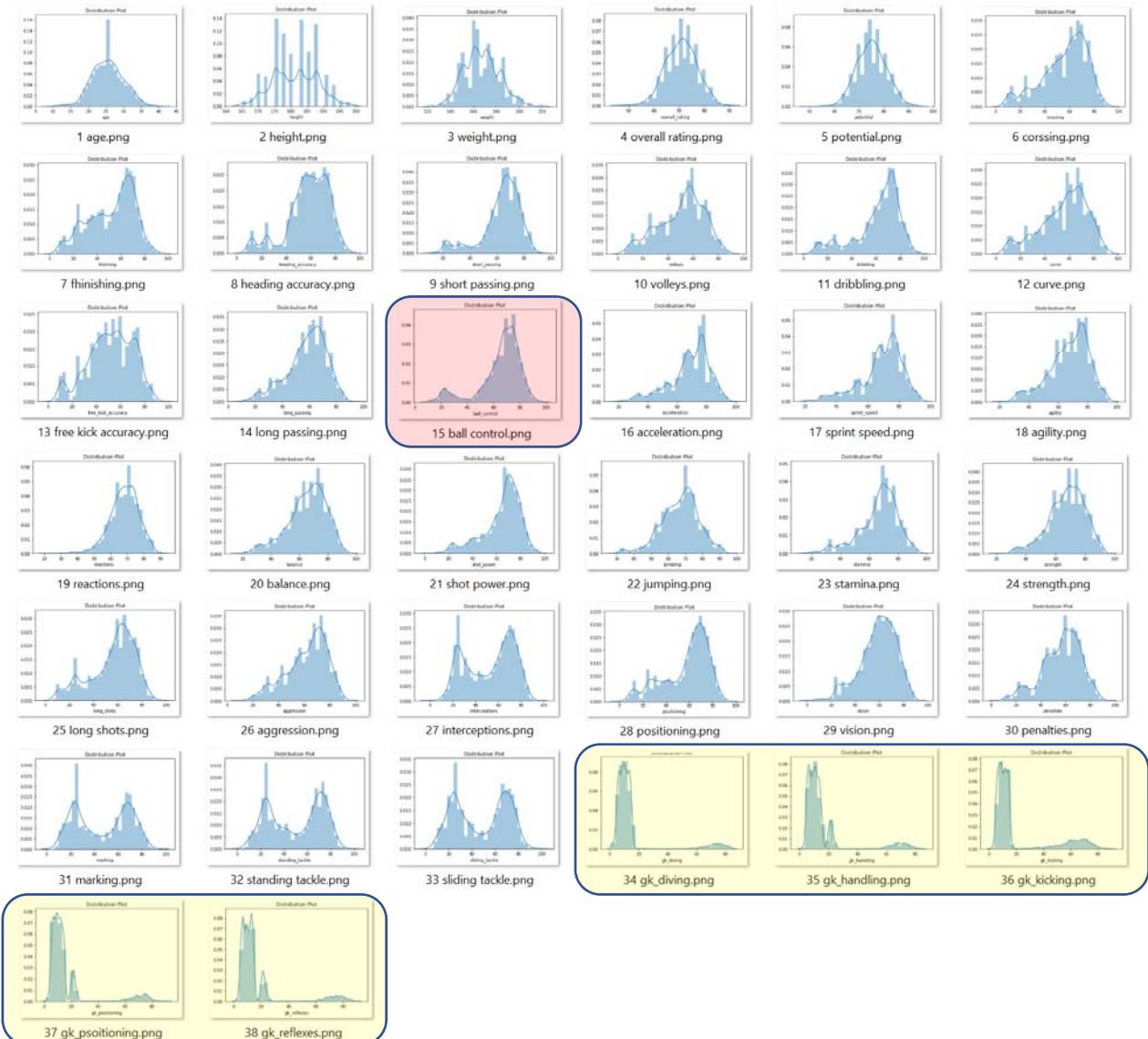
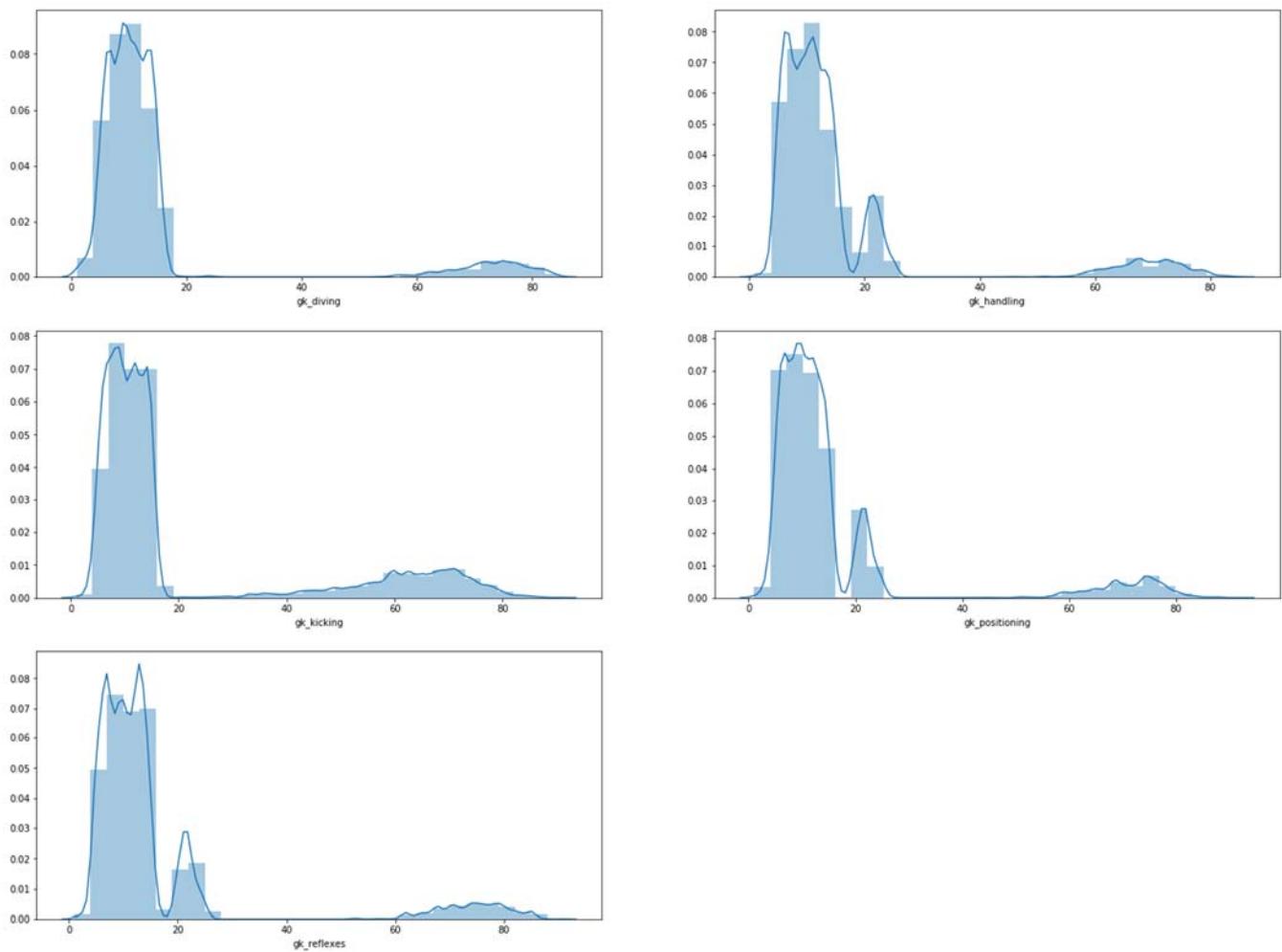


Figure 8. Distribution Plots of All Player Attributes



*Figure 9. Distribution Plots of Soccer Players by Goalkeeping Attributes:
Diving, Handling, Kicking, Positioning and Reflexes*

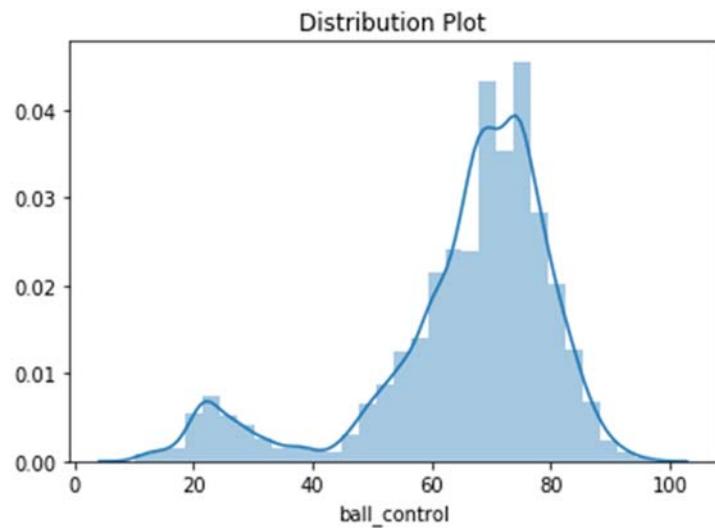


Figure 10. Distribution Plots of Soccer Players by Ball Control Attribute

Visualization in Ball Control Attribute Versus Goalkeeping Attributes

With the findings from above, jointplots were created using ball control attribute against the five goalkeeping attributes (see Figures 11 through 15). These scatter plots joint with distribution plots visualize goalkeepers as the smaller subgroup (top left) and non-goalkeepers as the larger subgroup (bottom).

Ball Control Vs gk_Diving

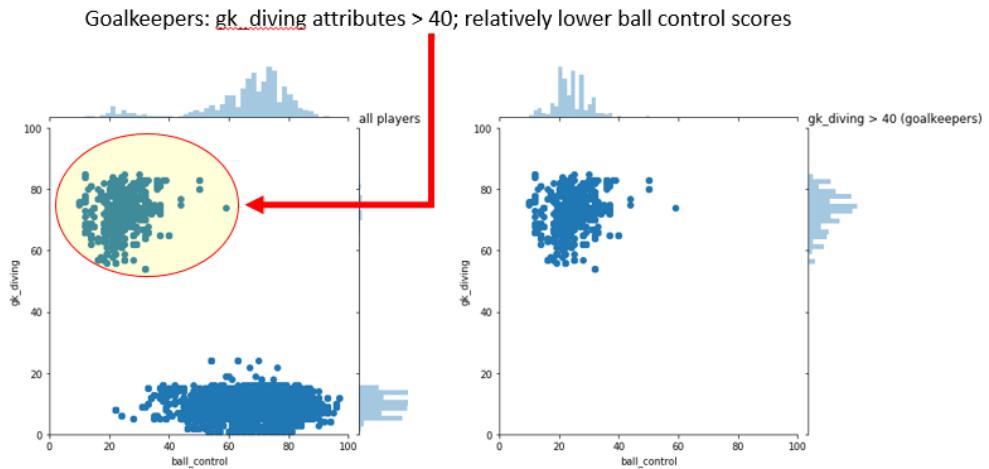


Figure 11. Jointplot: Ball Control Attribute Versus gk_Diving Attribute

Ball Control Vs gk_Handling

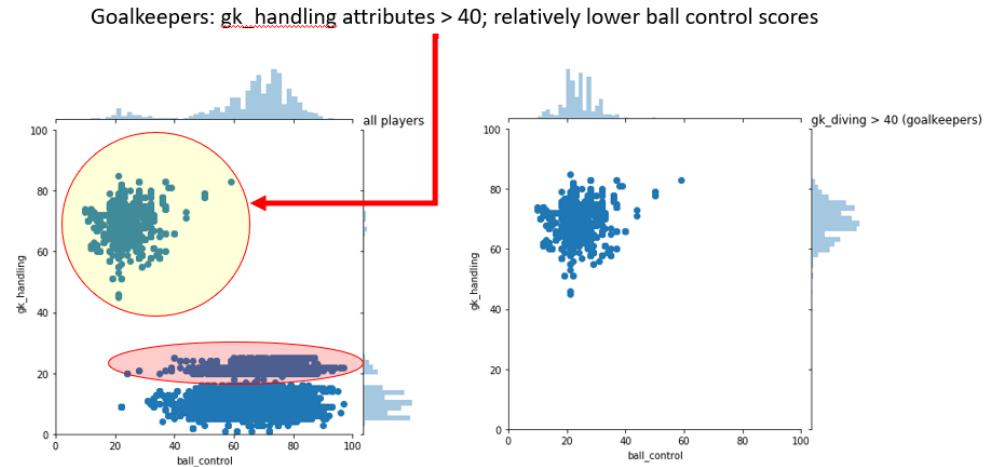


Figure 12. Jointplot: Ball Control Attribute Versus gk_Handling Attribute

Ball Control Vs gk_Positioning

Goalkeepers: gk_positioning attributes > 40; relatively lower ball control scores

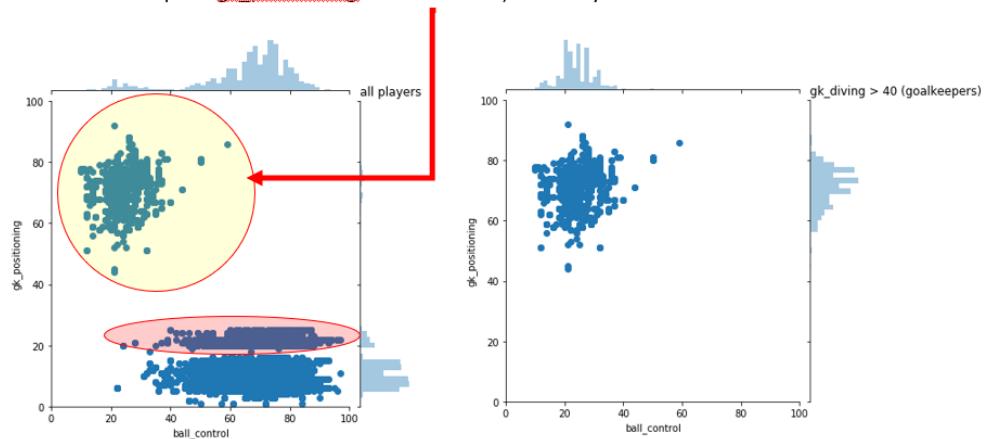


Figure 13. Jointplot: Ball Control Attribute Versus gk_Positioning Attribute

Ball Control Vs gk_Reflexes

Goalkeepers: gk_reflexes attributes > 40; relatively lower ball control scores

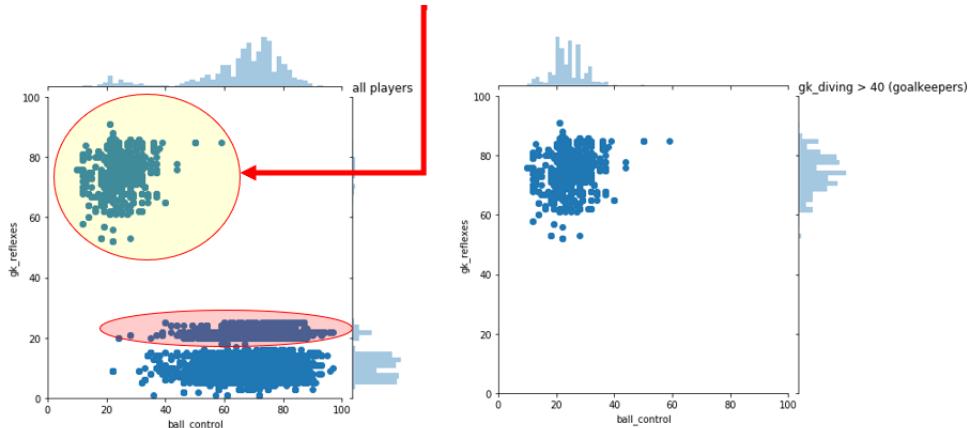
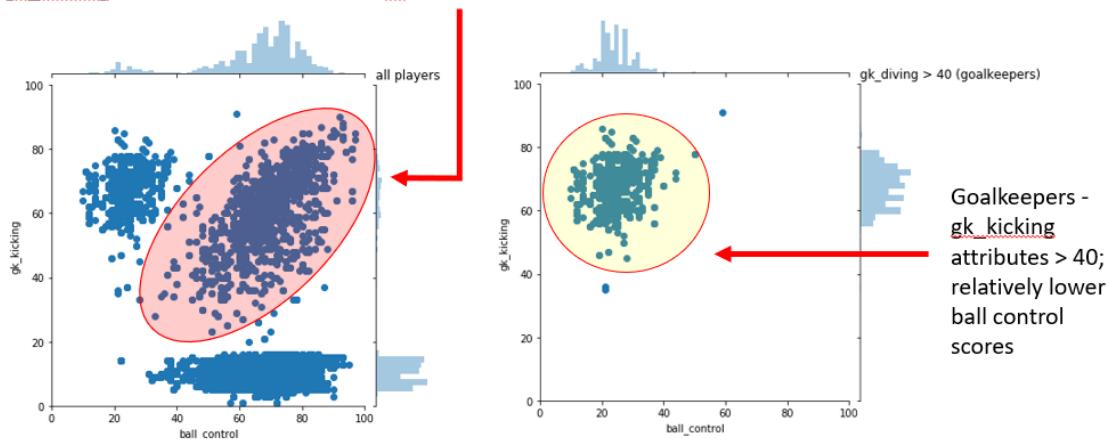


Figure 14. Jointplot: Ball Control Attribute Versus gk_Reflexes Attribute

Ball Control Vs gk_Kicking

non-goalkeepers - gk_diving < 40

gk_kicking attribute has a moderate +ve linear correlation with ball control attribute



gk_kicking attribute further separates the bigger subgroup into two smaller subgroups

Figure 15. Jointplot: Ball Control Attribute Versus gk_Kicking Attribute

Side Notes: gk_Kicking Attribute

Noted in Figure 15, non-goalkeepers were further split apart into two subgroups by gk_kicking attribute. The group highlighted in red had a moderate positive linear correlation between their scores in gk_kicking attribute and their scores in ball control attribute. This is not surprising as there may be some common techniques that are pertinent to both attributes that are important for certain player positions. Similar logic goes for other attribute pairs, such as: acceleration and sprint speed attributes (correlation coefficient = 0.9129), and ball control and dribbling (0.9273). It would be interesting to conduct further analysis to find out whether the players highlighted red in Figures 12 through 14 are the same group of players highlighted red in Figure 15.

Research Question: Which player attribute contributes most to a player's overall rating?

All Players

Correlation coefficient matrix was generated for all 38 player attributes. Table 2 below lists the correlation coefficients between overall rating and all other player attributes. Overall rating and potential has the highest correlation coefficient, 0.7840. Since the potential score may be considered some sort of rating similar to overall rating, the second highest correlation coefficient is considered. Overall rating has a strong positive correlation coefficient of 0.7248 with the reactions attribute (see highlighted).

Table 2. Correlation Coefficient between Overall Rating and All Numeric Attributes

	overall_rating		overall_rating		overall_rating
age	0.3826	long_passing	0.4300	interceptions	0.2373
height	0.0259	ball_control	0.3726	positioning	0.2782
weight	0.0351	acceleration	0.2137	vision	0.3992
overall_rating	1.0000	sprint_speed	0.2184	penalties	0.3373
potential	0.7840	agility	0.2211	marking	0.1106
crossing	0.2911	reactions	0.7248	standing_tackle	0.1456
finishing	0.2694	balance	0.1094	sliding_tackle	0.1159
heading_accuracy	0.2403	shot_power	0.3703	gk_diving	0.0501
short_passing	0.4161	jumping	0.2112	gk_handling	0.0373
volleys	0.2908	stamina	0.2487	gk_kicking	0.0795
dribbling	0.2991	strength	0.2404	gk_positioning	0.0361
curve	0.2987	long_shots	0.3267	gk_reflexes	0.0437
free_kick_accuracy	0.3048	aggression	0.2615		

For a visual confirmation, a scatter plot of the reactions attribute against overall rating with trendline was created. Figure 20 shows a strong positive linear correlation. Figure 21 presents the correlation relationship with hue for two categorical attributes, defensive work rate and attacking work rate. Both plots show strong positive linear correlation within each subcategory by defensive work rate and by attacking work rate.

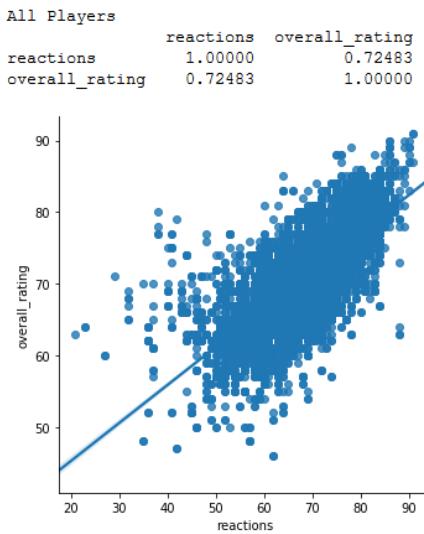


Figure 20. Scatter Plot of Reactions Attribute Against Overall Rating

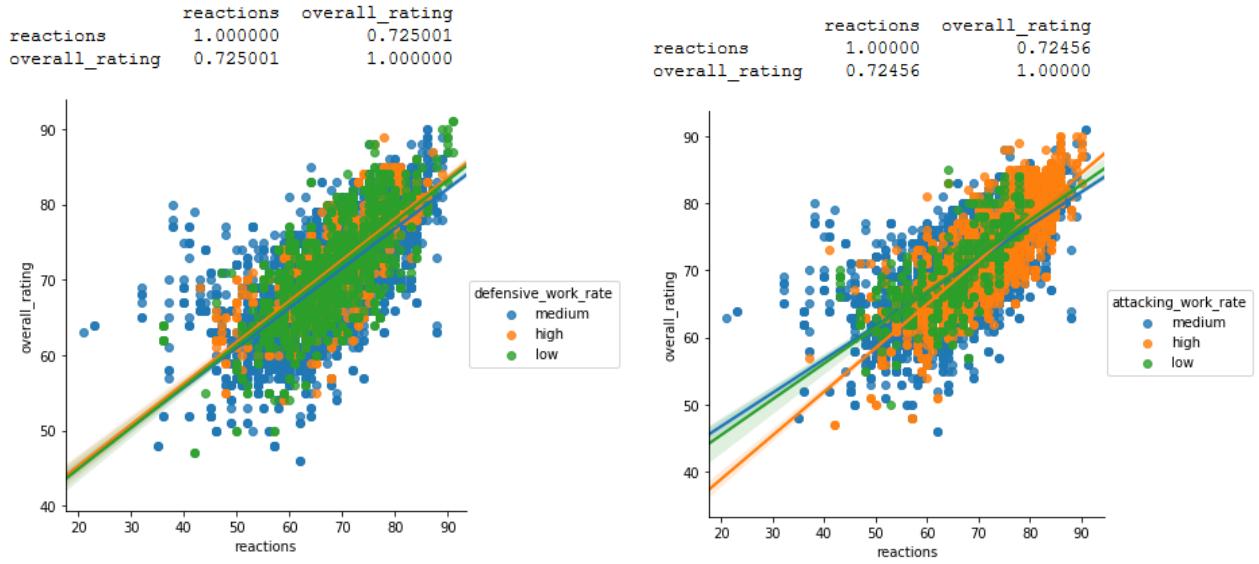


Figure 21. Scatter Plot of Reactions Attribute Against Overall Rating with Hue by Defensive Work Rate and Attacking Work Rate

Correlation Coefficients in Subgroups

However, there are two distinct subgroups in the dataset (goalkeepers and the rest players). Hence, the correlation of overall rating with player attributes is further evaluated in each subgroup separately. Tables 3 and 4 list the correlation coefficients between overall rating and all numeric attributes for the goalkeeper subgroup and the rest players, respectively. Highlighted numbers indicate strong positive correlation.

Table 3. Correlation Coefficient between Overall Rating and All Numeric Attributes - Goalkeeper Subgroup

	overall_rating	overall_rating	overall_rating
age	0.4274	long_passing	0.0919
height	-0.0283	ball_control	0.2050
weight	-0.0251	acceleration	0.3000
overall_rating	1.0000	sprint_speed	0.2066
potential	0.7805	agility	0.3076
crossing	0.0837	reactions	0.5609
finishing	0.0037	balance	-0.0955
heading_accuracy	-0.0007	shot_power	0.1054
short_passing	0.1165	jumping	0.4408
volleys	0.1428	stamina	0.0990
dribbling	0.0882	strength	0.2180
curve	0.0373	long_shots	0.0251
free_kick_accuracy	0.0166	aggression	0.1304
		interceptions	0.0655
		positioning	-0.0385
		vision	0.1784
		penalties	-0.0169
		marking	0.0523
		standing_tackle	0.1062
		sliding_tackle	0.1638
		gk_diving	0.8979
		gk_handling	0.8281
		gk_kicking	0.6697
		gk_positioning	0.8632
		gk_reflexes	0.8778

*Table 4. Correlation Coefficient between Overall Rating and All Numeric Attributes
- Non_Goalkeeper Subgroup*

	overall_rating		overall_rating		overall_rating
age	0.3814	long_passing	0.5553	interceptions	0.2728
height	0.0257	ball_control	0.6467	positioning	0.3837
weight	0.0368	acceleration	0.2624	vision	0.5061
overall_rating	1.0000	sprint_speed	0.2825	penalties	0.4523
potential	0.7848	agility	0.2420	marking	0.1299
crossing	0.4089	reactions	0.7595	standing_tackle	0.1708
finishing	0.3494	balance	0.1471	sliding_tackle	0.1333
heading_accuracy	0.3741	shot_power	0.5300	gk_diving	0.0717
short_passing	0.6655	jumping	0.1947	gk_handling	0.0059
volleys	0.3779	stamina	0.3278	gk_kicking	0.0744
dribbling	0.4587	strength	0.2426	gk_positioning	-0.0088
curve	0.4103	long_shots	0.4491	gk_reflexes	0.0254
free_kick_accuracy	0.4056	aggression	0.3122		

Visual Confirmation of Correlation in Subgroups

Scatter plots with trendlines were created for visual confirmation of correlation relationship between overall rating and correlated variables for each subgroup separately.

Non_Goalkeeper Subgroup

Figure 22 shows a stronger positive linear correlation between player reactions attribute and overall rating for non-goalkeeper subgroup alone, thus confirms that reactions attribute contributes most to the players' overall rating.

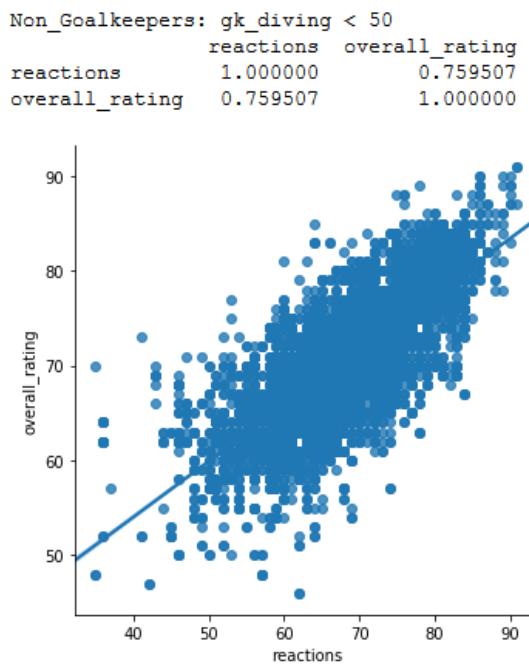


Figure 22. Non_Goalkeepers Subgroup: Scatter Plot of Reactions Attribute Against Overall Rating

Goalkeeper Subgroup

Figures 23 and 24 present the visual confirmation that gk_diving, gk_reflexes, gk_positioning and gk_handling attributes have a strong positive linear correlation with overall rating. The last goalkeeping attribute, gk_kicking, and reactions attribute only had a moderate positive linear correlation with overall rating (Figure 25).

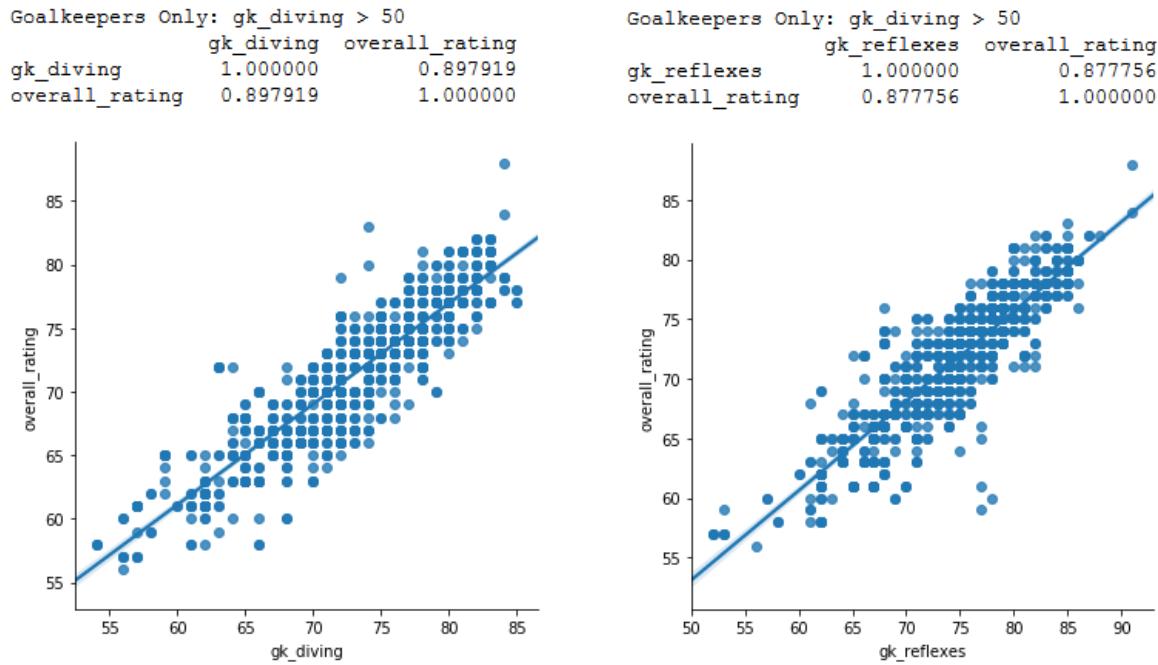


Figure 23. Goalkeeper Subgroup: Scatter Plots of gk_Diving (Left) and gk_Reflexes (Right) Attributes Against Overall Rating

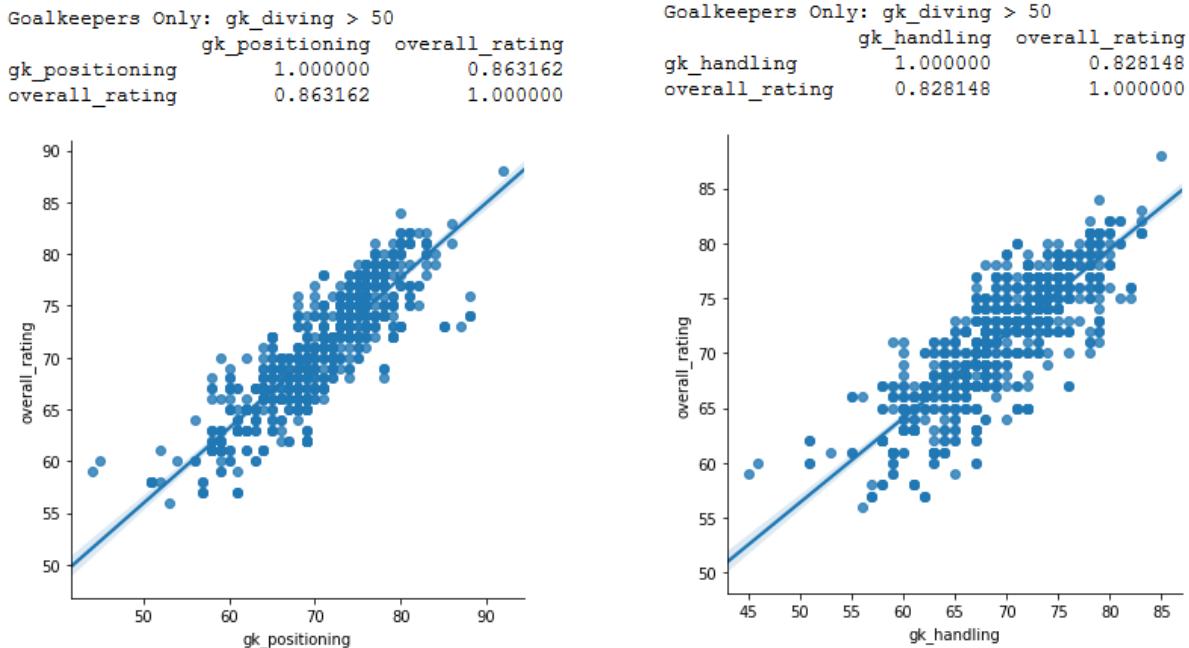


Figure 24. Goalkeeper Subgroup: Scatter Plots of gk_Positioning (Left) and gk_Handling (Right) Attributes Against Overall Rating

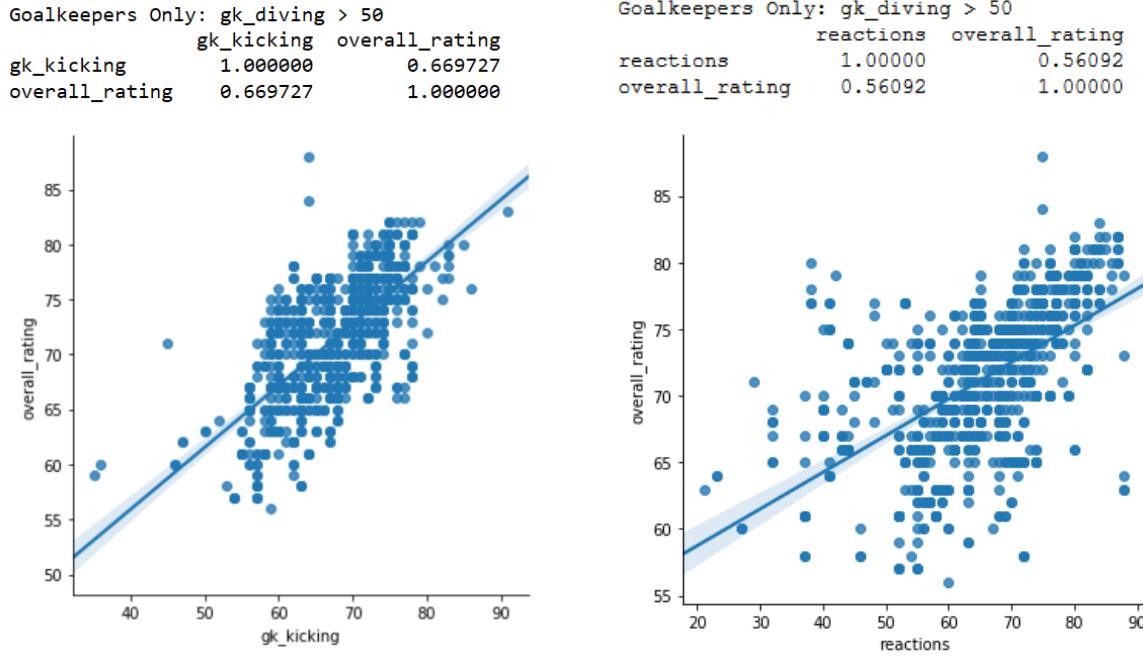


Figure 25. Goalkeeper Subgroup: Scatter Plots of gk_Kicking (Left) and Reactions (Right) Attributes Against Overall Rating

Conclusions and Answers to Research Question

Research Question: What attributes set the players apart?

Four of the five goalkeeping attributes and ball control attribute separated the players apart:

- a) a small subgroup of 930 goalkeepers, who have relatively high scores in goalkeeping attributes, gk_diving, gk_handling, gk_positioning, and gk_reflexes, but low in ball control, and
- b) a larger subgroup of the rest of the 9968 players, whose scores relatively low in the goalkeeping attributes, but high in ball control.

With gk_kicking attribute and ball control attribute, players were, however, divided into three apparent subgroups:

- a) one small subgroup of 930 goalkeepers, who have relatively high scores in gk_kicking, but low in ball control,
- b) one slightly larger subgroup of 1413 players, whose scores in gk_kicking (≥ 20) had a moderate positive linear correlation to their ball control scores, and
- c) one much larger subgroup of the rest of 8556 players, whose score in gk_kicking (< 20) had no correlation with their ball control scores.

Typically, soccer players concentrate their drills on the skills they need for their positions. Drills for goalkeepers are very different from other players because their position requires them to use primarily their hands to block the soccer ball from entering the goal. They typically divert the direction of the soccer away from the goal or take control of the soccer with their hands. Goalkeepers drills typically specialize in collecting balls, jumping to divert or collect overhead balls, goal positioning to the

attacking angle, diving for balls, ball distribution once collected, and so on. Specialized training brings special attributes that separate them apart from other players.

According to soccer game rules, if the offensive team kicks the ball out of the field, play is restarted with a goal kick. The goal kick can be taken by any player, rather than limited to the goalkeeper. So, it is not uncommon to have full back teammates to take the goal kick. This explains why there is a slightly larger subgroup of players who score well in gk_kicking and ball control.

Research Question: Which player attribute contributes most to a player's overall rating?

For all players in the dataset, reactions attribute with the strongest correlation coefficient (0.724830) contributes most to a player's overall rating.

When evaluating in subgroup levels, the answer is different for each group. For goalkeeper subgroup, gk_diving (corr coeff. = 0.897919), gk_reflexes (corr coeff. = 0.877756), gk_positioning (corr coeff. = 0.863162) and gk_handling (corr coeff. = 0.828148) goalkeeping attributes have very strong positive linear correlation with and contribute most to the players overall rating. For the rest of the players as the larger subgroup, reactions (corr coeff. = 0.759507) has a strong positive linear correlation with and contributes most to players' overall rating.

Limitation and Future Study

The limitation of this project is that the database does not contain player's position or team formation information for any game. So, to validate the answers to the research questions, such information will have to be obtained from other source, such as sofifa.com. Figure 26 shows the players and team formation of the soccer team, Liverpool, for 2018. Figure 27 presents attributes of a player from the team and similar players of other teams and their overall rating and potential scores.

As shown in Figure 28 there are various modern soccer team formations. Gray, blue, yellow and red dots represent players in goalkeeping, defensive, midfield, and attacking positions at the goal and in the back field, midfield and forefront. The number of players in the back, midfield and forward varies in different formations. But there is always only one goalkeeper. The drills of players may vary slightly based on their positions in the team formation. These team formation factors will affect the scores of various attributes for each player. However, there is a tactical theory in football, Total Football (Dutch: *totaalvoetbal*), in which any outfield player can take over the role of any other player in a team. The theory requires players to be comfortable in multiple positions; hence, it places high technical and physical demands on them. For teams that embrace this theory, players are likely to have attributes across positions. If player position and team formation information is available, the study may be bring to a whole new level for the future.



Figure 26. Players and Team Formation of Soccer Team, Liverpool, for 2018.

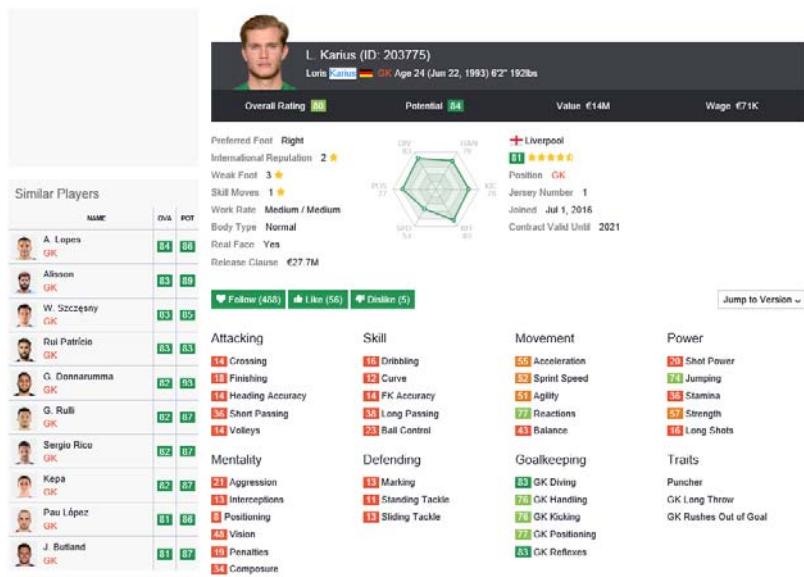


Figure 27. Attributes of a Player from the Soccer Team, Liverpool and Similar Players with Overall Rating and Potential

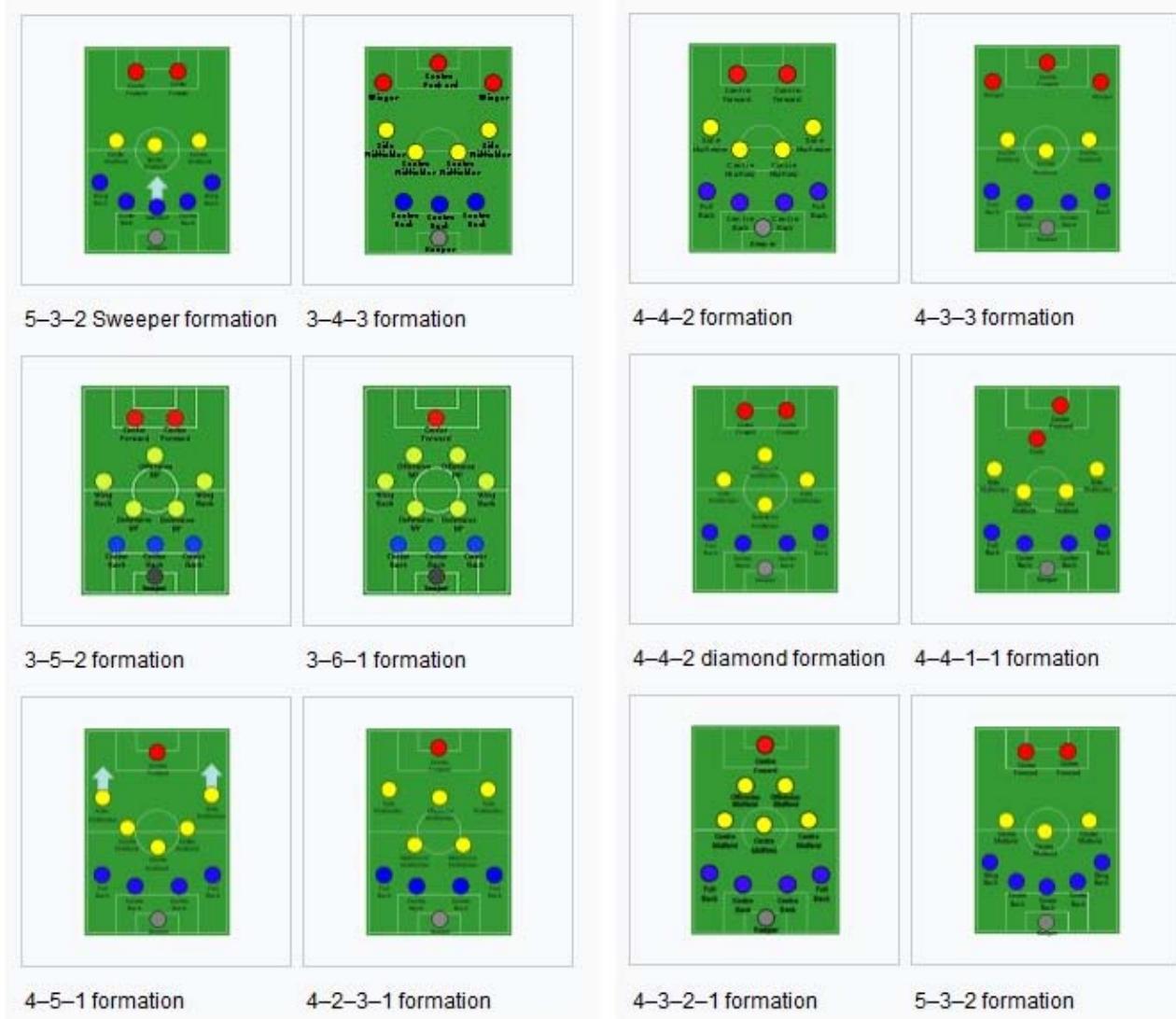


Figure 28. Examples of Modern Soccer Team Formations

APPENDIX A: Source of Data

Data for this project was downloaded from this website: <https://www.kaggle.com/hugomathien/soccer>

The following presents brief information of the database in SQLite3 format:

- +25,000 matches
- +10,000 players
- 11 European Countries with their lead championship
- Seasons 2008 to 2016
- Players and Teams attributes sourced from EA Sports FIFA video game series #
- Team line up with squad formation (X, Y coordinates)
- Betting odds from up to 10 providers
- Detailed match events (goal types, possession, corner, cross, fouls, cards etc...) for +10,000 matches

*16th Oct 2016: New table containing teams attributes from FIF !

Original data source for players and teams attributes tables: <http://sofifa.com/> : players and teams attributes from EA Sports FIFA games. *FIFA series and all FIFA assets property of EA Sports.* Foreign keys "api_id" for players and matches are the same as the original data sources.

APPENDIX B: Jupyter Notebook - Python Codes and Outputs

Import required libraries, fetch data from database into pandas.DataFrame and format data type for PCA

```
In [3]: import numpy as np
import sqlite3
import matplotlib.pyplot as plt
import datetime as DT
import seaborn as sns
np.set_printoptions(precision=5)
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
#### import the PCA library from scikit learn library
from sklearn.decomposition import PCA
from mpl_toolkits.mplot3d import Axes3D # didn't get to use this
%matplotlib inline

# Normalizing (scaling) the data is VERY important - indeed can be important to many machine
# learning algorithms. Take the original features and scale them so that they all have zero
# mean and unit variance
from sklearn import preprocessing

## pandas actually has a command to read_sql or read_sql_query and return a pandas.DataFrame
## coerce_float=True argument to force float data type
## Need to FIRST connect to the database by creating a connection object called conn.
conn = sqlite3.connect('database.sqlite')

# create a cursor object using the conn object method
# The cursor object has methods for accessing the data
# c = conn.cursor() # this is not needed for the pd.read_sql IO tool

# Get the database table list from information in the sqlite_master table
# Follow convention to type SQL commands in all caps

# preview all tables in the database
print ('=====')
print ('Tables in the database')
# Set the execute SQL command, Fetch and print all table names and info, and
# return a pandas DataFrame
df_tables = pd.read_sql("""SELECT * FROM sqlite_master WHERE type='table'""", conn)
print('df_tables shape: ', df_tables.shape)
print(df_tables)
print ('=====')
print('Player_Attributes table:')
print(df_tables.sql[1]) # get sql that CREATE the Player_Attribtues table
print ('=====')
```

```

print('Player table')
print(df_tables.sql[2]) # get sql that CREATE the Player table
print ('=====')
print ('Player table')
df_Player = pd.read_sql("""SELECT * FROM Player """, conn)
print('df_Player.shape:', df_Player.shape)
print(df_Player.columns)
print(df_Player.head())
print ('=====')
print ('Player_Attributes table')
df_Player_Attributes = pd.read_sql("""SELECT * FROM Player_Attributes""", conn)
print('df_Player_Attributes.shape:', df_Player_Attributes.shape)
print(df_Player_Attributes.columns)
print(df_Player_Attributes.head())
print ('=====')

# acquire data from database using pd.read_sql_query(sql, , ,)
# build SQL to SELECT all columns from both Player and Player_Attributes tables
# for rows reocrds w/ matching player_fifa_api_id
sql="SELECT * FROM Player INNER JOIN Player_Attributes ON Player.player_fifa_api_id=Player_Attributes.player_fifa_api_id;"
df_all_col=pd.read_sql_query(sql, conn, coerce_float=True, params=None, parse_dates=['birthday', 'date'], chunksize=None)
# calculate age of player at the time attributes were collected
df_all_col['age'] = (df_all_col.date - df_all_col.birthday).astype('timedelta64[Y]')

#Tally total score per player attribute category
df_all_col['total_attack'] = df_all_col.crossing + df_all_col.finishing +
df_all_col.heading_accuracy + \
                               df_all_col.short_passing + df_all_col.volleys
df_all_col['total_skill'] = df_all_col.dribbling + df_all_col.curve + df_all_col.free_kick_accuracy + df_all_col.long_passing + \
                               df_all_col.ball_control
df_all_col['total_movement'] = df_all_col.acceleration + df_all_col.sprint_speed + df_all_col.agility + \
                               df_all_col.reactions + df_all_col.balance
df_all_col['total_power'] = df_all_col.shot_power + df_all_col.jumping + df_all_col.stamina + df_all_col.strength + \
                               df_all_col.long_shots
df_all_col['total_mentality'] = df_all_col.aggression + df_all_col.interceptions + df_all_col.positioning + \
                               df_all_col.vision + df_all_col.penalties
df_all_col['total_defending'] = df_all_col.marking + df_all_col.standing_tackle + df_all_col.sliding_tackle
df_all_col['total_goalkeeping'] = df_all_col.gk_diving + df_all_col.gk_handling + df_all_col.gk_kicking + \
                               df_all_col.gk_positioning + df_all_col.gk_reflexes
print('df_all_col.columns: ', df_all_col.columns) # print column labels for all columns from both tables
print('df_all_col.shape:', df_all_col.shape)
print(df_all_col.info())

```

```
# identify non_numeric and numeric columns of interest and create two lists of column labels
non_numeric_col=['player_fifa_api_id', 'player_api_id','player_name', 'birthday', 'date', \
                  'preferred_foot', 'attacking_work_rate', 'defensive_work_rate']
numeric_col = [ 'age', 'height', 'weight','overall_rating', 'potential','crossing', 'finishing', \
                  'heading_accuracy', 'short_passing', 'volleys', 'dribbling', 'curve', \
                  'free_kick_accuracy', 'long_passing', 'ball_control', 'acceleration', \
                  'sprint_speed', 'agility', 'reactions', 'balance', 'shot_power', 'jumping', \
                  'stamina', 'strength', 'long_shots', 'aggression', 'interceptions', 'positioning', \
                  'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle', 'gk_diving', \
                  'gk_handling', 'gk_kicking', 'gk_positioning', 'gk_reflexes' ]
numeric_few_col = [ 'age', 'height', 'weight','overall_rating', 'potential', 'total_attack', 'total_skill', \
                  'total_movement', 'total_power', 'total_mentality', 'total_defending', 'total_goalkeeping' ]
```

=====

Tables in the database

df_tables shape: (8, 5)

	type	name	tbl_name	rootpage	\
0	table	sqlite_sequence	sqlite_sequence	4	
1	table	Player_Attributes	Player_Attributes	11	
2	table	Player	Player	14	
3	table	Match	Match	18	
4	table	League	League	24	
5	table	Country	Country	26	
6	table	Team	Team	29	
7	table	Team_Attributes	Team_Attributes	2	

sql

```
0      CREATE TABLE sqlite_sequence(name,seq)
1 CREATE TABLE "Player_Attributes" (\n\t`id`\tINTEGER PRIMA...
2 CREATE TABLE `Player` (\n\t`id`\tINTEGER PRIMA...
3 CREATE TABLE `Match` (\n\t`id`\tINTEGER PRIMAR...
4 CREATE TABLE `League` (\n\t`id`\tINTEGER PRIMA...
5 CREATE TABLE `Country` (\n\t`id`\tINTEGER PRIM...
6 CREATE TABLE "Team" (\n\t`id`\tINTEGER PRIMARY...
7 CREATE TABLE `Team_Attributes` (\n\t`id`\tINTE...
```

=====

Player_Attributes table:

```
CREATE TABLE "Player_Attributes" (
    `id`      INTEGER PRIMARY KEY AUTOINCREMENT,
    `player_fifa_api_id`    INTEGER,
    `player_api_id`  INTEGER,
    `date`     TEXT,
    `overall_rating`        INTEGER,
    `potential`    INTEGER,
    `preferred_foot`       TEXT,
```

```

`attacking_work_rate`    TEXT,
`defensive_work_rate`   TEXT,
`crossing`      INTEGER,
`finishing`     INTEGER,
`heading_accuracy`   INTEGER,
`short_passing`   INTEGER,
`volleys`        INTEGER,
`dribbling`      INTEGER,
`curve`          INTEGER,
`free_kick_accuracy` INTEGER,
`long_passing`   INTEGER,
`ball_control`    INTEGER,
`acceleration`   INTEGER,
`sprint_speed`    INTEGER,
`agility`         INTEGER,
`reactions`       INTEGER,
`balance`         INTEGER,
`shot_power`      INTEGER,
`jumping`         INTEGER,
`stamina`        INTEGER,
`strength`        INTEGER,
`long_shots`      INTEGER,
`aggression`     INTEGER,
`interceptions`  INTEGER,
`positioning`    INTEGER,
`vision`          INTEGER,
`penalties`       INTEGER,
`marking`         INTEGER,
`standing_tackle` INTEGER,
`sliding_tackle`  INTEGER,
`gk_diving`       INTEGER,
`gk_handling`    INTEGER,
`gk_kicking`     INTEGER,
`gk_positioning` INTEGER,
`gk_reflexes`    INTEGER,
FOREIGN KEY(`player_fifa_api_id`) REFERENCES `Player`(`player_fifa
_api_id`),
FOREIGN KEY(`player_api_id`) REFERENCES `Player`(`player_api_id`)
)
=====

Player table:
CREATE TABLE `Player` (
    `id`      INTEGER PRIMARY KEY AUTOINCREMENT,
    `player_api_id` INTEGER UNIQUE,
    `player_name`   TEXT,
    `player_fifa_api_id` INTEGER UNIQUE,
    `birthday`    TEXT,
    `height`      INTEGER,
    `weight`      INTEGER
)
=====

Player table
df_Player.shape: (11060, 7)
Index(['id', 'player_api_id', 'player_name', 'player_fifa_api_id', 'birthd
ay',
       'height', 'weight'],
      dtype='object')

```

```

      id  player_api_id      player_name  player_fifa_api_id \
0    1          505942   Aaron Appindangoye           218353
1    2          155782   Aaron Cresswell            189615
2    3          162549     Aaron Doran             186170
3    4          30572    Aaron Galindo            140161
4    5          23780    Aaron Hughes             17725

              birthday  height  weight
0  1992-02-29 00:00:00  182.88    187
1  1989-12-15 00:00:00  170.18    146
2  1991-05-13 00:00:00  170.18    163
3  1982-05-08 00:00:00  182.88    198
4  1979-11-08 00:00:00  182.88    154
=====

Player_Attributes table
df_Player_Attributes.shape: (183978, 42)
Index(['id', 'player_fifa_api_id', 'player_api_id', 'date', 'overall_rating',
       'potential', 'preferred_foot', 'attacking_work_rate',
       'defensive_work_rate', 'crossing', 'finishing', 'heading_accuracy',
       'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_accuracy',
       'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
       'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'stamina',
       'strength', 'long_shots', 'aggression', 'interceptions', 'positioning',
       'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle',
       'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
       'gk_reflexes'],
      dtype='object')
      id  player_fifa_api_id  player_api_id          date  overall_rating \
0    1          218353        505942  2016-02-18 00:00:00               6
7.0
1    2          218353        505942  2015-11-19 00:00:00               6
7.0
2    3          218353        505942  2015-09-21 00:00:00               6
2.0
3    4          218353        505942  2015-03-20 00:00:00               6
1.0
4    5          218353        505942  2007-02-22 00:00:00               6
1.0

      potential  preferred_foot  attacking_work_rate  defensive_work_rate  crossing \
0         71.0          right                medium                 medium        4
9.0
1         71.0          right                medium                 medium        4
9.0
2         66.0          right                medium                 medium        4
9.0
3         65.0          right                medium                 medium        4
8.0
4         65.0          right                medium                 medium        4
8.0

```

```

...           vision  penalties  marking  standing_tackle  sliding_tackl
e \
0   ...        54.0       48.0      65.0            69.0       69.
0
1   ...        54.0       48.0      65.0            69.0       69.
0
2   ...        54.0       48.0      65.0            66.0       69.
0
3   ...        53.0       47.0      62.0            63.0       66.
0
4   ...        53.0       47.0      62.0            63.0       66.
0

gk_diving  gk_handling  gk_kicking  gk_positioning  gk_reflexes
0          6.0         11.0        10.0          8.0        8.0
1          6.0         11.0        10.0          8.0        8.0
2          6.0         11.0        10.0          8.0        8.0
3          5.0         10.0        9.0           7.0        7.0
4          5.0         10.0        9.0           7.0        7.0

[5 rows x 42 columns]
=====
df_all_col.columns: Index(['id', 'player_api_id', 'player_name', 'player_fifa_api_id', 'birthday',
                           'height', 'weight', 'id', 'player_fifa_api_id', 'player_api_id', 'date',
                           'overall_rating', 'potential', 'preferred_foot', 'attacking_work_rate',
                           'defensive_work_rate', 'crossing', 'finishing', 'heading_accuracy',
                           'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_accuracy',
                           'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
                           'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'stamina',
                           'strength', 'long_shots', 'aggression', 'interceptions', 'positioning',
                           'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle',
                           'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
                           'gk_reflexes', 'age', 'total_attack', 'total_skill', 'total_movement',
                           'total_power', 'total_mentality', 'total_defending',
                           'total_goalkeeping'],
                           dtype='object')
df_all_col.shape: (183929, 57)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 183929 entries, 0 to 183928
Data columns (total 57 columns):
id                  183929 non-null int64
player_api_id       183929 non-null int64
player_name         183929 non-null object
player_fifa_api_id 183929 non-null int64
birthday           183929 non-null datetime64[ns]
height              183929 non-null float64
weight              183929 non-null int64
id                  183929 non-null int64

```

```

player_fifa_api_id      183929 non-null int64
player_api_id           183929 non-null int64
date                   183929 non-null datetime64[ns]
overall_rating          183142 non-null float64
potential              183142 non-null float64
preferred_foot         183142 non-null object
attacking_work_rate    180748 non-null object
defensive_work_rate   183142 non-null object
crossing               183142 non-null float64
finishing              183142 non-null float64
heading_accuracy       183142 non-null float64
short_passing          183142 non-null float64
volleys                181265 non-null float64
dribbling              183142 non-null float64
curve                  181265 non-null float64
free_kick_accuracy    183142 non-null float64
long_passing          183142 non-null float64
ball_control           183142 non-null float64
acceleration          183142 non-null float64
sprint_speed           183142 non-null float64
agility                 181265 non-null float64
reactions              183142 non-null float64
balance                181265 non-null float64
shot_power             183142 non-null float64
jumping                181265 non-null float64
stamina                183142 non-null float64
strength               183142 non-null float64
long_shots             183142 non-null float64
aggression              183142 non-null float64
interceptions          183142 non-null float64
positioning             183142 non-null float64
vision                  181265 non-null float64
penalties              183142 non-null float64
marking                 183142 non-null float64
standing_tackle        183142 non-null float64
sliding_tackle         181265 non-null float64
gk_diving              183142 non-null float64
gk_handling             183142 non-null float64
gk_kicking              183142 non-null float64
gk_positioning          183142 non-null float64
gk_reflexes             183142 non-null float64
age                     183929 non-null float64
total_attack            181265 non-null float64
total_skill              181265 non-null float64
total_movement          181265 non-null float64
total_power              181265 non-null float64
total_mentality          181265 non-null float64
total_defending          181265 non-null float64
total_goalkeeping        183142 non-null float64
dtypes: datetime64[ns](2), float64(44), int64(7), object(4)
memory usage: 80.0+ MB
None

```

data cleaning

```
In [7]: df_all_col.replace(r'\s+', np.nan, regex=True, inplace = True)
df_all_col.dropna(axis=0, how='any', inplace=True) #drop row (sample) with any NA entry
df_all_col.sort_values('player_name',axis=0, inplace=True)
df_all_col.drop_duplicates(inplace=True)
df_all_col.to_csv('df_all_col.csv')

print ('df_all_col.shape: ', df_all_col.shape)
print(df_all_col.shape)
print(df_all_col.info())
print(df_all_col.head())
print(df_all_col.tail())
print(df_all_col['defensive_work_rate'][0:60]) # need more data cleaning for col before plotting
print ('=====')
```

df_unscaled_data = df_all_col[numERIC_col]

```
print('df_unscaled_data.columns:', df_unscaled_data.columns)
print('df_unscaled_data.shape:', df_unscaled_data.shape)
print('df_unscaled_data.info:', df_unscaled_data.info())
```

```
df_all_col.shape: (10898, 57)
(10898, 57)
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10898 entries, 1045 to 183615
Data columns (total 57 columns):
id                10898 non-null int64
player_api_id     10898 non-null int64
player_name       10898 non-null object
player_fifa_api_id 10898 non-null int64
birthday          10898 non-null datetime64[ns]
height             10898 non-null float64
weight             10898 non-null int64
id                10898 non-null int64
player_fifa_api_id 10898 non-null int64
player_api_id     10898 non-null int64
date               10898 non-null datetime64[ns]
overall_rating    10898 non-null float64
potential          10898 non-null float64
preferred_foot    10898 non-null object
attacking_work_rate 10898 non-null object
defensive_work_rate 10898 non-null object
crossing           10898 non-null float64
finishing          10898 non-null float64
heading_accuracy   10898 non-null float64
short_passing      10898 non-null float64
volleys            10898 non-null float64
dribbling          10898 non-null float64
curve              10898 non-null float64
free_kick_accuracy 10898 non-null float64
long_passing       10898 non-null float64
ball_control       10898 non-null float64
acceleration       10898 non-null float64
sprint_speed        10898 non-null float64
agility             10898 non-null float64
reactions           10898 non-null float64
balance            10898 non-null float64
```

```

shot_power           10898 non-null float64
jumping             10898 non-null float64
stamina             10898 non-null float64
strength            10898 non-null float64
long_shots          10898 non-null float64
aggression          10898 non-null float64
interceptions       10898 non-null float64
positioning         10898 non-null float64
vision              10898 non-null float64
penalties            10898 non-null float64
marking              10898 non-null float64
standing_tackle      10898 non-null float64
sliding_tackle       10898 non-null float64
gk_diving            10898 non-null float64
gk_handling          10898 non-null float64
gk_kicking            10898 non-null float64
gk_positioning        10898 non-null float64
gk_reflexes           10898 non-null float64
age                  10898 non-null float64
total_attack          10898 non-null float64
total_skill            10898 non-null float64
total_movement         10898 non-null float64
total_power             10898 non-null float64
total_mentality          10898 non-null float64
total_defending          10898 non-null float64
total_goalkeeping         10898 non-null float64
dtypes: datetime64[ns](2), float64(44), int64(7), object(4)
memory usage: 4.8+ MB
None
      id  player_api_id player_name  player_fifa_api_id    birthday   height
      \
1045  67            40938      Abel                 17880 1978-12-22    177.8
1046  67            40938      Abel                 17880 1978-12-22    177.8
1047  67            40938      Abel                 17880 1978-12-22    177.8
1048  67            40938      Abel                 17880 1978-12-22    177.8
1049  67            40938      Abel                 17880 1978-12-22    177.8

      weight     id  player_fifa_api_id  player_api_id ...
      \
1045     165  1046                  17880            40938 ...
1046     165  1047                  17880            40938 ...
1047     165  1048                  17880            40938 ...
1048     165  1049                  17880            40938 ...
1049     165  1050                  17880            40938 ...

      gk_positioning  gk_reflexes    age total_attack total_skill \
1045            13.0         10.0  31.0        311.0        319.0

```

1046	20.0	20.0	31.0	307.0	318.0	
1047	20.0	20.0	30.0	305.0	316.0	
1048	20.0	20.0	28.0	298.0	311.0	
1049	8.0	12.0	28.0	288.0	335.0	
	total_movement	total_power	total_mentality	total_defending	\	
1045	375.0	355.0	351.0	219.0		
1046	375.0	355.0	372.0	216.0		
1047	377.0	357.0	372.0	221.0		
1048	372.0	355.0	370.0	221.0		
1049	366.0	346.0	356.0	212.0		
	total_goalkeeping					
1045	42.0					
1046	143.0					
1047	143.0					
1048	141.0					
1049	102.0					
[5 rows x 57 columns]						
	id	player_api_id	player_name	player_fifa_api_id	birthday	\
183611	11055	532766	Zizo	222537	1996-01-10	
183612	11055	532766	Zizo	222537	1996-01-10	
183613	11055	532766	Zizo	222537	1996-01-10	
183614	11055	532766	Zizo	222537	1996-01-10	
183615	11055	532766	Zizo	222537	1996-01-10	
	height	weight	id	player_fifa_api_id	player_api_id	\
183611	175.26	148	183661	222537	532766	
183612	175.26	148	183662	222537	532766	
183613	175.26	148	183663	222537	532766	
183614	175.26	148	183664	222537	532766	
183615	175.26	148	183665	222537	532766	
	gk_positioning	gk_reflexes	age	total_attack
\						
183611	...		11.0	8.0	19.0	250.0
183612	...		11.0	8.0	18.0	248.0
183613	...		11.0	8.0	18.0	247.0
183614	...		11.0	8.0	18.0	247.0
183615	...		11.0	8.0	11.0	247.0
	total_skill	total_movement	total_power	total_mentality	\	
183611	297.0	337.0	253.0	285.0		
183612	296.0	334.0	238.0	271.0		
183613	281.0	335.0	235.0	233.0		
183614	281.0	335.0	235.0	233.0		
183615	281.0	335.0	235.0	233.0		
	total_defending	total_goalkeeping				
183611	148.0		43.0			
183612	144.0		43.0			

183613	101.0	43.0
183614	101.0	43.0
183615	101.0	43.0

[5 rows x 57 columns]

1045	o
1046	o
1047	o
1048	o
1049	o
1315	medium
1314	medium
1306	medium
1307	medium
1308	medium
1309	medium
1310	medium
1311	medium
1316	medium
1317	medium
1318	medium
1319	medium
1320	medium
1321	medium
1322	medium
1313	medium
1312	medium
2444	medium
2445	medium
2443	medium
2446	medium
2447	medium
2450	medium
2448	medium
2449	medium
2541	medium
2542	medium
2543	medium
2544	medium
2545	medium
2546	medium
2547	medium
3145	medium
3152	medium
3141	medium
3142	medium
3143	medium
3144	medium
3151	medium
3146	medium
3147	medium
3148	medium
3140	medium
3150	medium
3139	medium
3149	medium
3154	medium

```

3156    medium
3165    medium
3153    medium
3157    medium
3158    medium
3159    medium
3160    medium
3155    medium
Name: defensive_work_rate, dtype: object
=====
df_unscaled_data.columns: Index(['age', 'height', 'weight', 'overall_rating',
       'potential', 'crossing',
       'finishing', 'heading_accuracy', 'short_passing', 'volleys',
       'dribbling', 'curve', 'free_kick_accuracy', 'long_passing',
       'ball_control', 'acceleration', 'sprint_speed', 'agility', 'reactions',
       'balance', 'shot_power', 'jumping', 'stamina', 'strength', 'long_shots',
       'aggression', 'interceptions', 'positioning', 'vision', 'penalties',
       'marking', 'standing_tackle', 'sliding_tackle', 'gk_diving',
       'gk_handling', 'gk_kicking', 'gk_positioning', 'gk_reflexes'],
      dtype='object')
df_unscaled_data.shape: (10898, 38)
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10898 entries, 1045 to 183615
Data columns (total 38 columns):
age            10898 non-null float64
height         10898 non-null float64
weight          10898 non-null int64
overall_rating 10898 non-null float64
potential       10898 non-null float64
crossing        10898 non-null float64
finishing       10898 non-null float64
heading_accuracy 10898 non-null float64
short_passing   10898 non-null float64
volleys         10898 non-null float64
dribbling        10898 non-null float64
curve           10898 non-null float64
free_kick_accuracy 10898 non-null float64
long_passing    10898 non-null float64
ball_control    10898 non-null float64
acceleration    10898 non-null float64
sprint_speed    10898 non-null float64
agility          10898 non-null float64
reactions        10898 non-null float64
balance          10898 non-null float64
shot_power       10898 non-null float64
jumping          10898 non-null float64
stamina          10898 non-null float64
strength         10898 non-null float64
long_shots       10898 non-null float64
aggression       10898 non-null float64
interceptions    10898 non-null float64
positioning       10898 non-null float64
vision           10898 non-null float64
penalties         10898 non-null float64

```

```

marking           10898 non-null float64
standing_tackle  10898 non-null float64
sliding_tackle   10898 non-null float64
gk_diving        10898 non-null float64
gk_handling      10898 non-null float64
gk_kicking       10898 non-null float64
gk_positioning   10898 non-null float64
gk_reflexes      10898 non-null float64
dtypes: float64(37), int64(1)
memory usage: 3.2 MB
df_unscaled_data.info: None

```

Preprocess data and conduct PCA - principal component analysis

```

In [9]: scaled_data = preprocessing.scale(df_unscaled_data) #center and scale the data
print('scaled data:')
print (scaled_data) # preview scaled data

# create a PCA object.
# sklean uses this PCA object that can be trained using one dataset and applied to another dataset
pca = PCA()
print(type(pca))
# do PCA math, calculate loading scores and the variation each PCA accounts for
pca.fit(scaled_data)
# generate coordinates for a PCA graph based on the loading scores and the scaled data
pca_data = pca.transform(scaled_data)

scaled data:
[[ 1.20435 -0.50782 -0.12955 ... -0.64338 -0.19821 -0.36672]
 [ 1.20435 -0.50782 -0.12955 ...  2.36822  0.20348  0.18149]
 [ 0.98217 -0.50782 -0.12955 ...  2.36822  0.20348  0.18149]
 ...
 [-1.68402 -0.88666 -1.24685 ... -0.64338 -0.31297 -0.47637]
 [-1.68402 -0.88666 -1.24685 ... -0.64338 -0.31297 -0.47637]
 [-3.2393 -0.88666 -1.24685 ... -0.64338 -0.31297 -0.47637]]
<class 'sklearn.decomposition.pca.PCA'>

```

Present Explained Variance, Scree Plot and Principal Components Scatter Matrix

```

In [11]: # pca.explained_variance_ratio_ is <class 'numpy.ndarray'>.
# It calculates the percentage of variance that each principal component accounts for
per_var = np.round(pca.explained_variance_ratio_*100, decimals =1)
print('=====')
print('percent of explained variance: ')
print(per_var)
PC_labels = [ 'PC'+ str(x) for x in range(1,len(per_var)+1)] # labels for t

```

```

# Scree Plot: PC1, PC2 ...
print(' ')
print('=====')

# create Scree Plot
plt.figure(figsize=(18, 6))
plt.bar(x=range(1, len(per_var)+1), height=per_var, tick_label=PC_labels)
plt.ylabel('Percentage of Explained Variance', fontsize='14')
plt.xlabel('Principal Component', fontsize='14')
plt.title('Scree Plot', fontsize='18')
plt.show()
plt.close()

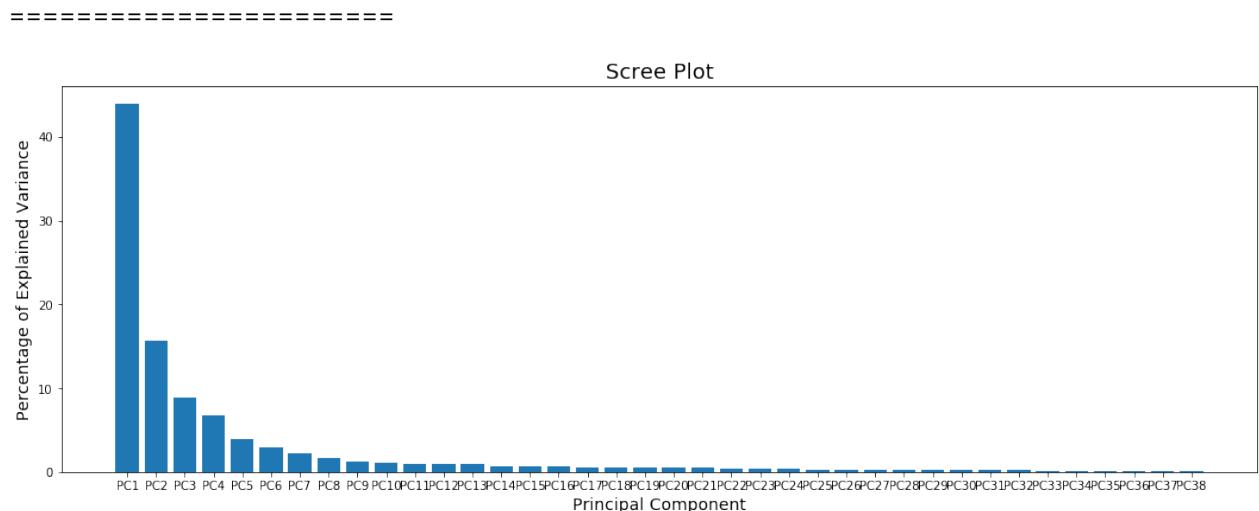
# put pca_data with DataFrame with PC_labels
pca_df = pd.DataFrame(pca_data, index=None, columns=PC_labels)
print(pca_df.head()) # preview transformed and scaled
print('=====')

print('Principal Components Scatter Matrix')
df_pc_matrix= pca_df[['PC'+ str(x) for x in range(1,21)]] # scatter matrix
for PC1, PC2, ..., PC15
pd.plotting.scatter_matrix(df_pc_matrix, alpha=0.1, figsize=(14, 14), diagonal='kde', range_padding =0.1)
plt.tight_layout()
plt.show()
plt.close()

```

```

=====
percent of explained variance:
[43.9 15.7 8.9 6.8 3.9 3. 2.2 1.7 1.3 1.1 1. 1. 0.9 0.7
 0.7 0.7 0.6 0.6 0.5 0.5 0.5 0.4 0.4 0.4 0.3 0.3 0.3 0.3
 0.2 0.2 0.2 0.2 0.1 0.1 0.1 0.1 0.1 0.1 0.1]
```



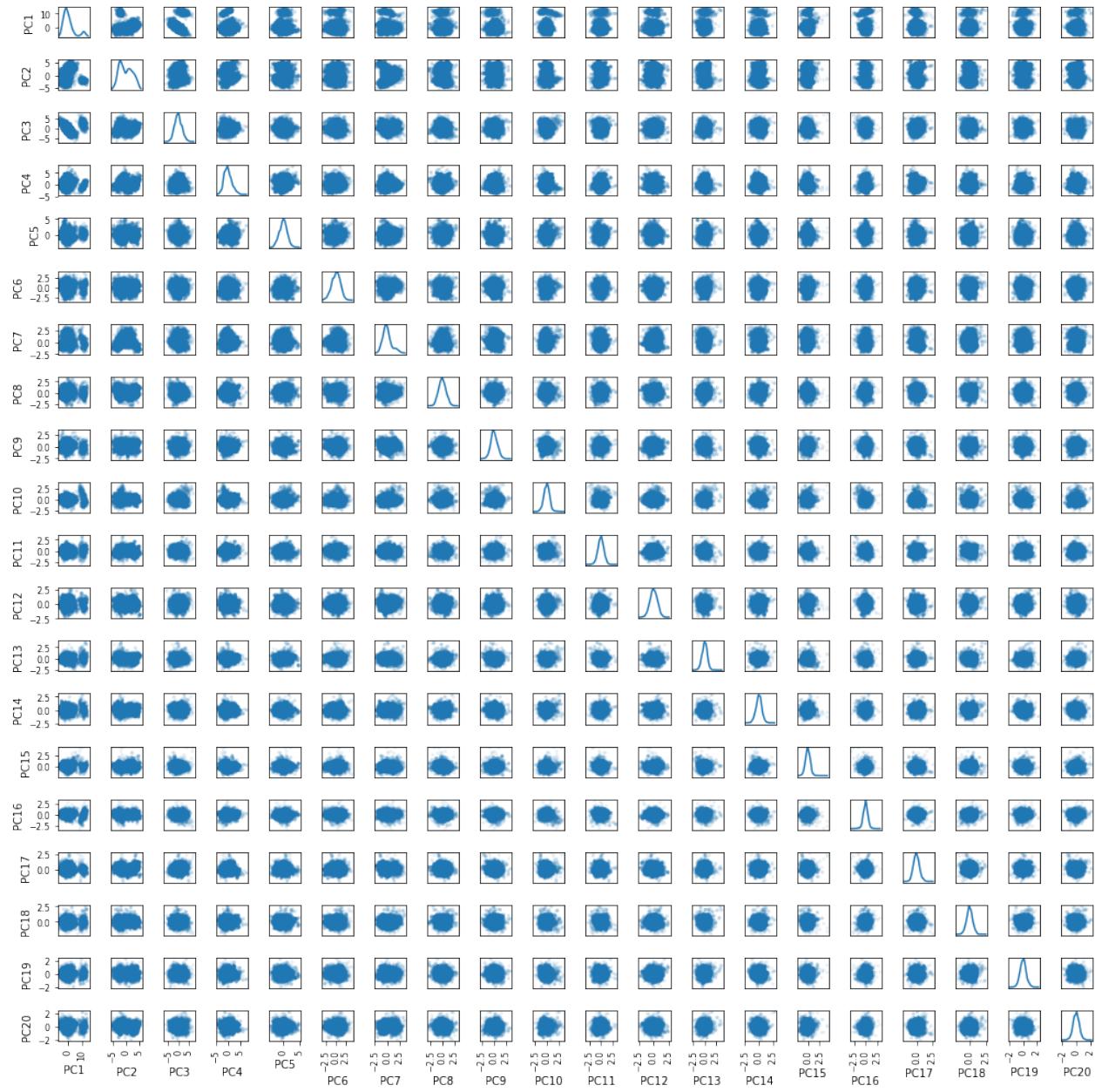
	PC1	PC2	PC3	PC4	PC5	PC6	PC7	\
0	-2.640114	1.968095	0.650296	-1.085030	0.432893	-0.997913	-0.036752	
1	-1.974144	1.547071	1.945571	-1.698880	0.785256	-0.431110	2.178955	
2	-1.998243	1.632706	1.964671	-1.818000	0.655679	-0.269227	2.168876	
3	-1.777293	1.508709	1.666837	-1.760435	0.469343	-0.019567	2.297794	
4	-2.056297	0.869395	0.466741	-1.179464	0.879790	-0.124301	2.485314	

	PC8	PC9	PC10	...	PC29	PC30	PC31	\
0	-0.345769	-0.212905	0.717640	...	-0.243024	-0.082784	-0.287740	
1	-0.524871	0.683914	0.138753	...	-0.313755	-0.136859	-0.142348	
2	-0.553466	0.642269	0.157086	...	-0.304040	0.008732	-0.030220	
3	-0.364947	0.543994	0.246732	...	-0.335973	0.075856	0.170031	
4	0.184533	0.201187	-0.439294	...	-0.316171	0.099965	0.113237	
	PC32	PC33	PC34	PC35	PC36	PC37	PC38	
0	-0.044248	-0.119196	-0.127400	-0.024028	0.054763	0.303422	0.202196	
1	-0.096422	-0.082920	0.120610	0.091425	-0.042255	0.005217	-0.040199	
2	-0.022378	-0.068867	0.080547	0.128276	0.035120	0.007889	-0.050469	
3	0.044900	-0.031943	0.052600	0.120342	0.048383	0.012409	-0.034910	
4	0.045786	-0.056746	0.502708	0.322174	-0.167724	-0.178028	-0.046963	

[5 rows x 38 columns]

=====

Principal Components Scatter Matrix



Take a Look at the Loading Scores of PC1, PC2 and PC3 to determine which feature has the largest influence in each principal component

```
In [12]: loading_scores_PC1 = pd.Series(pca.components_[0], index=numeric_col)
loading_scores_PC1_sorted = loading_scores_PC1.abs().sort_values(ascending=False)
print('Sorted PC1 Loading Scores (abs)')
print('PC1 sorted components: ', loading_scores_PC1_sorted.index)
print(loading_scores_PC1_sorted)
print('=====')

loading_scores_PC2 = pd.Series(pca.components_[1], index=numeric_col)
loading_scores_PC2_sorted = loading_scores_PC2.abs().sort_values(ascending=False)
print('Sorted PC2 Loading Scores (abs)')
print('PC2 sorted components: ', loading_scores_PC2_sorted.index)
print(loading_scores_PC2_sorted)
print('=====')

loading_scores_PC3 = pd.Series(pca.components_[2], index=numeric_col)
loading_scores_PC3_sorted = loading_scores_PC3.abs().sort_values(ascending=False)
print('Sorted PC3 Loading Scores (abs)')
print('PC3 sorted components: ', loading_scores_PC3_sorted.index)
print(loading_scores_PC3_sorted)
```

```
Sorted PC1 Loading Scores (abs)
PC1 sorted components: Index(['ball_control', 'dribbling', 'short_passing',
 'crossing', 'curve',
 'long_shots', 'positioning', 'shot_power', 'vision', 'gk_diving',
 'free_kick_accuracy', 'gk_reflexes', 'gk_handling', 'gk_positioning',
 'volleys', 'finishing', 'acceleration', 'penalties', 'sprint_speed',
 'long_passing', 'agility', 'stamina', 'balance', 'height',
 'heading_accuracy', 'weight', 'gk_kicking', 'reactions', 'aggressio
n',
 'overall_rating', 'potential', 'interceptions', 'standing_tackle',
 'sliding_tackle', 'marking', 'strength', 'age', 'jumping'],
 dtype='object')
ball_control          0.233895
dribbling            0.226609
short_passing        0.220168
crossing             0.213529
curve                0.211442
long_shots           0.211257
positioning          0.204851
shot_power           0.199655
vision               0.197614
gk_diving            0.196600
free_kick_accuracy   0.196366
gk_reflexes          0.195234
gk_handling          0.195011
gk_positioning       0.193662
volleys              0.193077
```

```

finishing          0.191773
acceleration      0.185480
penalties          0.183370
sprint_speed       0.182038
long_passing      0.177636
agility            0.168734
stamina           0.158548
balance            0.153811
height             0.134852
heading_accuracy   0.132158
weight             0.125473
gk_kicking         0.121982
reactions          0.105162
aggression         0.087644
overall_rating     0.083033
potential          0.080174
interceptions      0.071469
standing_tackle    0.065230
sliding_tackle     0.059237
marking            0.052163
strength           0.028138
age                0.005573
jumping            0.001561
dtype: float64
=====
Sorted PC2 Loading Scores (abs)
PC2 sorted components: Index(['marking', 'standing_tackle', 'sliding_tackle', 'interceptions',
                             'aggression', 'strength', 'heading_accuracy', 'agility', 'height',
                             'weight', 'finishing', 'stamina', 'volleys', 'balance',
                             'gk_positioning', 'gk_handling', 'acceleration', 'gk_reflexes',
                             'positioning', 'gk_diving', 'jumping', 'dribbling', 'age', 'curve',
                             'gk_kicking', 'sprint_speed', 'long_passing', 'penalties', 'long_shots',
                             'vision', 'short_passing', 'free_kick_accuracy', 'overall_rating',
                             'shot_power', 'crossing', 'reactions', 'potential', 'ball_control'],
                             dtype='object')
marking            0.360160
standing_tackle    0.357693
sliding_tackle     0.351805
interceptions      0.323485
aggression         0.310414
strength           0.269961
heading_accuracy   0.231016
agility            0.172402
height             0.171104
weight             0.151075
finishing          0.149702
stamina           0.122820
volleys            0.117505
balance            0.116588
gk_positioning     0.116035
gk_handling        0.111481
acceleration       0.109991
gk_reflexes        0.108521
positioning        0.108035

```

```

gk_diving           0.108023
jumping            0.103647
dribbling          0.086999
age                0.083970
curve              0.080749
gk_kicking          0.079368
sprint_speed        0.075460
long_passing       0.073430
penalties          0.069148
long_shots         0.061820
vision             0.061094
short_passing      0.060972
free_kick_accuracy 0.047318
overall_rating     0.043017
shot_power         0.028401
crossing           0.025229
reactions          0.019854
potential          0.011417
ball_control       0.011066
dtype: float64
=====
Sorted PC3 Loading Scores (abs)
PC3 sorted components: Index(['overall_rating', 'potential', 'reactions',
 'gk_kicking', 'gk_reflexes',
 'gk_positioning', 'gk_handling', 'age', 'gk_diving', 'strength',
 'weight', 'jumping', 'vision', 'height', 'long_passing', 'penalties',
 '',
 'shot_power', 'long_shots', 'free_kick_accuracy', 'volleys',
 'interceptions', 'short_passing', 'positioning', 'finishing', 'balance',
 'aggression', 'sliding_tackle', 'sprint_speed', 'marking',
 'acceleration', 'curve', 'standing_tackle', 'ball_control', 'stamina',
 'agility', 'crossing', 'heading_accuracy', 'dribbling'],
 dtype='object')
overall_rating      0.456070
potential          0.370266
reactions          0.361309
gk_kicking          0.263859
gk_reflexes         0.243520
gk_positioning      0.243479
gk_handling         0.242146
age                0.232184
gk_diving          0.222118
strength           0.169583
weight             0.150248
jumping            0.133864
vision             0.130572
height              0.128780
long_passing       0.127784
penalties          0.104386
shot_power         0.071653
long_shots         0.066604
free_kick_accuracy 0.062229
volleys            0.062204
interceptions      0.062061
short_passing      0.057178

```

```

positioning          0.056740
finishing           0.054228
balance             0.054159
aggression          0.051947
sliding_tackle      0.038316
sprint_speed         0.036239
marking              0.036117
acceleration        0.033763
curve                0.032351
standing_tackle     0.025389
ball_control         0.020478
stamina              0.017236
agility               0.011974
crossing             0.010881
heading_accuracy    0.006203
dribbling            0.001945
dtype: float64

```

Present graphs pertinent to the first three principal components:

Note: When plotting PC1 versus PC2 AND PC1 versus PC3, two clusters are displayed.

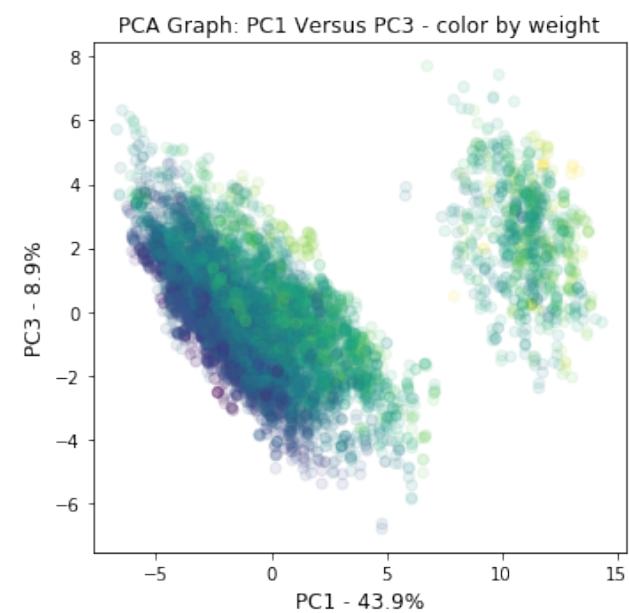
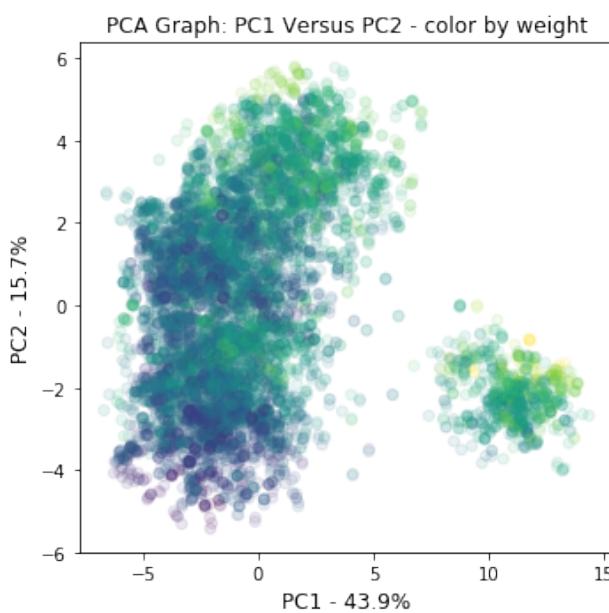
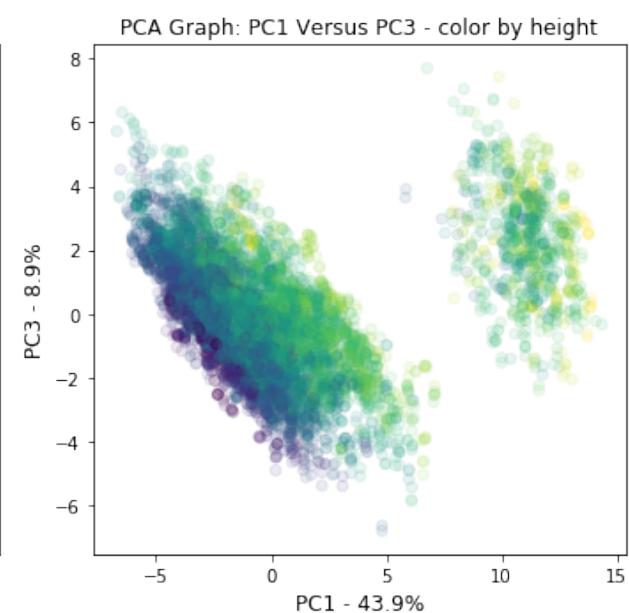
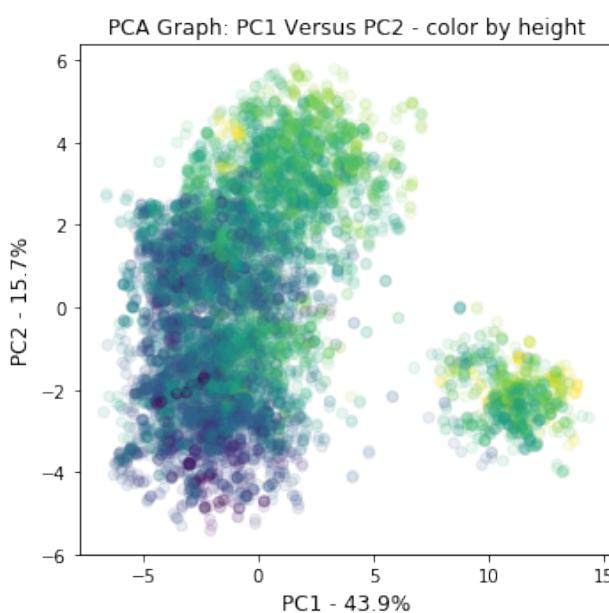
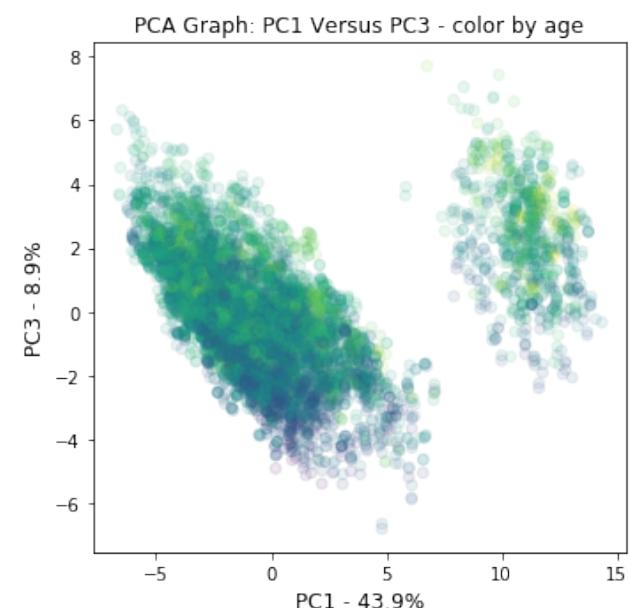
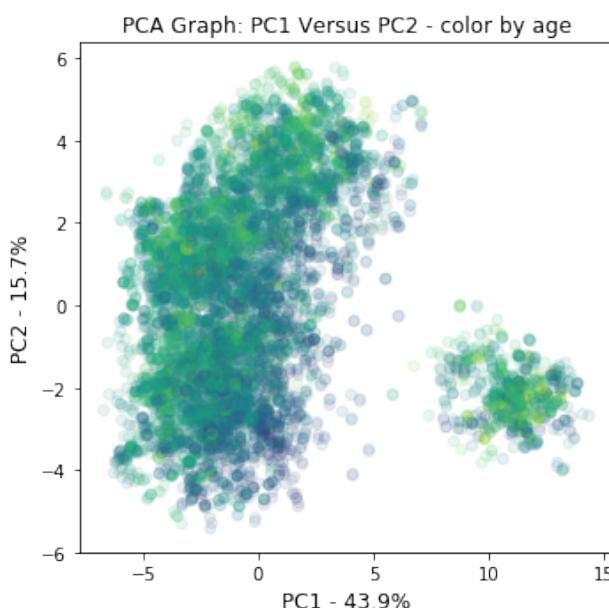
Next, we will visualize further in PC1, PC2 and PC3.

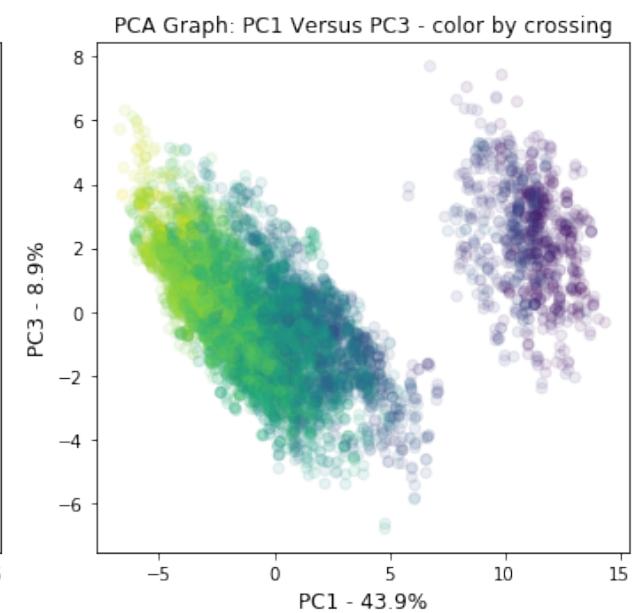
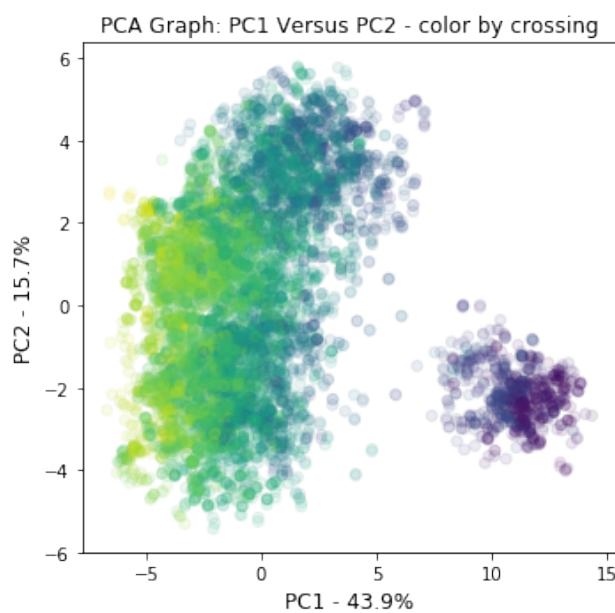
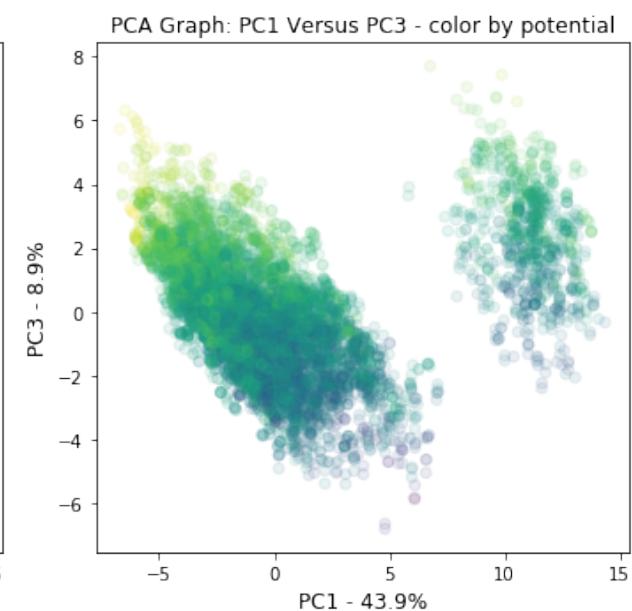
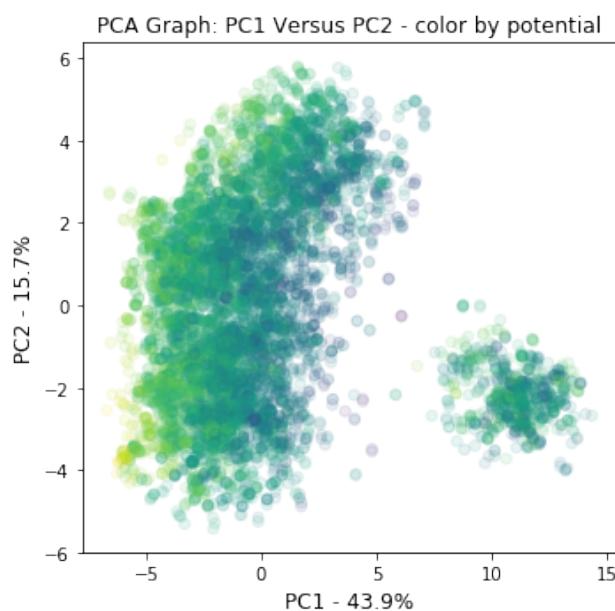
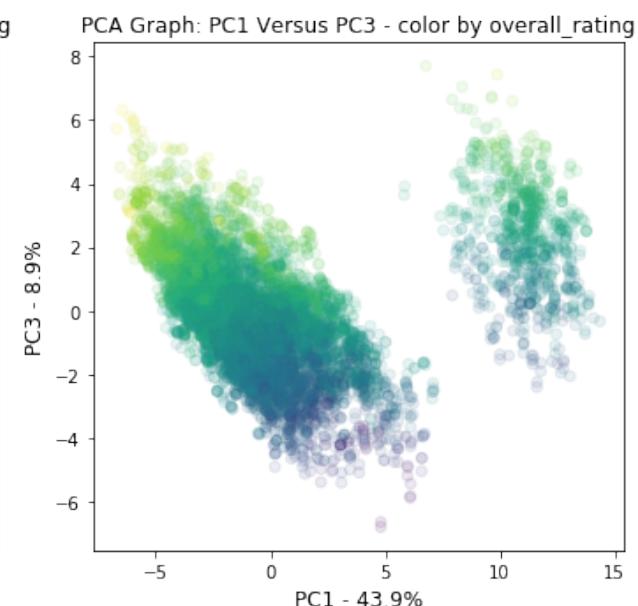
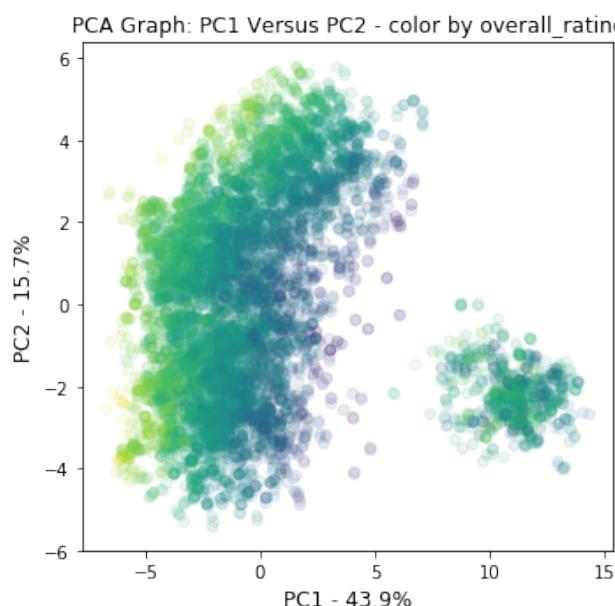
```
In [13]: # draw PCA 2D plot: PC1 Vs PC2 and PC1 Vs PC3
def color_plot (i):
    plt.figure(figsize=(10, 5))
    plt.subplot(1,2,1)

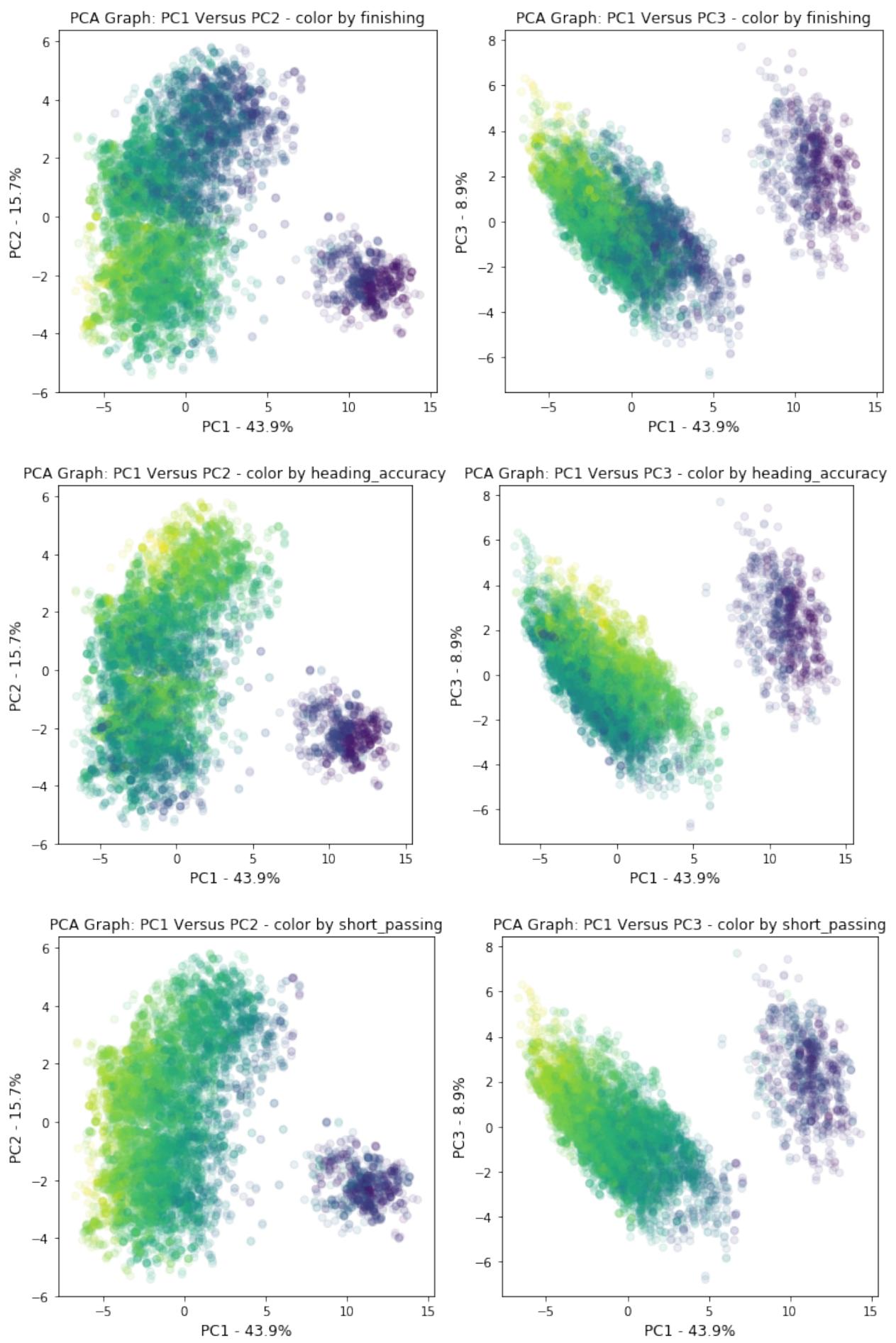
    plt.scatter(pca_df.PC1, pca_df.PC2, c=scaled_data[:,i], alpha=0.1)
    plt.title("PCA Graph: PC1 Versus PC2 - color by " + numeric_col[i], fontsize='12')
    plt.xlabel('PC1 - {0}%'.format(per_var[0]), fontsize='12')
    plt.ylabel('PC2 - {0}%'.format(per_var[1]), fontsize='12')
    plt.tight_layout()

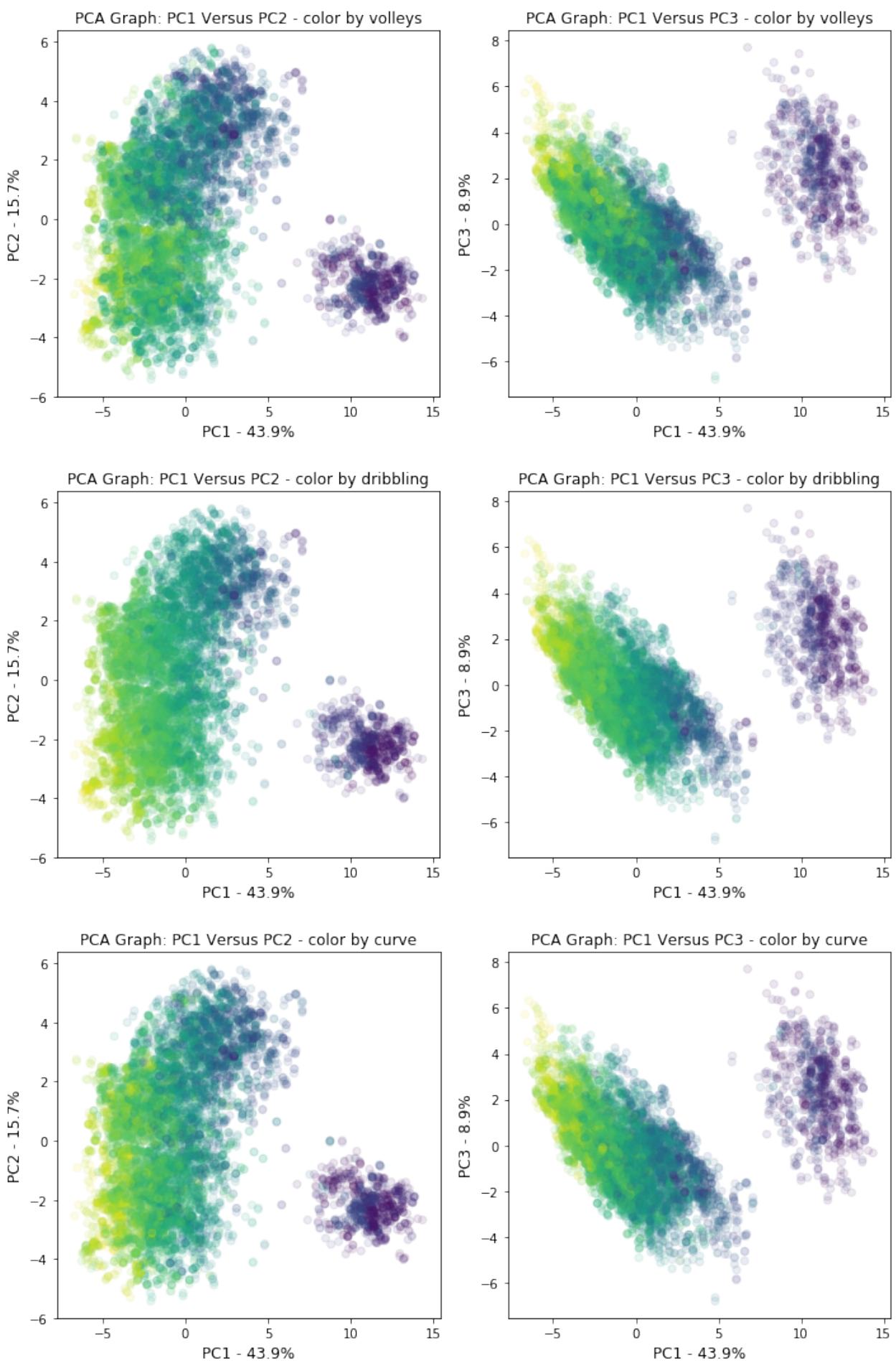
    plt.subplot(1,2,2)
    plt.scatter(pca_df.PC1, pca_df.PC3, c=scaled_data[:,i], alpha=0.1)
    plt.title("PCA Graph: PC1 Versus PC3 - color by " + numeric_col[i], fontsize='12')
    plt.xlabel('PC1 - {0}%'.format(per_var[0]), fontsize='12')
    plt.ylabel('PC3 - {0}%'.format(per_var[2]), fontsize='12')
    plt.tight_layout()
    plt.show()
    plt.close()

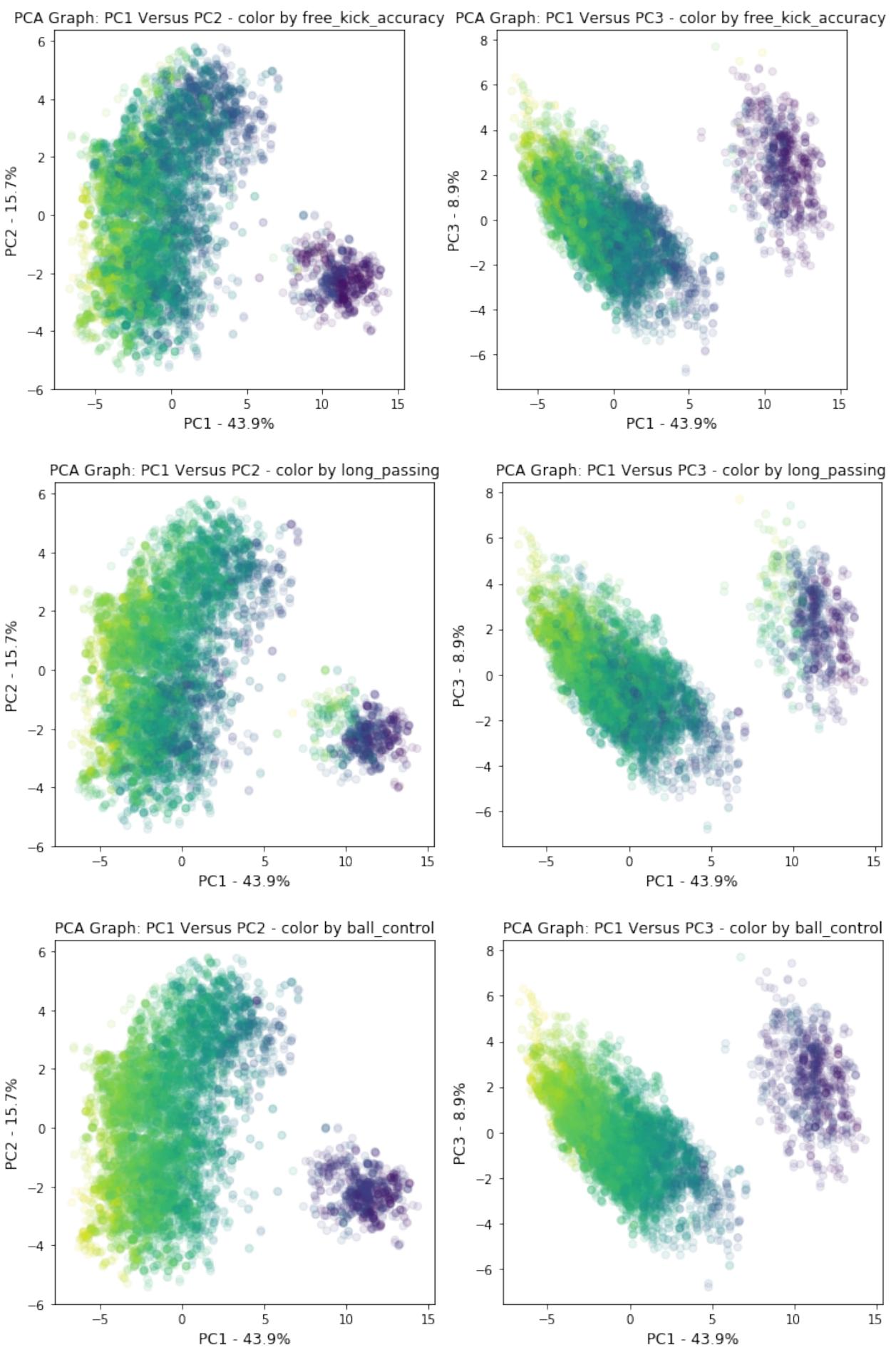
for j in range(0,38):
    color_plot (j)
```

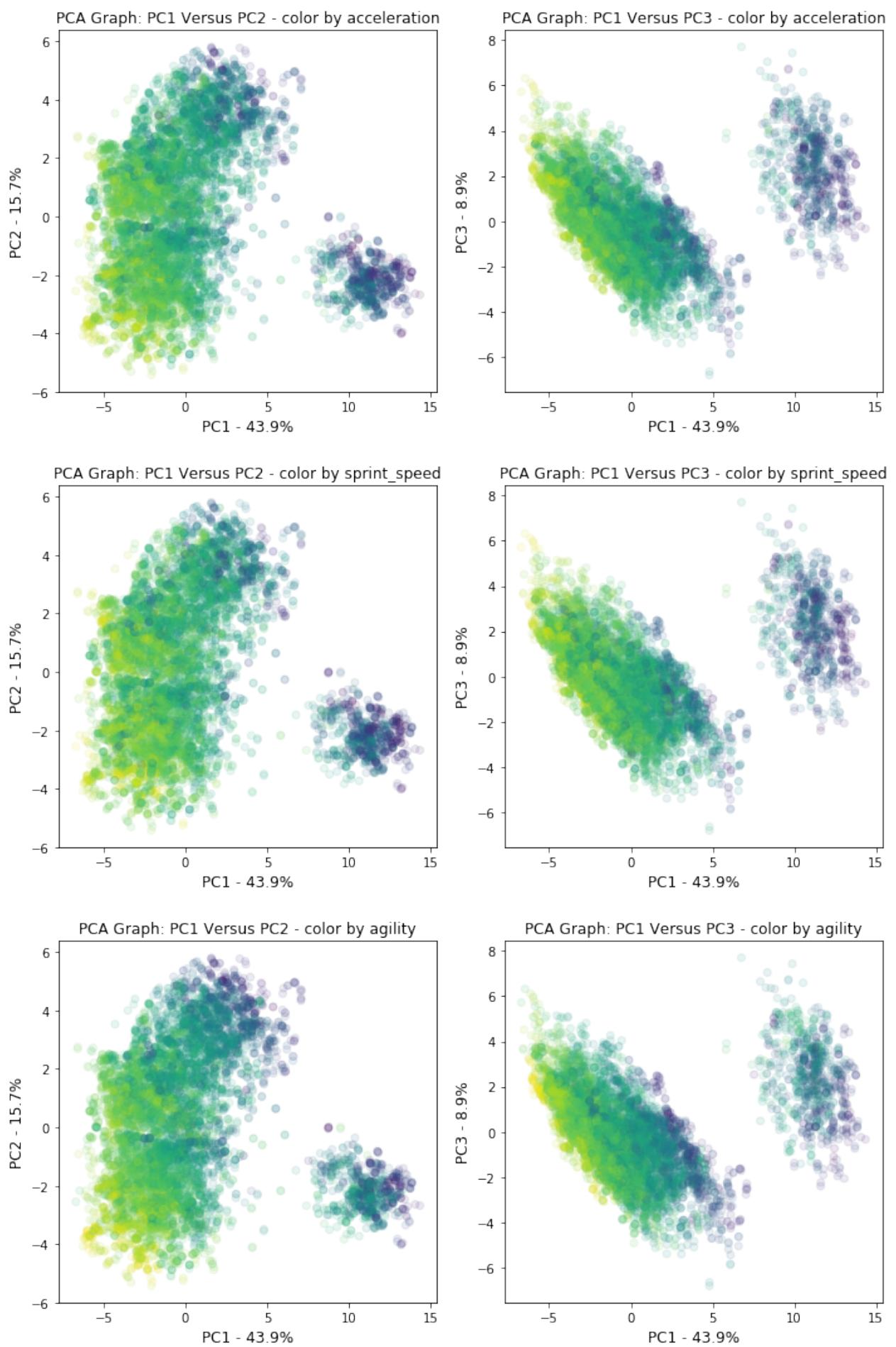


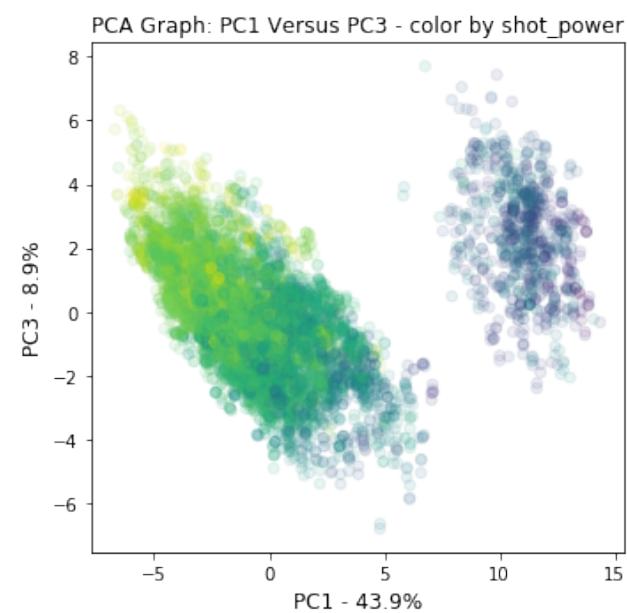
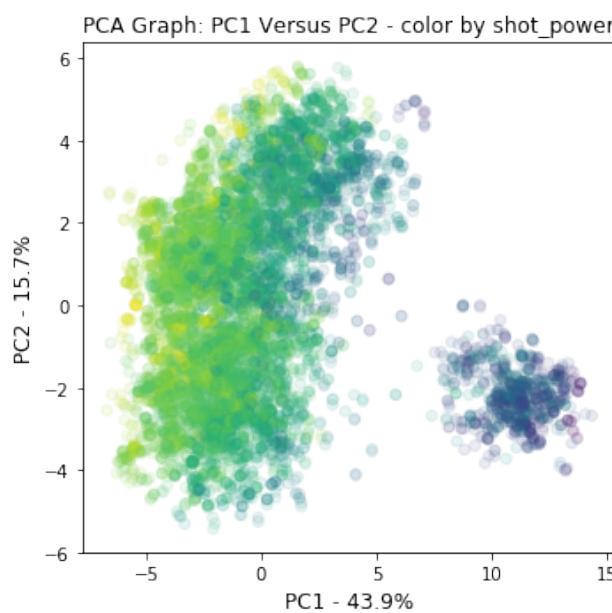
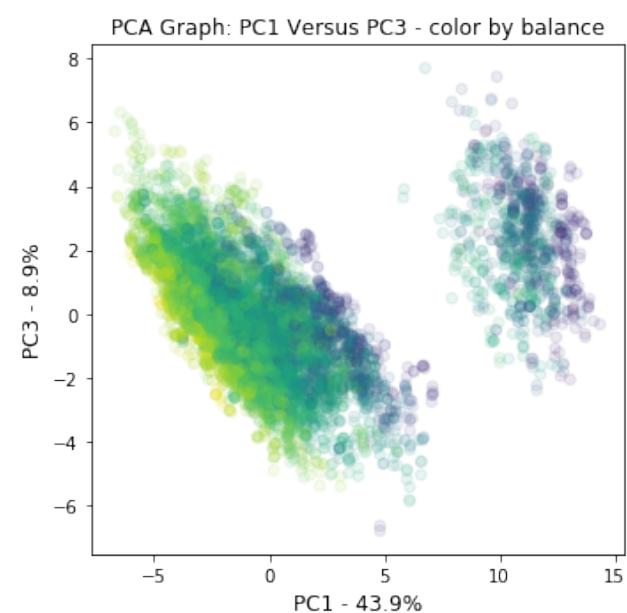
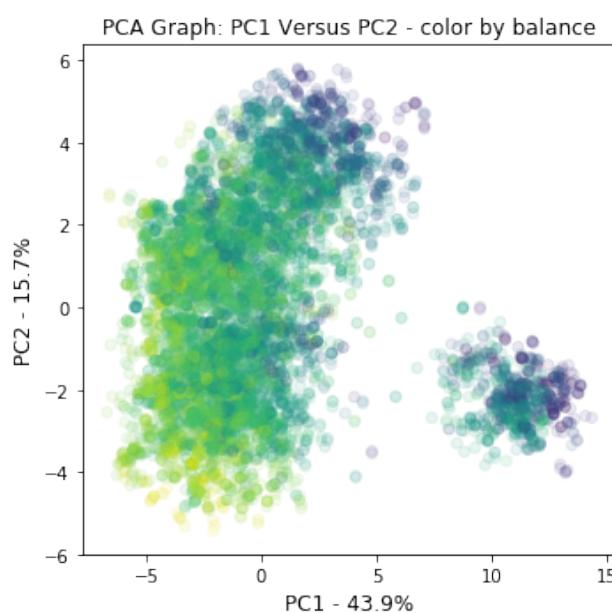
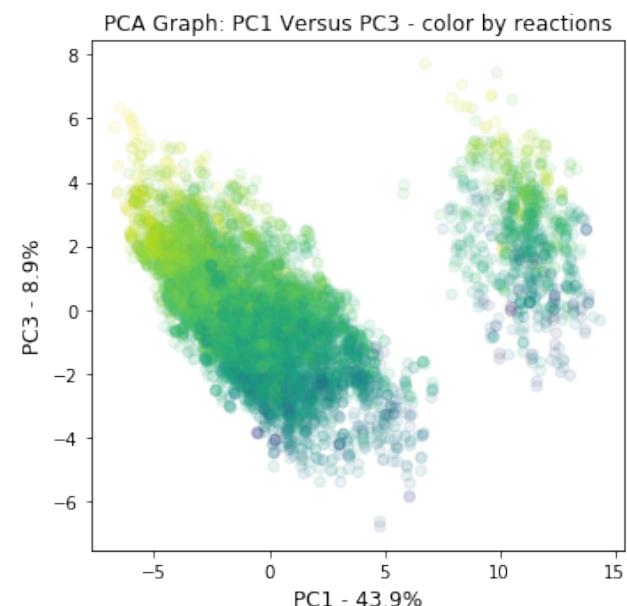
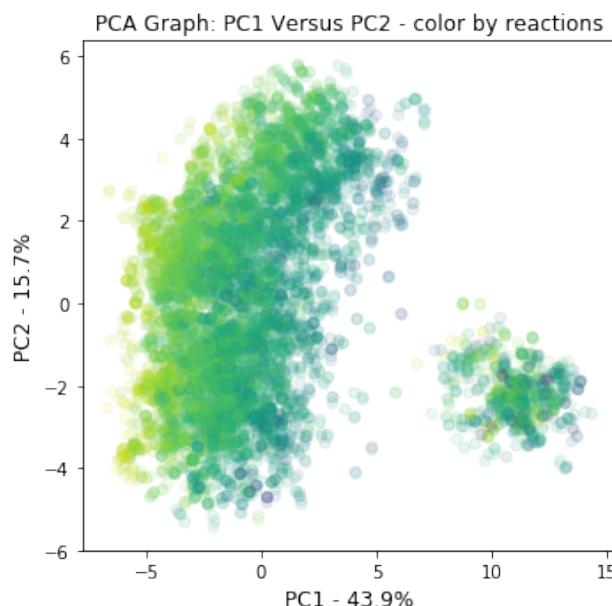


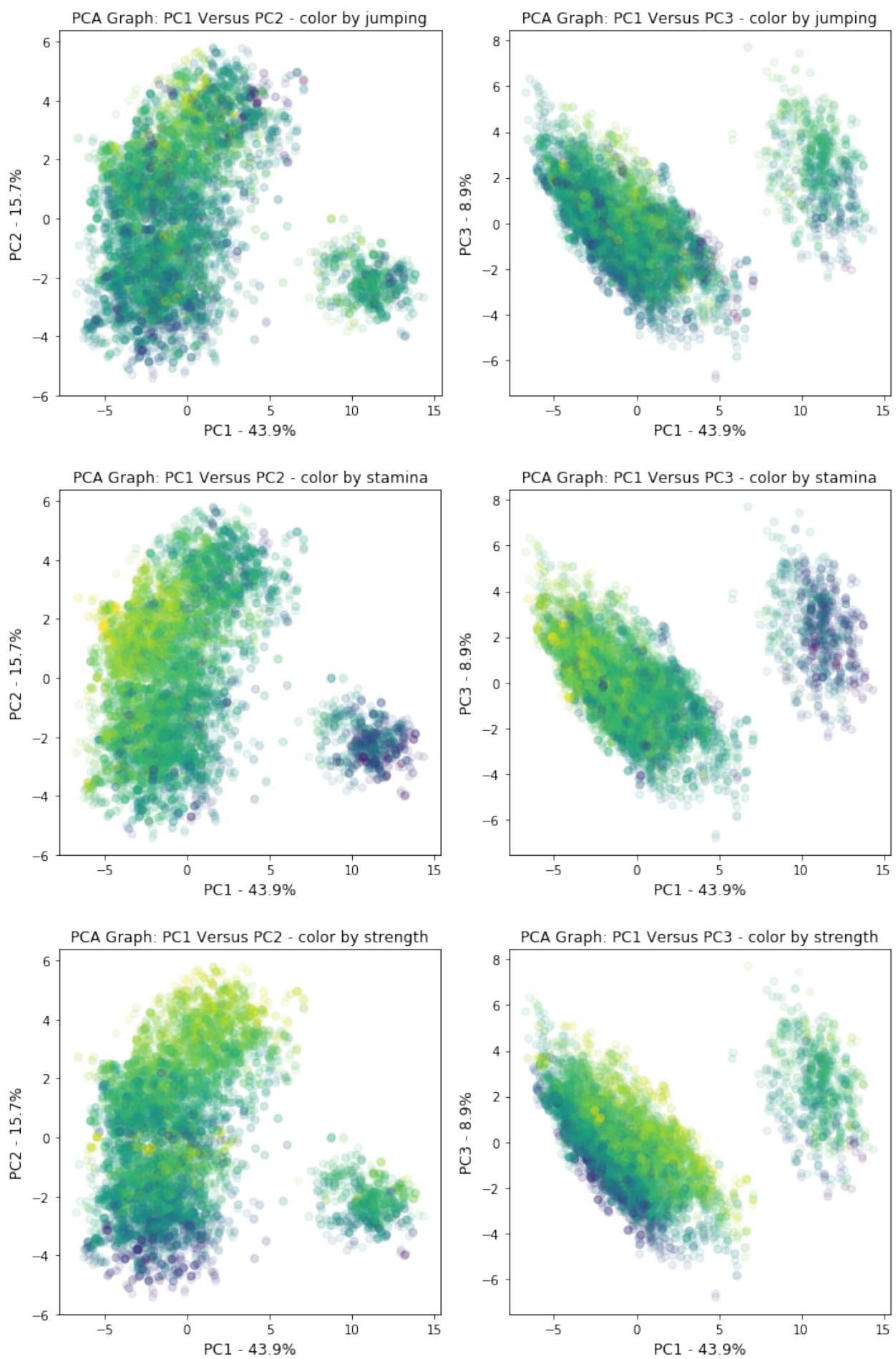


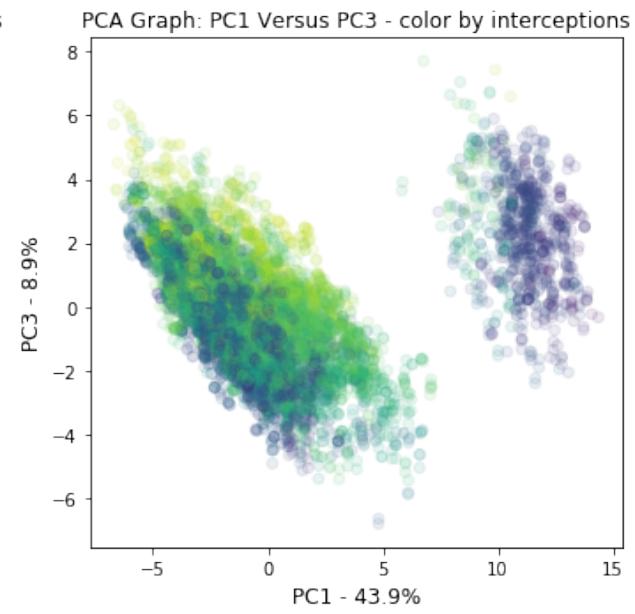
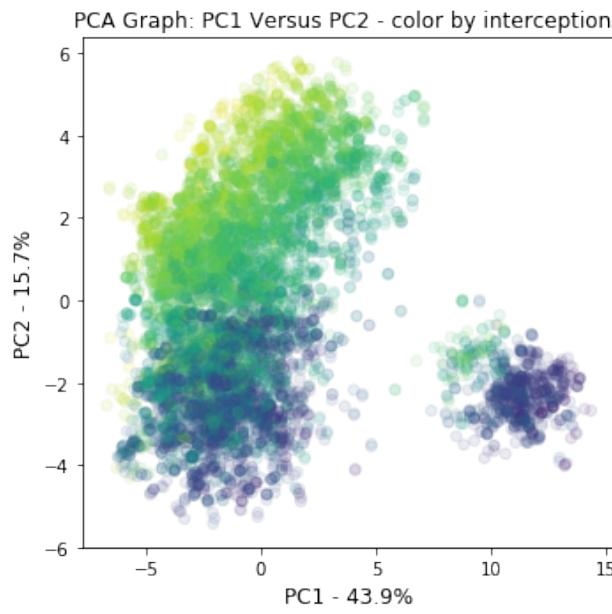
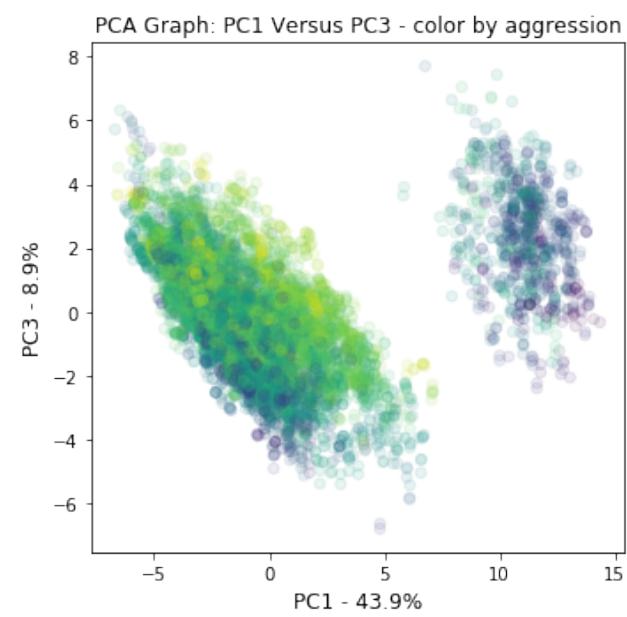
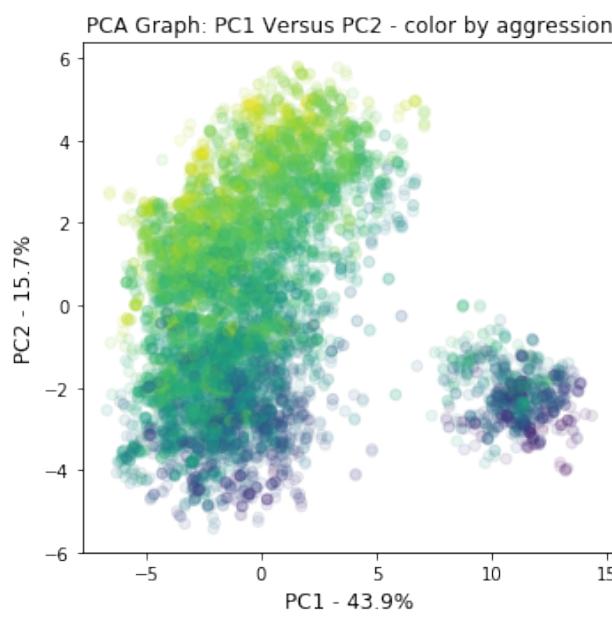
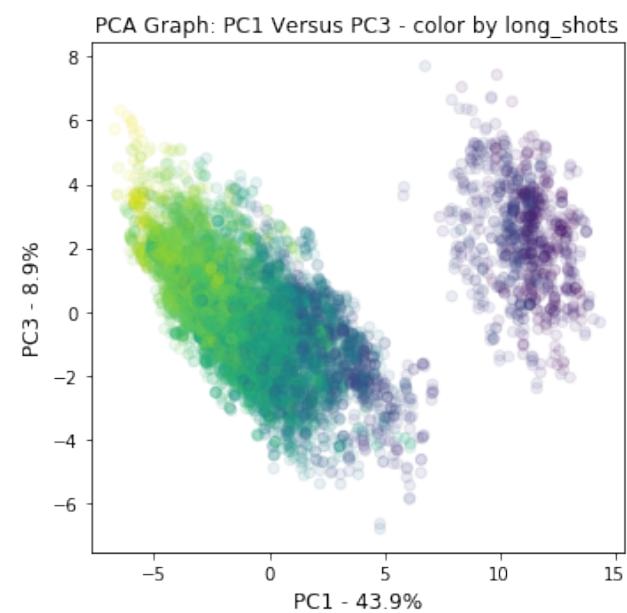
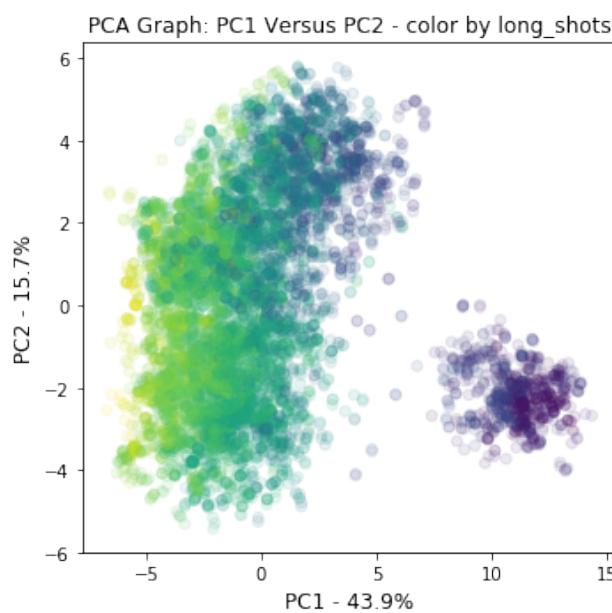


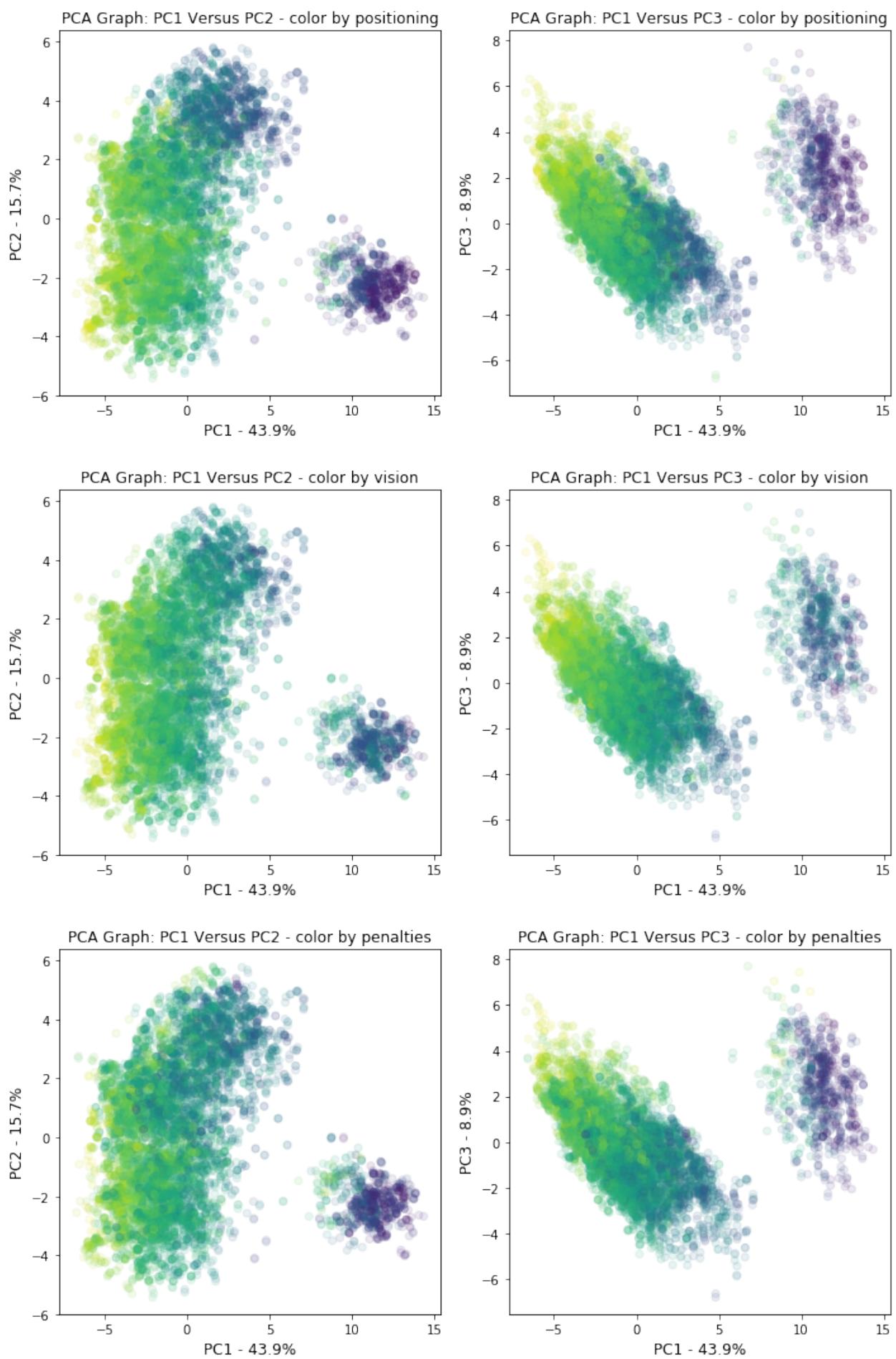


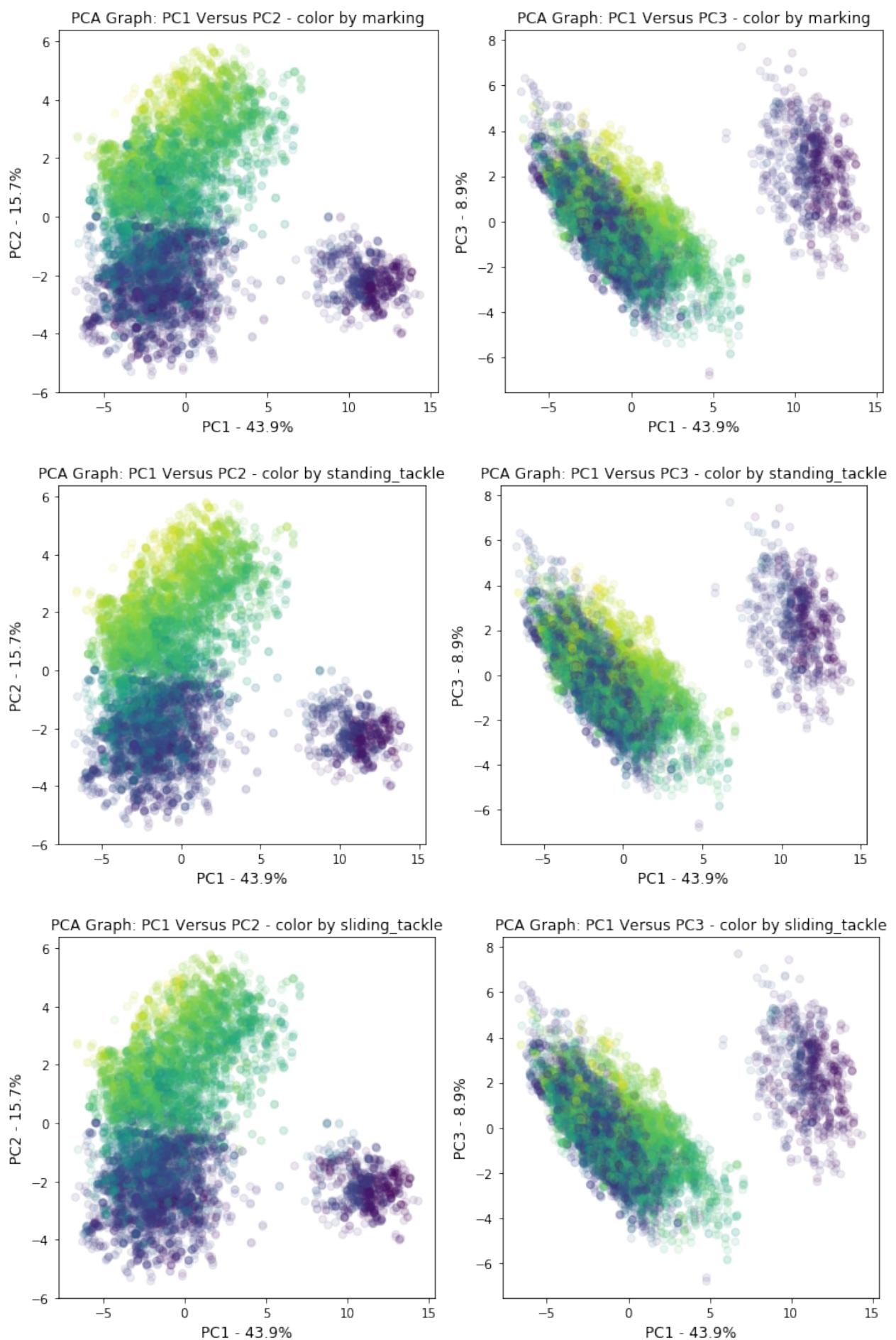


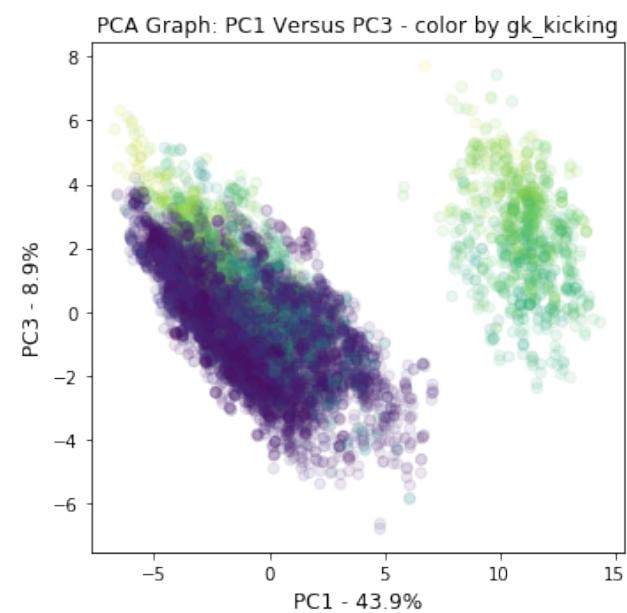
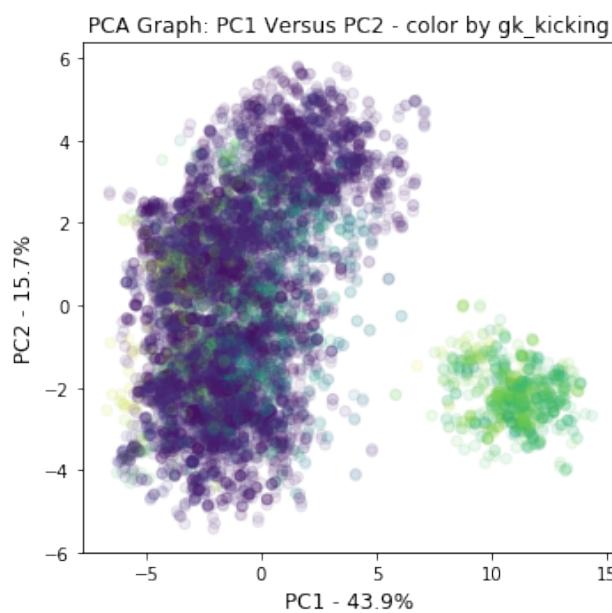
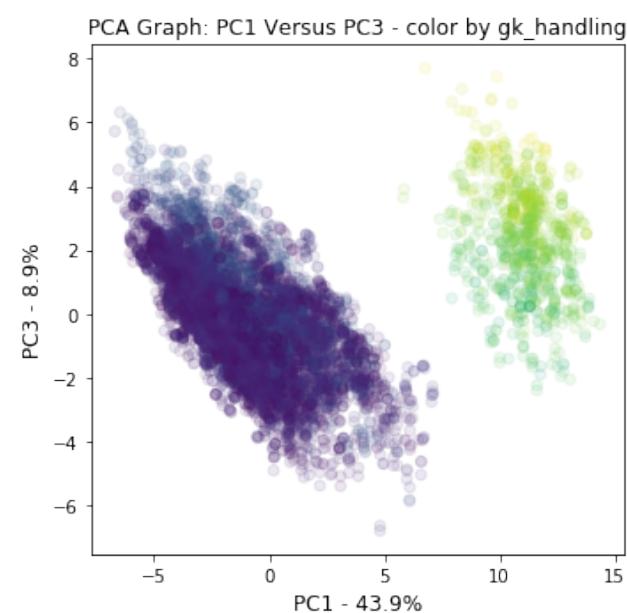
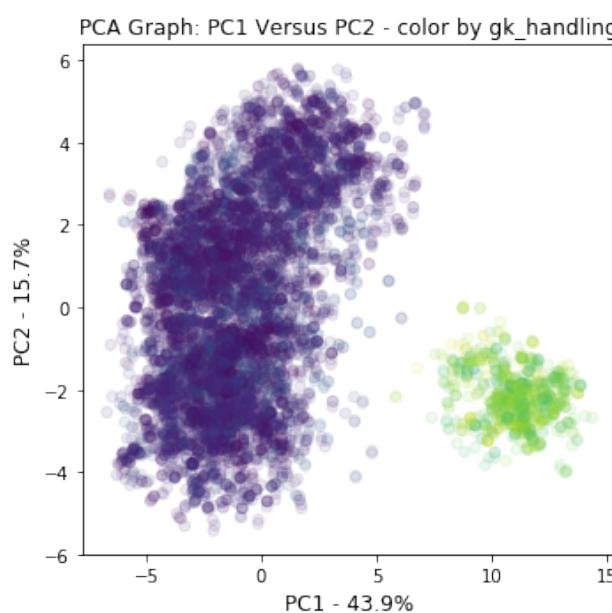
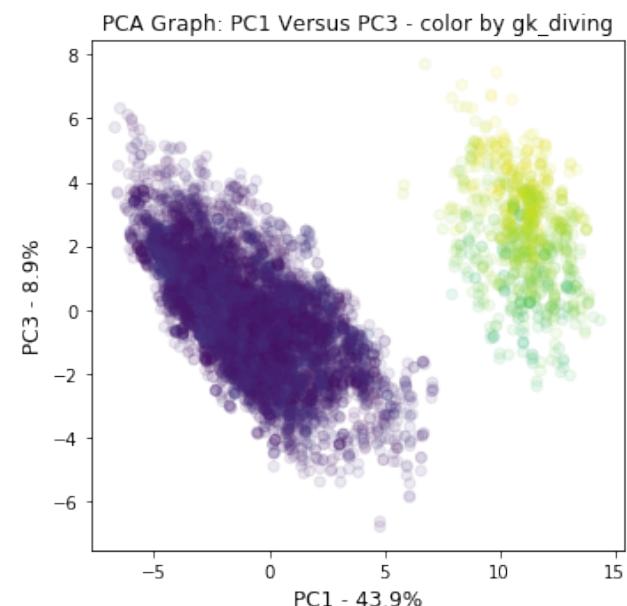
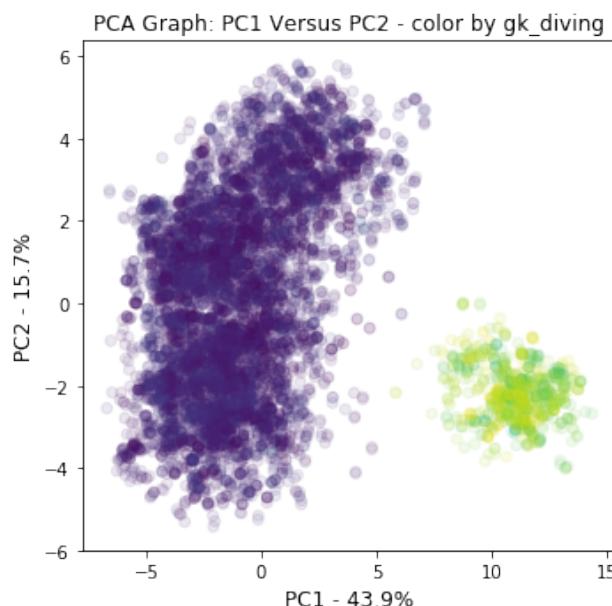


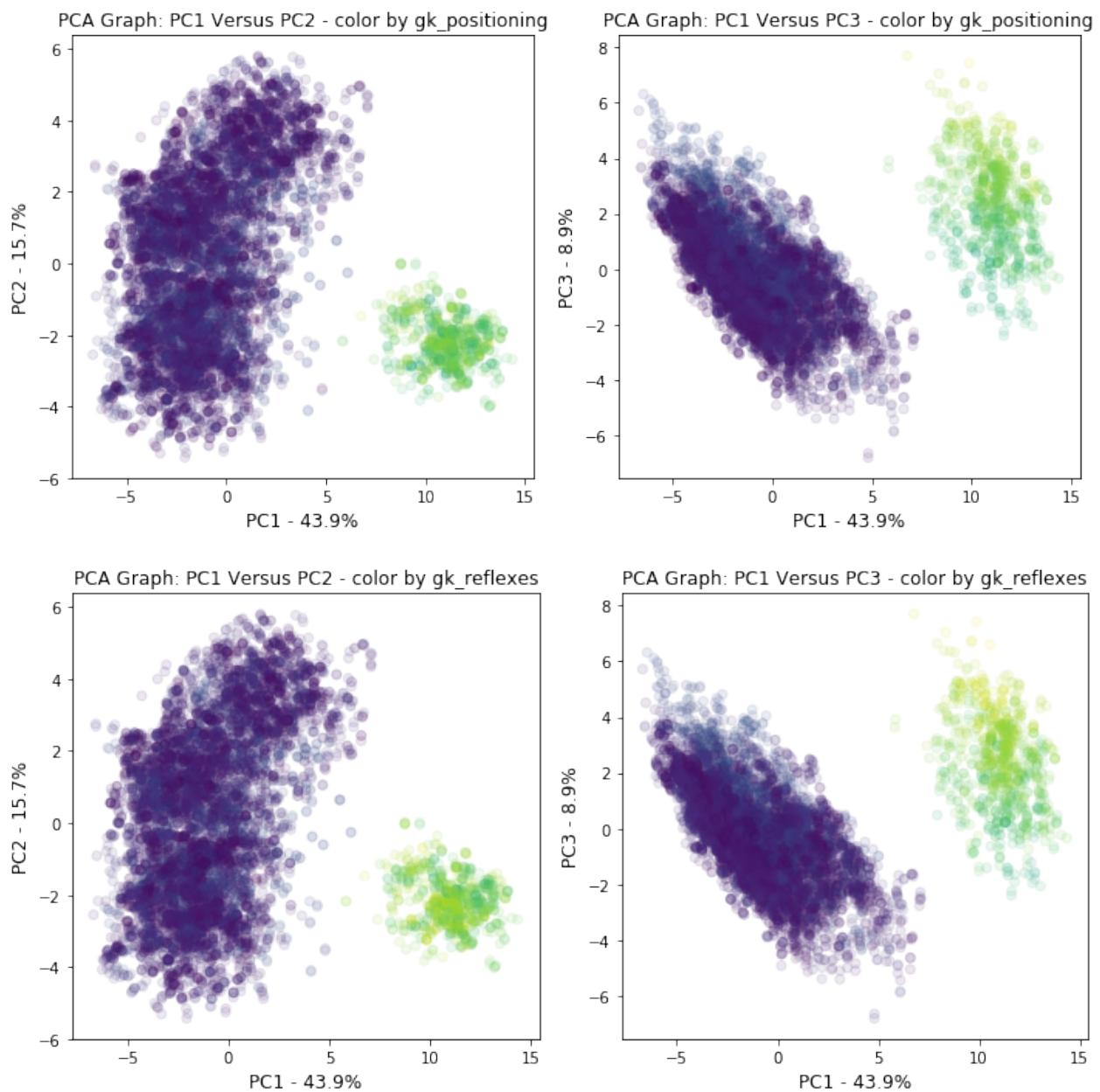












```
In [8]: #'attacking_work_rate', 'defensive_work_rate']
# plt.title('players with gk_diving > 50 (goalkeepers)')
# create scatter plot with ball control (highest loading score in PC1) against marking (highest loading score in PC2)
def plot (df_all, df_sub, hue_col):
    # first plot
    vis1=sns.lmplot(x='ball_control', y='marking', hue=hue_col, sharex=False,
lse, data=df_all, scatter=True, fit_reg=False, units=None, order=1, legend=True)
    plt.title('all players')
    plt.xlim(0,100)
    plt.ylim(0,100)
    plt.show()
    plt.close()
    # second plot: goalkeepers only
    vis2=sns.lmplot(x='ball_control', y='marking', hue=hue_col, sharex=False,
lse, data=df_sub, scatter=True, fit_reg=False, units=None, order=1, legend=True)
```

```

plt.title('players with gk_diving > 50 (goalkeepers)')
plt.xlim(0,100)
plt.ylim(0,100)
plt.show()
plt.close()

print('gk_diving > 50 (goalkeepers)')
df_goalkeepers=df_all_col.loc[df_all_col['gk_diving']>50]
#print(df_goalkeepers.head())
plot(df_all_col, df_goalkeepers, None)

# color by lefty and righty
plot(df_all_col, df_goalkeepers, 'preferred_foot')
# plot lefty only
print('plot preferred left foot')
df1=df_all_col.loc[df_all_col['preferred_foot']=='left']
df2=df_goalkeepers.loc[df_goalkeepers['preferred_foot']=='left']
plot(df1,df2,None)

# color by 'attaching_work_rate'
df3=df_all_col[df_all_col['attaching_work_rate'].isin(['low','medium','high'])]
df4=df_goalkeepers[df_goalkeepers['attaching_work_rate'].isin(['low','medium','high'])]
plot(df3, df4, 'attaching_work_rate')

#plot jointplot with goal keeper attributes:
'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning', 'gk_reflexes'
def joint_plot (df, title) :
    vis=sns.jointplot(x='ball_control',y='gk_diving', data=df, xlim=(0,100),
    ylim=(0,100), stat_func=None)
    plt.title(title, loc='left')
    plt.show()
    plt.close()
    vis=sns.jointplot(x='ball_control',y='gk_handling', data=df, xlim=(0,100),
    ylim=(0,100), stat_func=None)
    plt.title(title, loc='left')
    plt.show()
    plt.close()
    vis=sns.jointplot(x='ball_control',y='gk_kicking', data=df, xlim=(0,100),
    ylim=(0,100), stat_func=None)
    plt.title(title, loc='left')
    plt.show()
    plt.close()
    vis=sns.jointplot(x='ball_control',y='gk_positioning', data=df, xlim=(0,100),
    ylim=(0,100), stat_func=None)
    plt.title(title, loc='left')
    plt.show()
    plt.close()
    vis=sns.jointplot(x='ball_control',y='gk_reflexes', data=df, xlim=(0,100),
    ylim=(0,100), stat_func=None)
    plt.title(title, loc='left')
    plt.show()
    plt.close()

    vis=sns.jointplot(x='marking',y='gk_diving', data=df, xlim=(0,100), yl

```

```

im=(0,100), stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='marking',y='gk_handling', data=df, xlim=(0,100),
ylim=(0,100), stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='marking',y='gk_kicking', data=df, xlim=(0,100), y
lim=(0,100), stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='marking',y='gk_positioning', xlim=(0,100), ylim=(0,1
00), data=df, stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='marking',y='gk_reflexes', xlim=(0,100), ylim=(0,1
00), data=df, stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()

'''

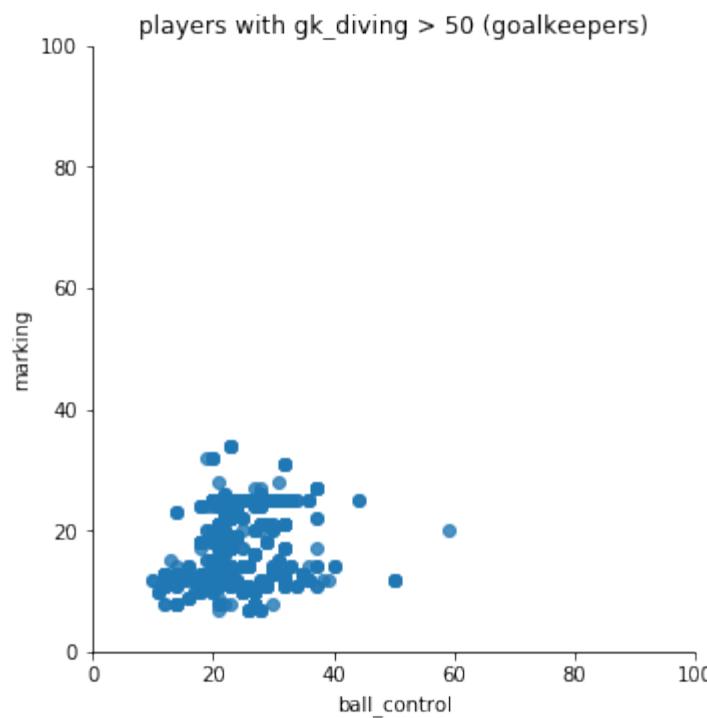
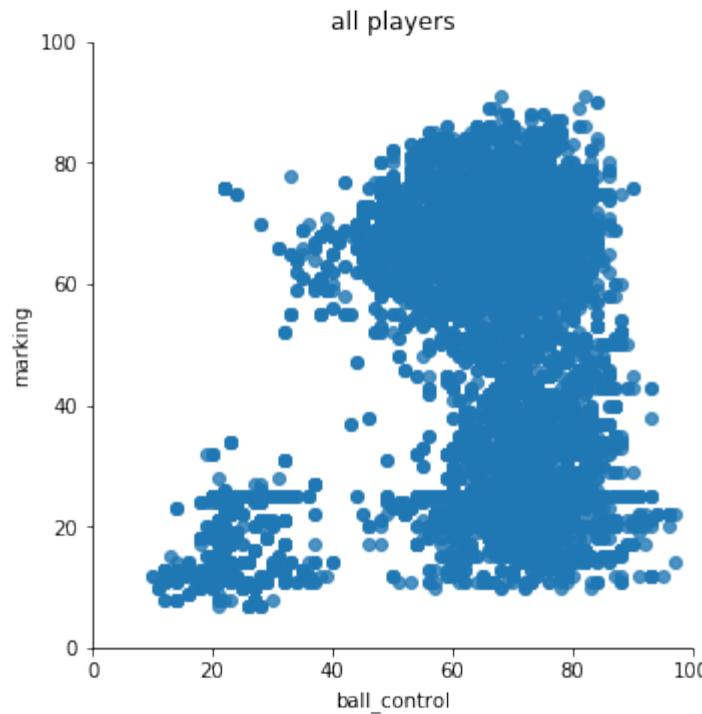
vis=sns.jointplot(x='dribbling',y='gk_diving', data=df, xlim=(0,100),
ylim=(0,100), stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='dribbling',y='gk_handling', data=df, xlim=(0,100)
, ylim=(0,100), stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='dribbling',y='gk_kicking', data=df, xlim=(0,100),
ylim=(0,100), stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='dribbling',y='gk_positioning', xlim=(0,100), ylim
=(0,100), data=df, stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()
vis=sns.jointplot(x='dribbling',y='gk_reflexes', xlim=(0,100), ylim=(0
,100), data=df, stat_func=None)
plt.title(title, loc='left')
plt.show()
plt.close()

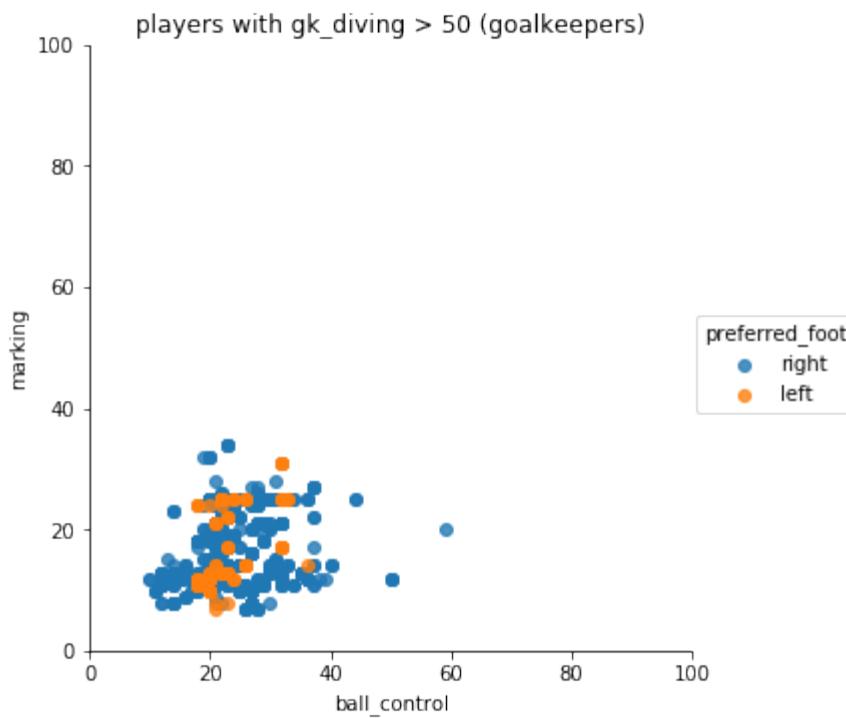
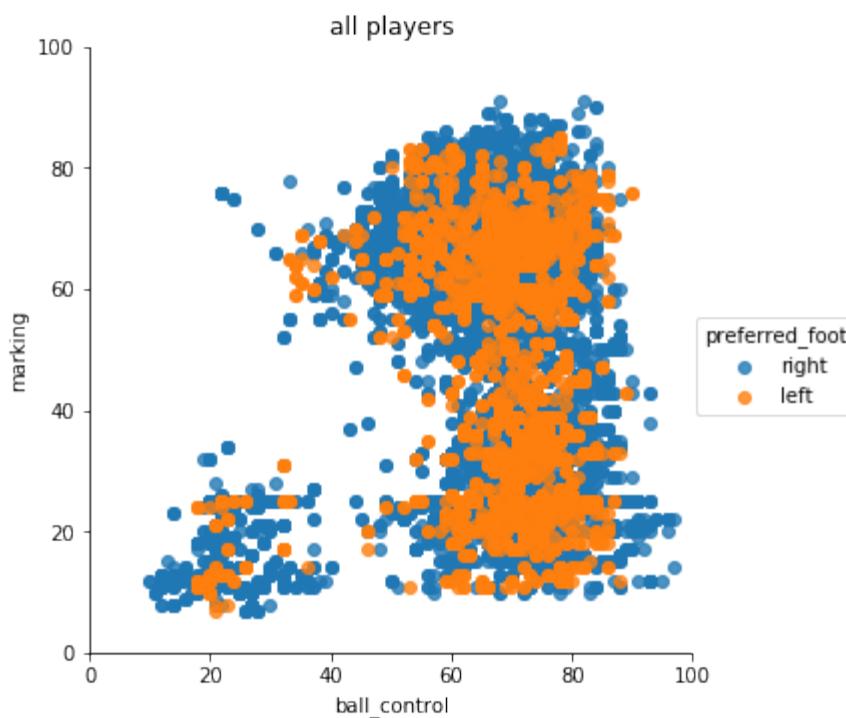
'''

# df_goalkeepers=df_all_col.loc[df_all_col['gk_diving']>40]
joint_plot (df_all_col, 'all players')
joint_plot (df_goalkeepers, 'gk_diving > 50 (goalkeepers)')

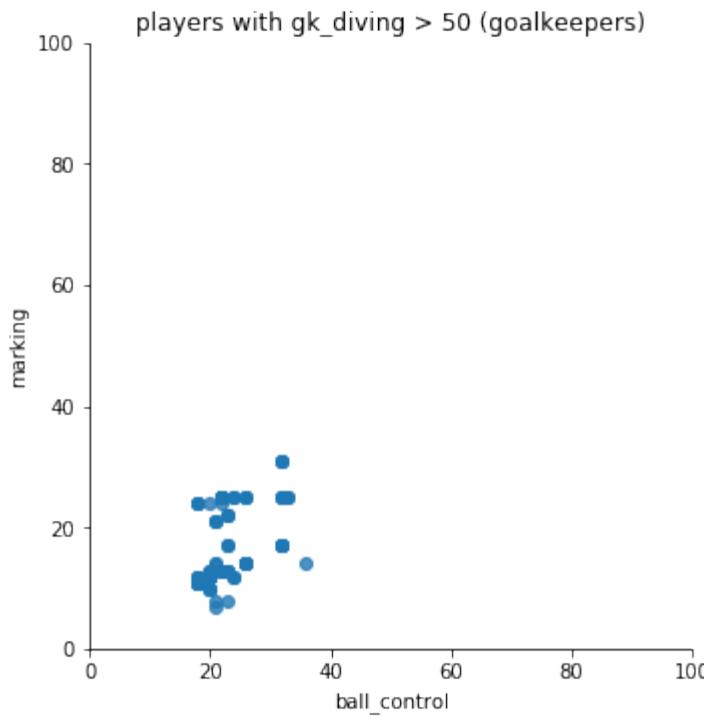
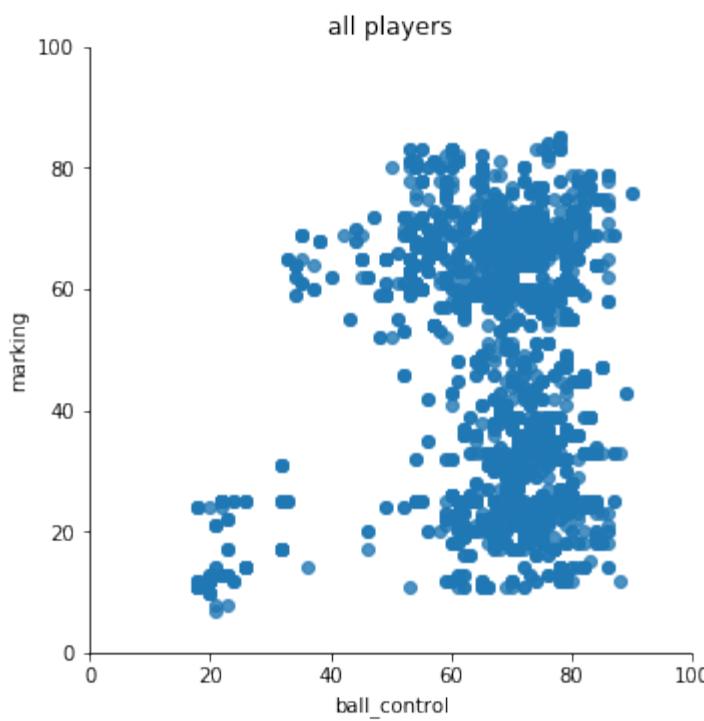
```

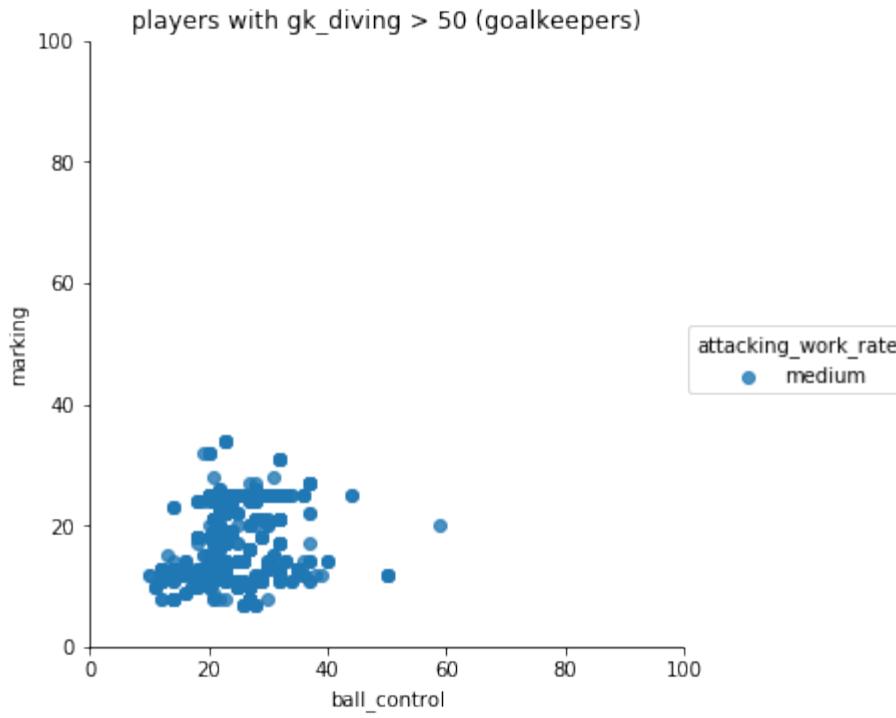
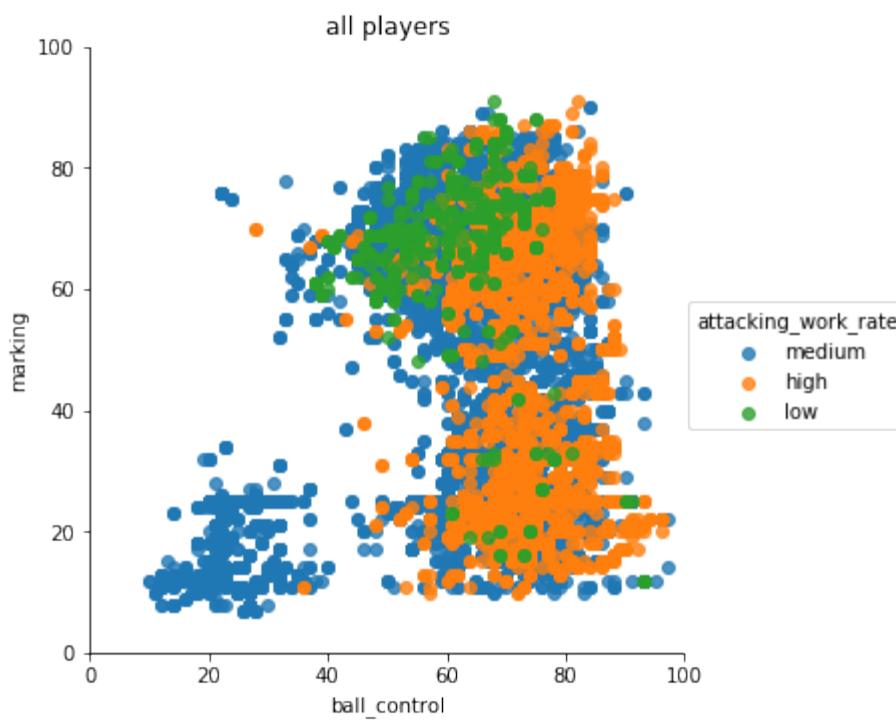
gk_diving > 50 (goalkeepers)

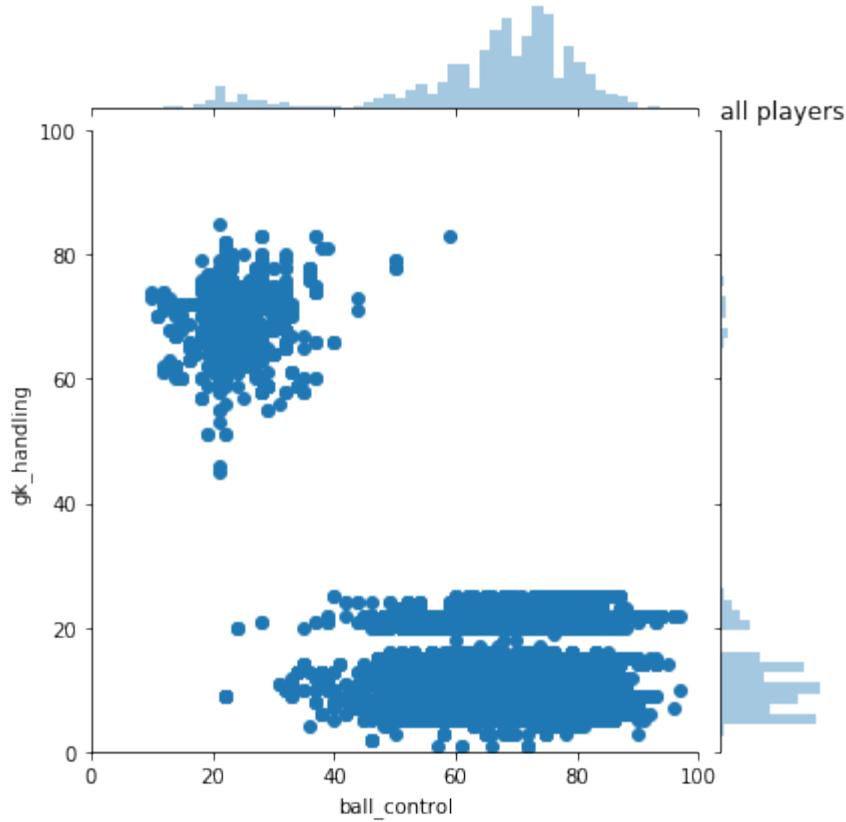
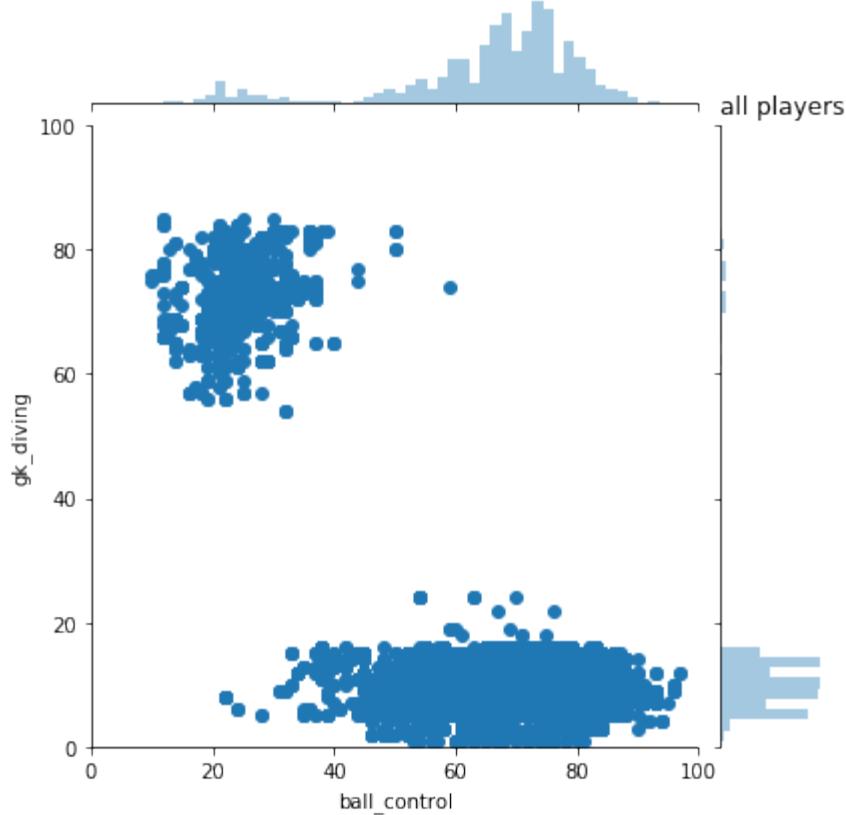


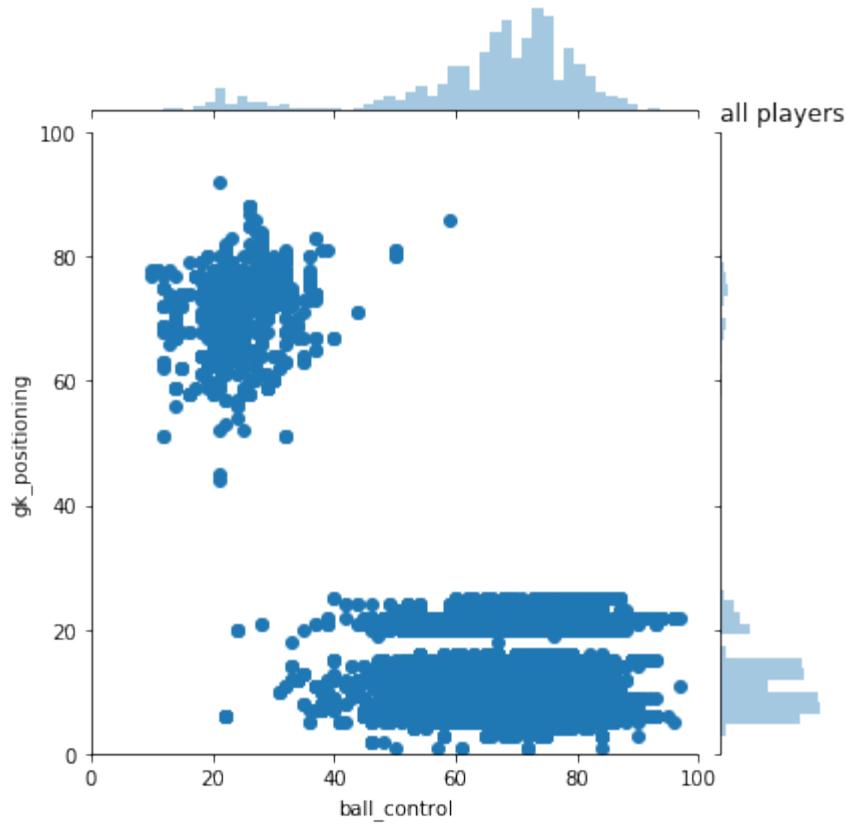
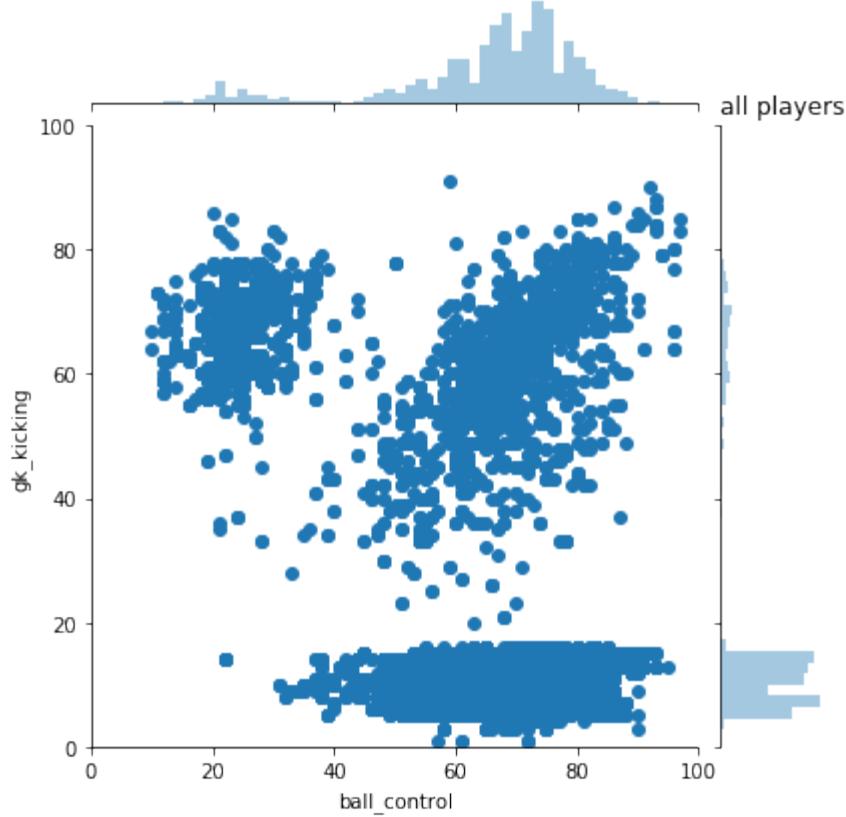


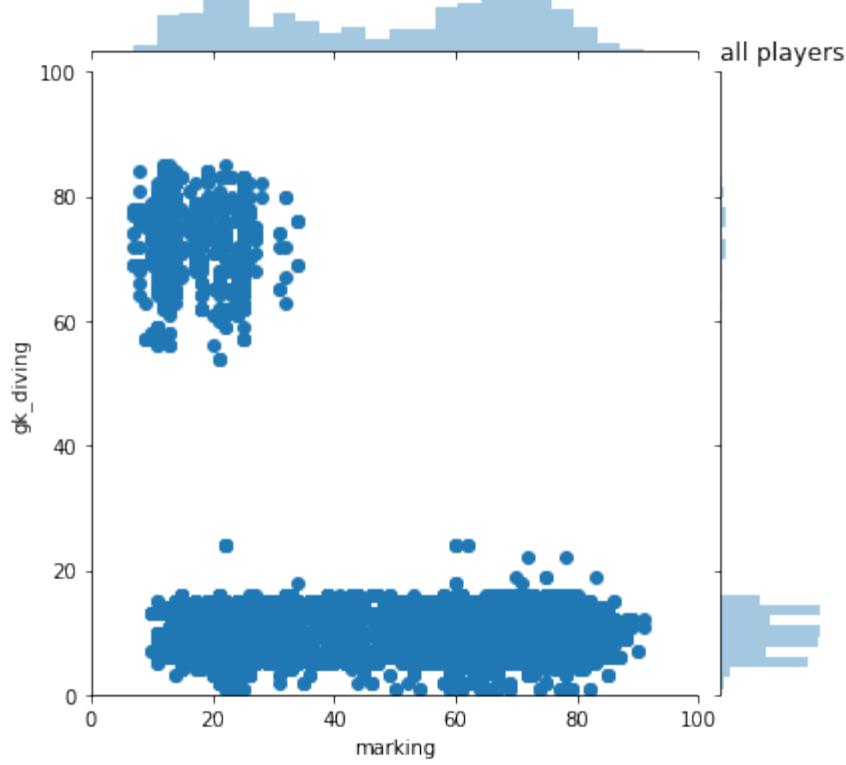
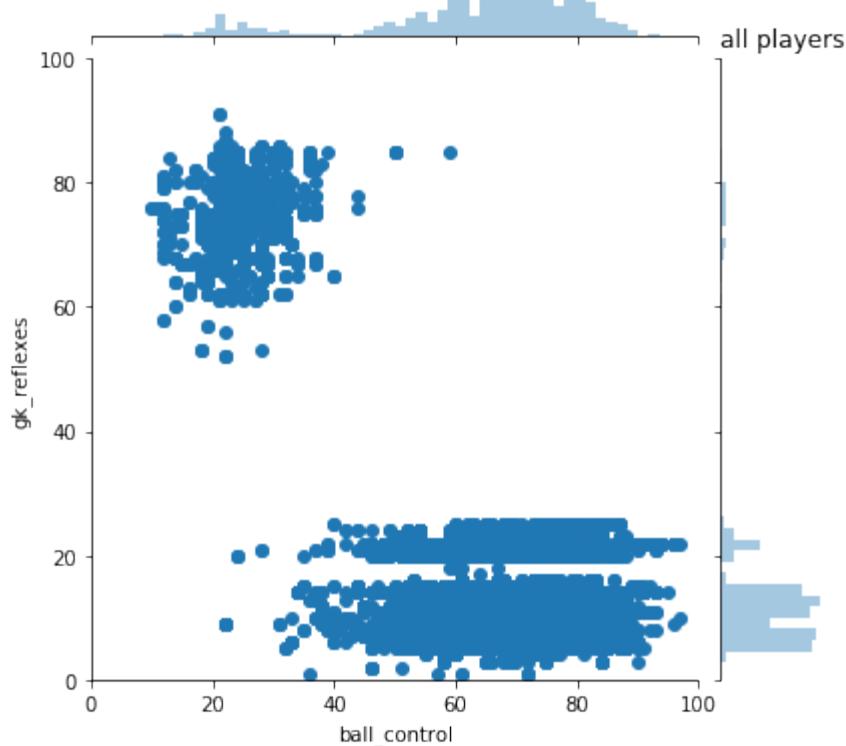
plot preferred left foot

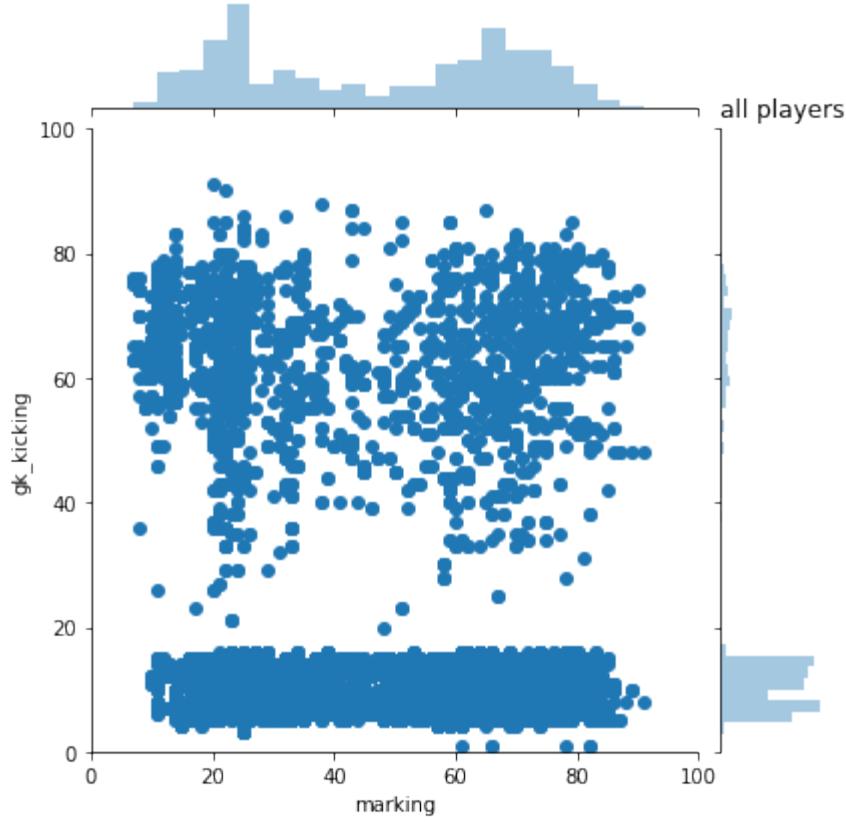
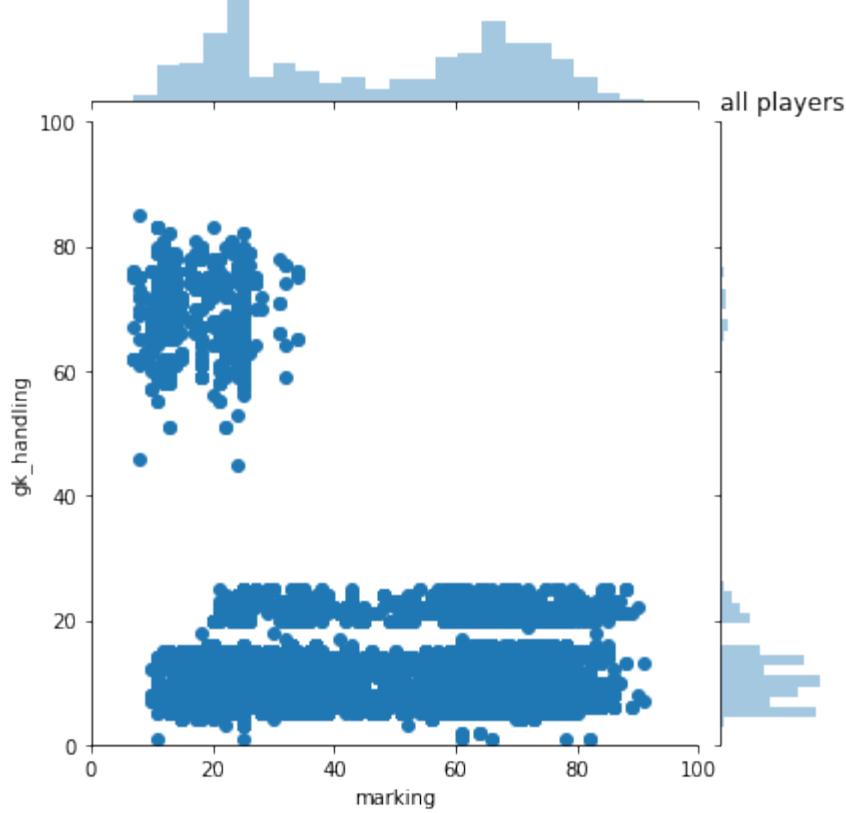


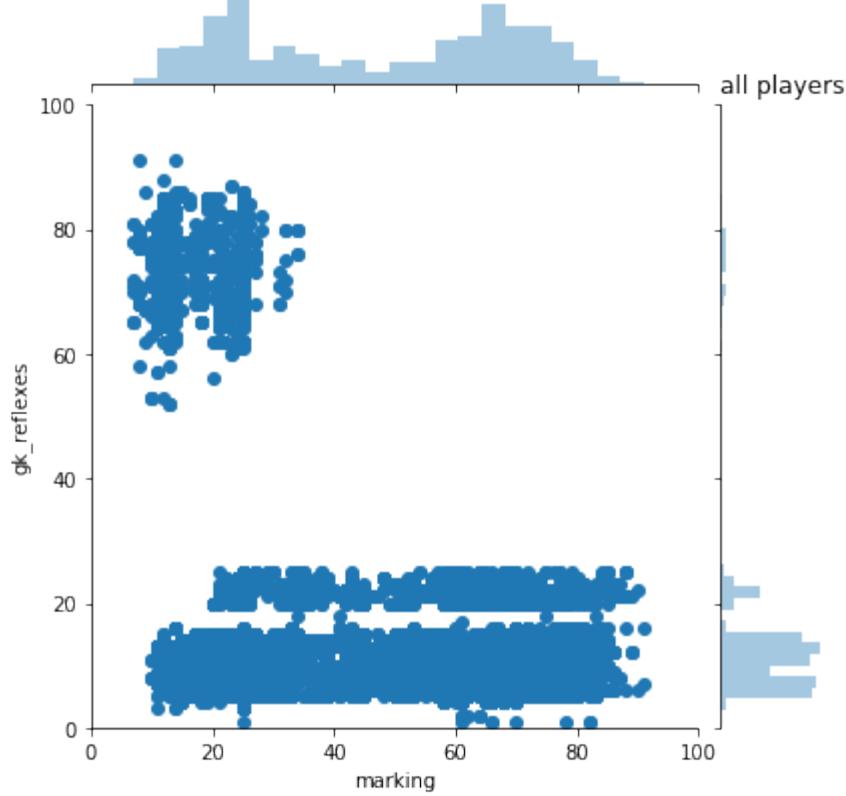
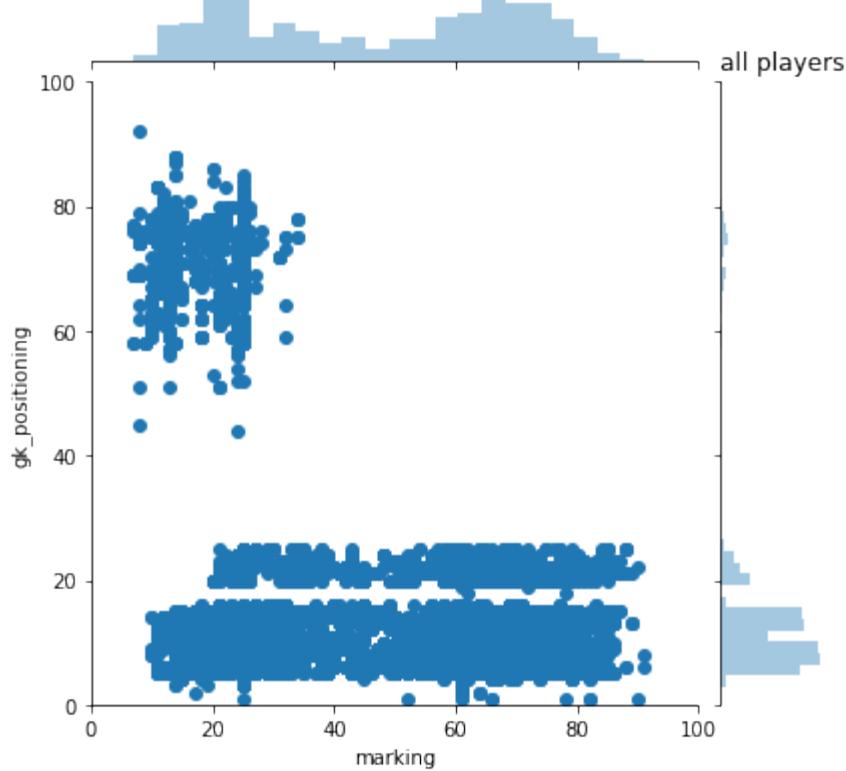


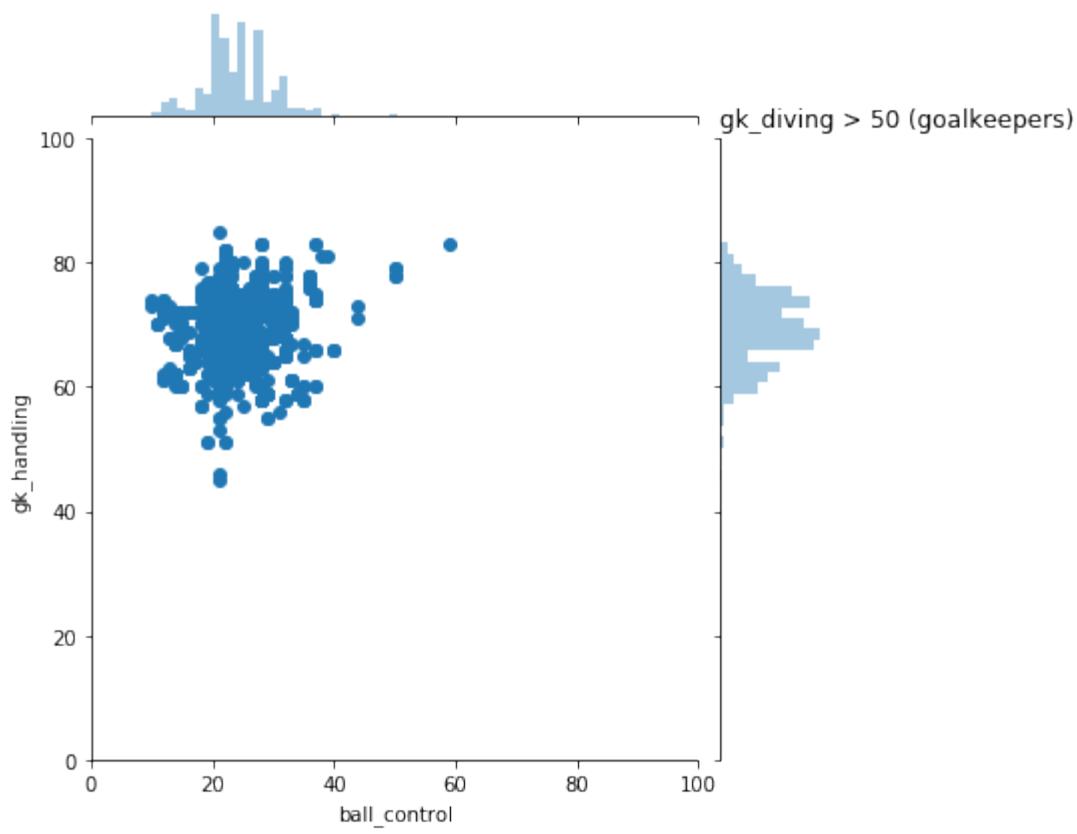
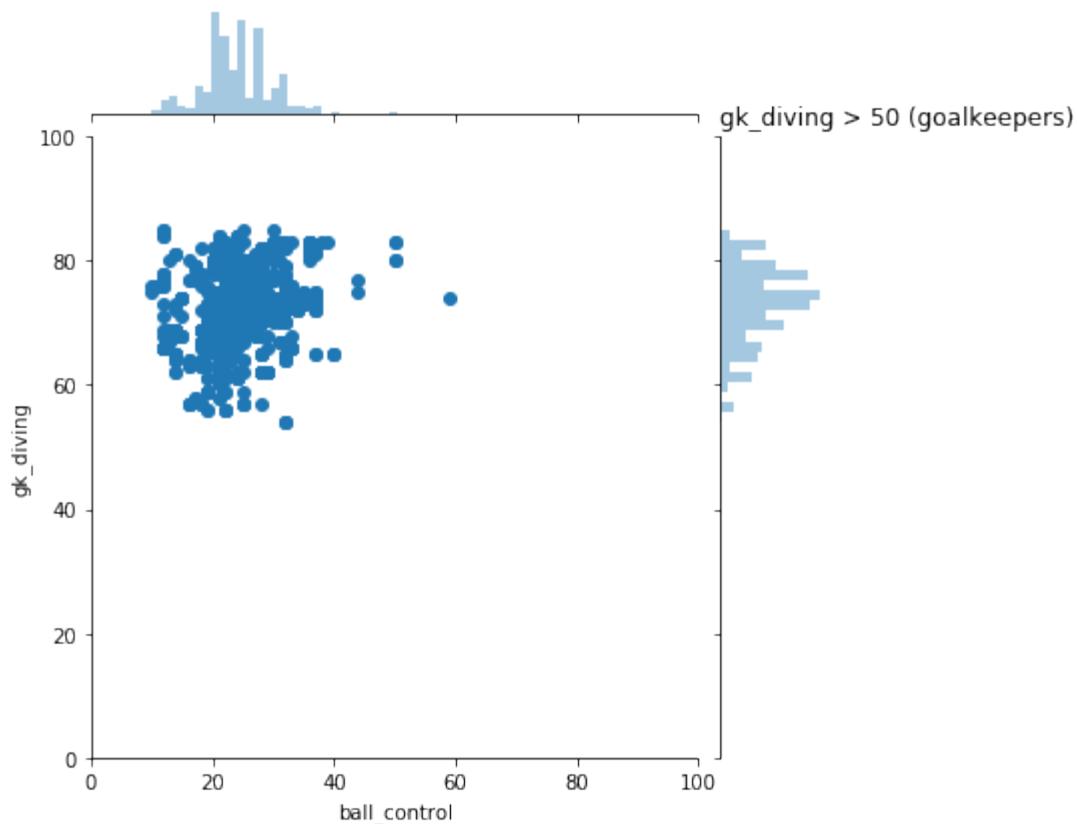


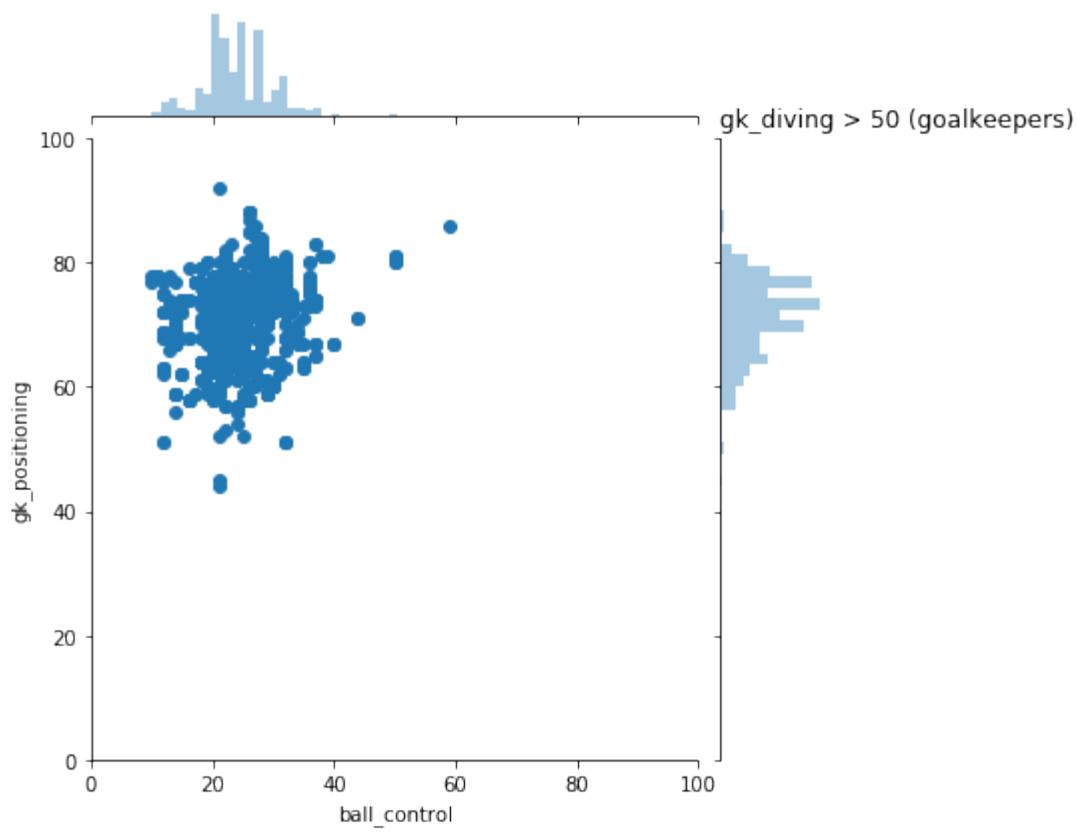
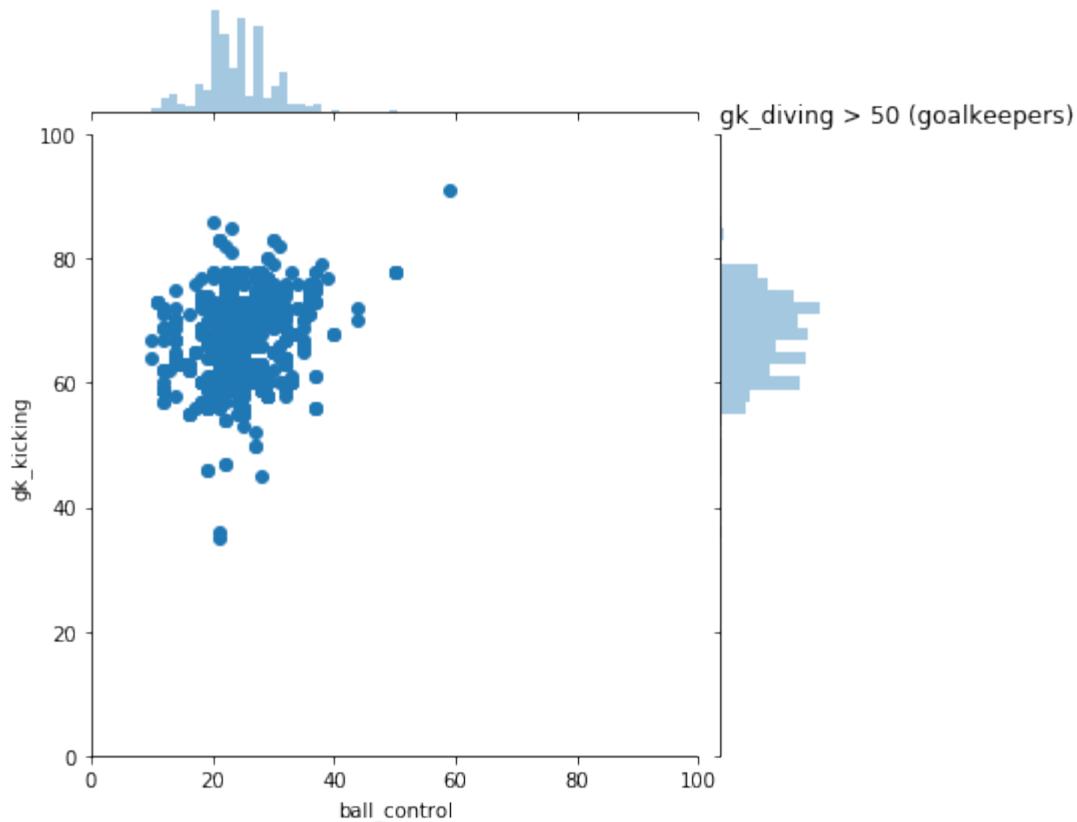


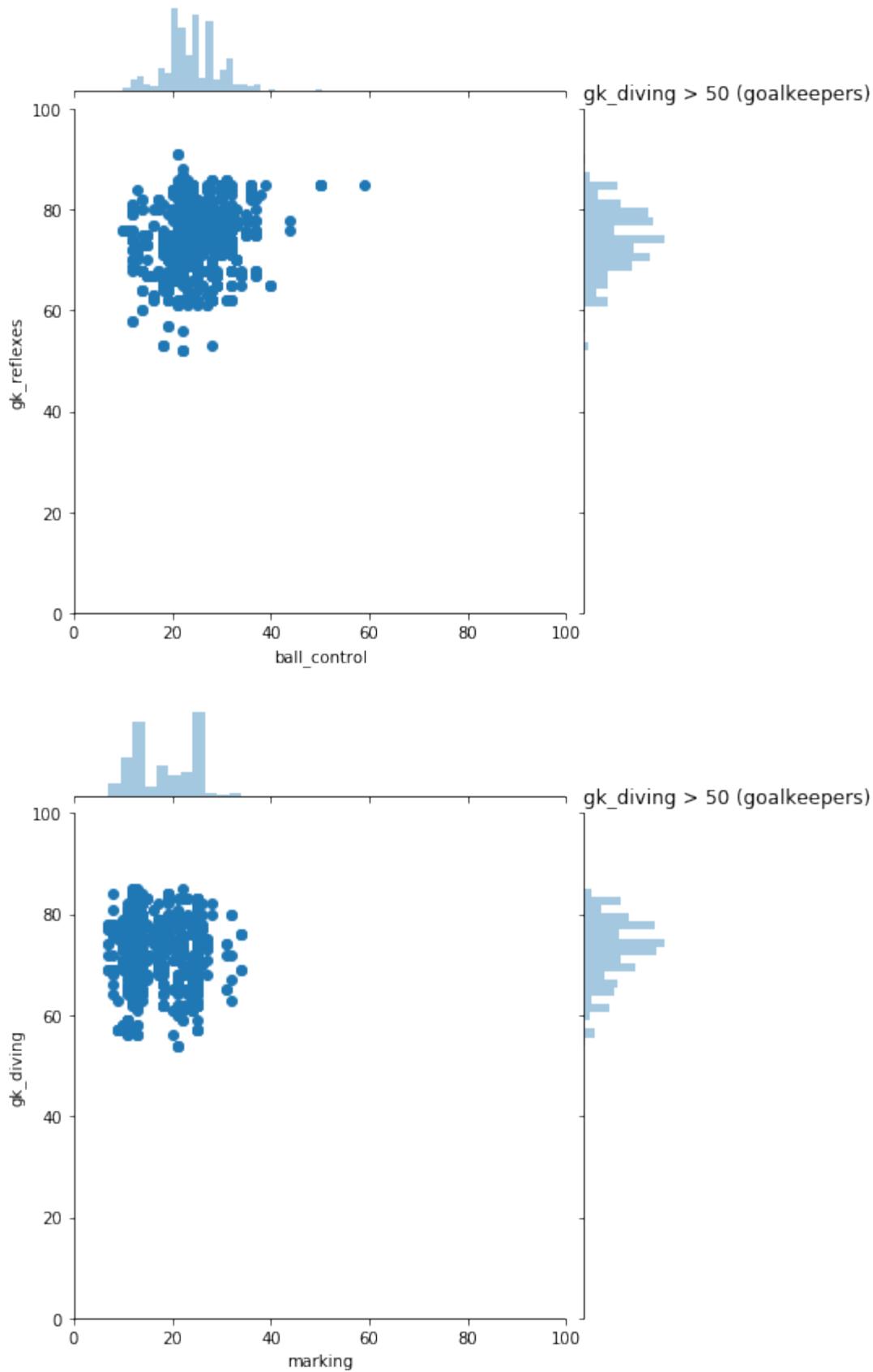


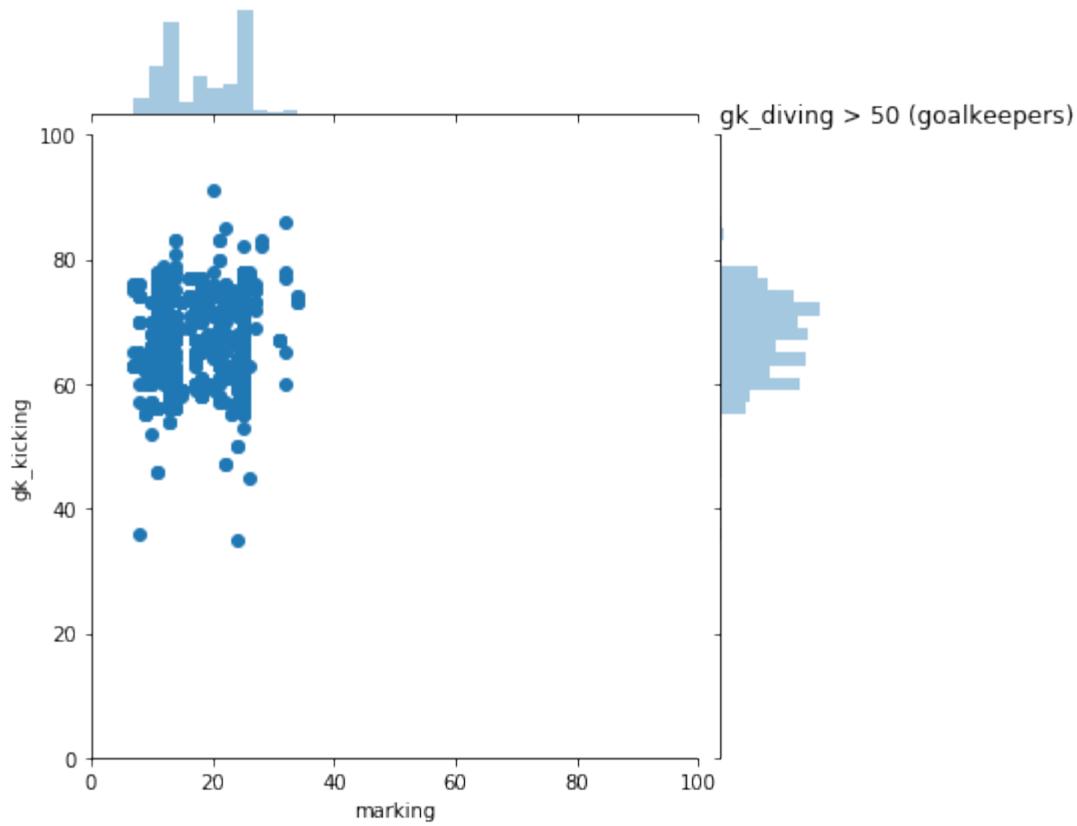
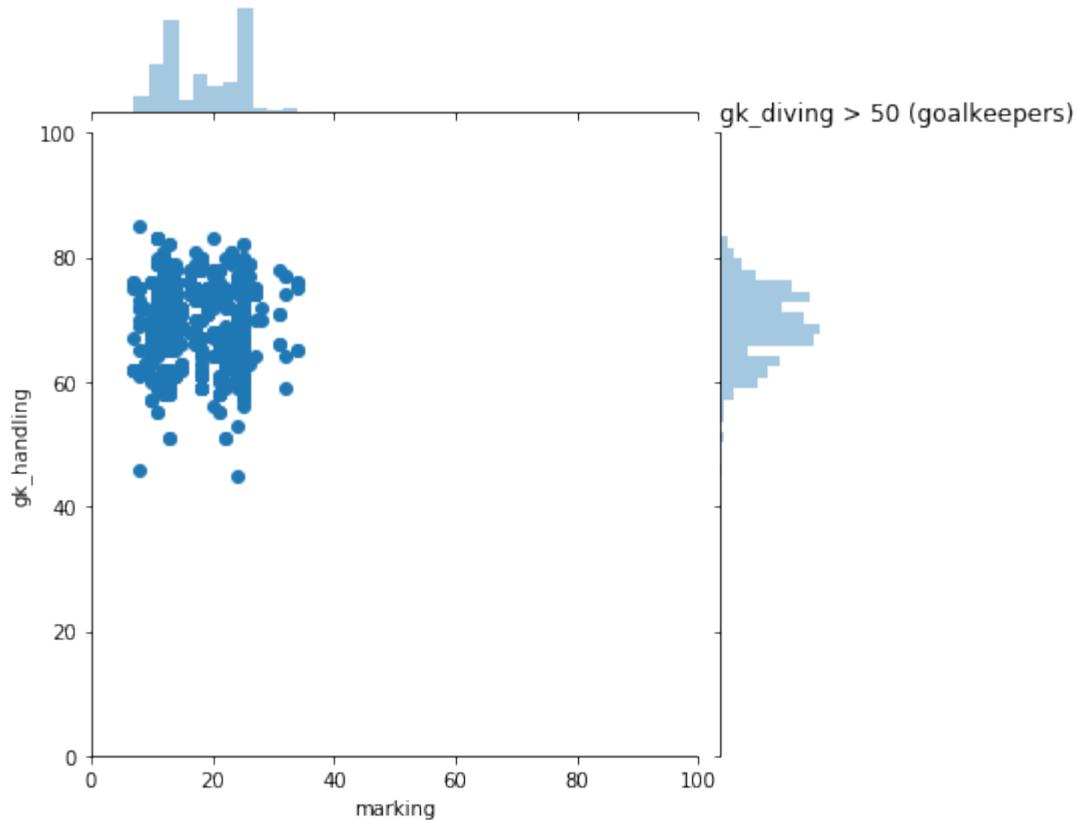


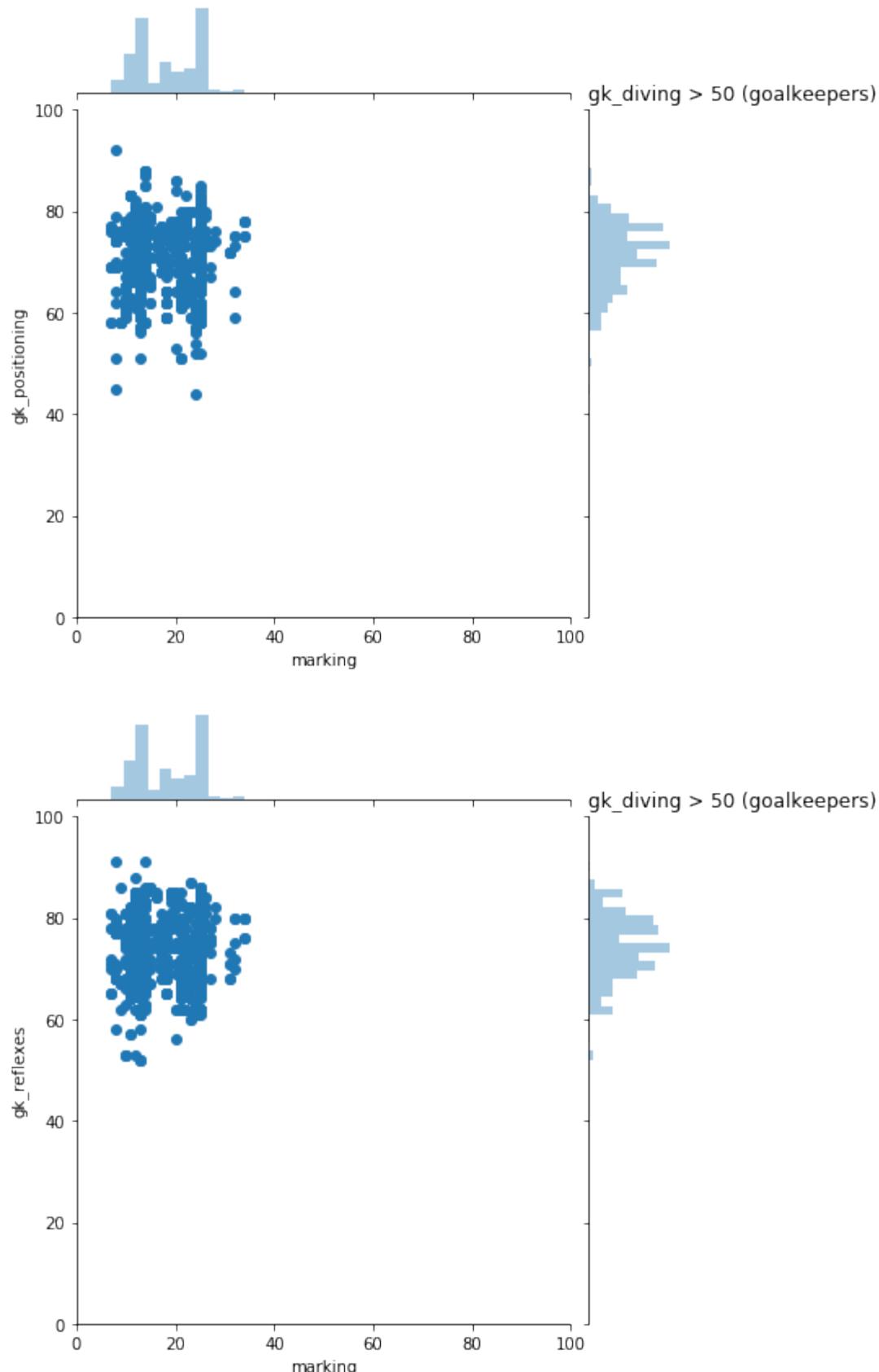












```
In [42]: import seaborn as sns
df1=df_all_col[df_all_col['defensive_work_rate'].isin(['low','medium','high'])]
df2=df_all_col[df_all_col['defensive_work_rate'].isin(['high'])]
df3=df_all_col[df_all_col['defensive_work_rate'].isin(['medium'])]
```

```

df4=df_all_col[df_all_col['defensive_work_rate'].isin(['low'])]

def lmplot (df):
    vis=sns.lmplot(x='marking', y='overall_rating', hue='defensive_work_rate', sharex=False, data=df, \
                    scatter=True, fit_reg=False, units=None, order=1, legend=True)
    plt.title('Colored By Defensive Work Rate')
    plt.show()
    plt.close()

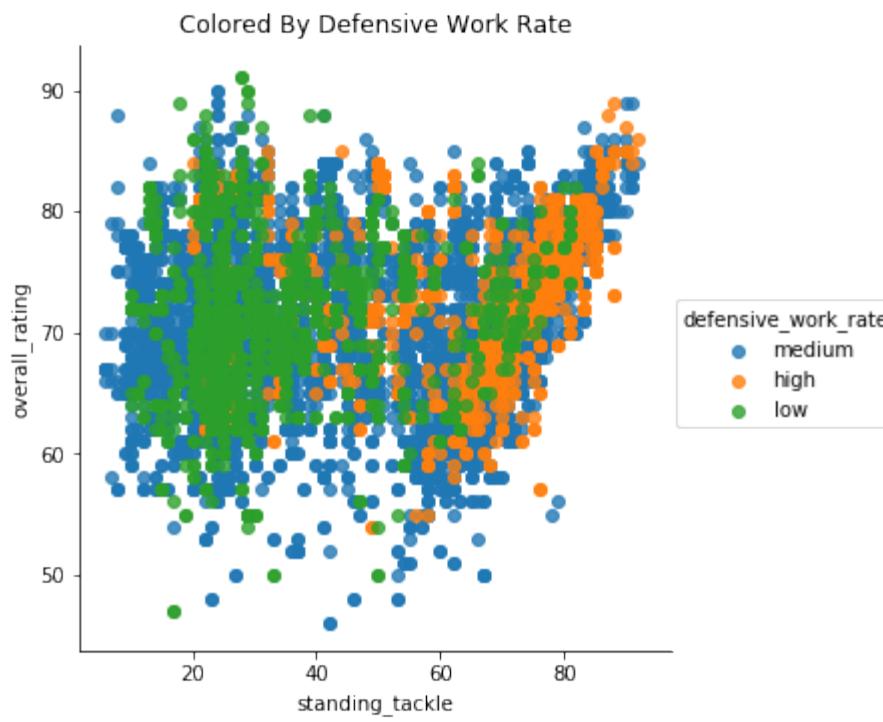
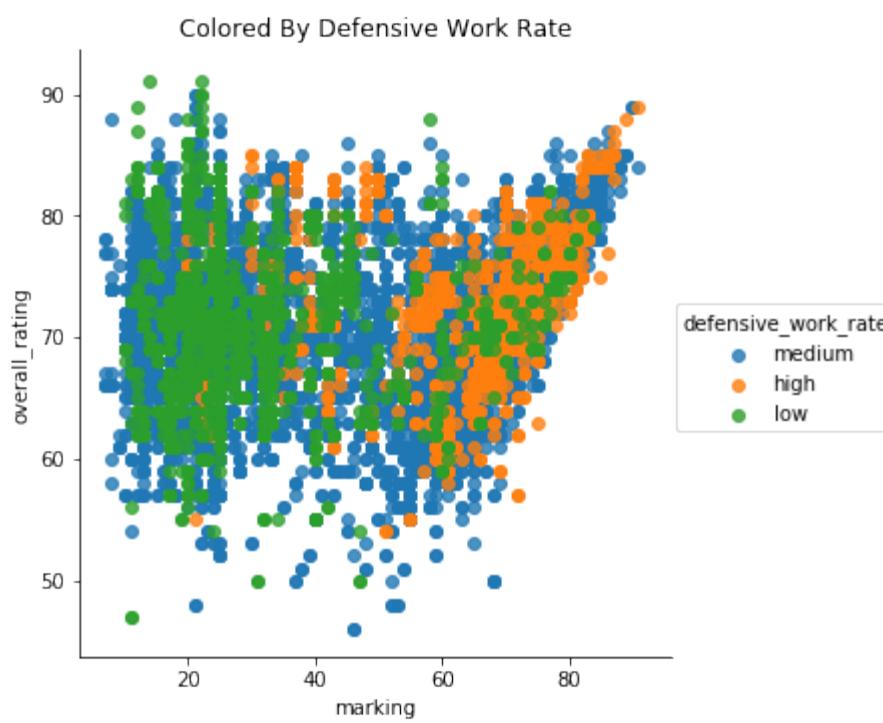
    vis=sns.lmplot(x='standing_tackle', y='overall_rating', hue='defensive_work_rate', sharex=False, data=df, \
                    scatter=True, fit_reg=False, units=None, order=1, legend=True)
    plt.title('Colored By Defensive Work Rate')
    plt.show()
    plt.close()

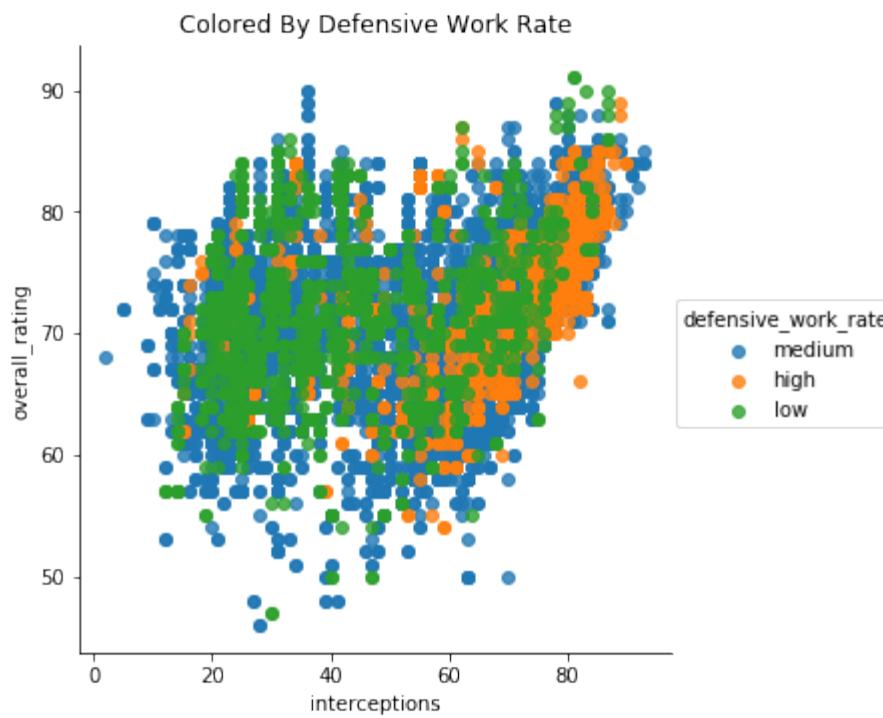
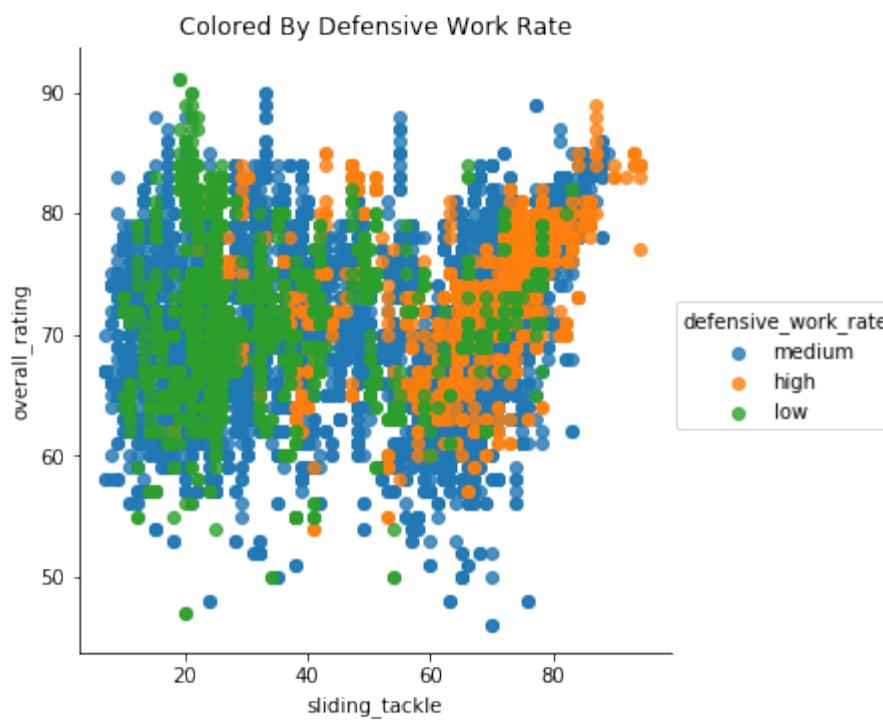
    vis=sns.lmplot(x='sliding_tackle', y='overall_rating', hue='defensive_work_rate', sharex=False, data=df, \
                    scatter=True, fit_reg=False, units=None, order=1, legend=True)
    plt.title('Colored By Defensive Work Rate')
    plt.show()
    plt.close()

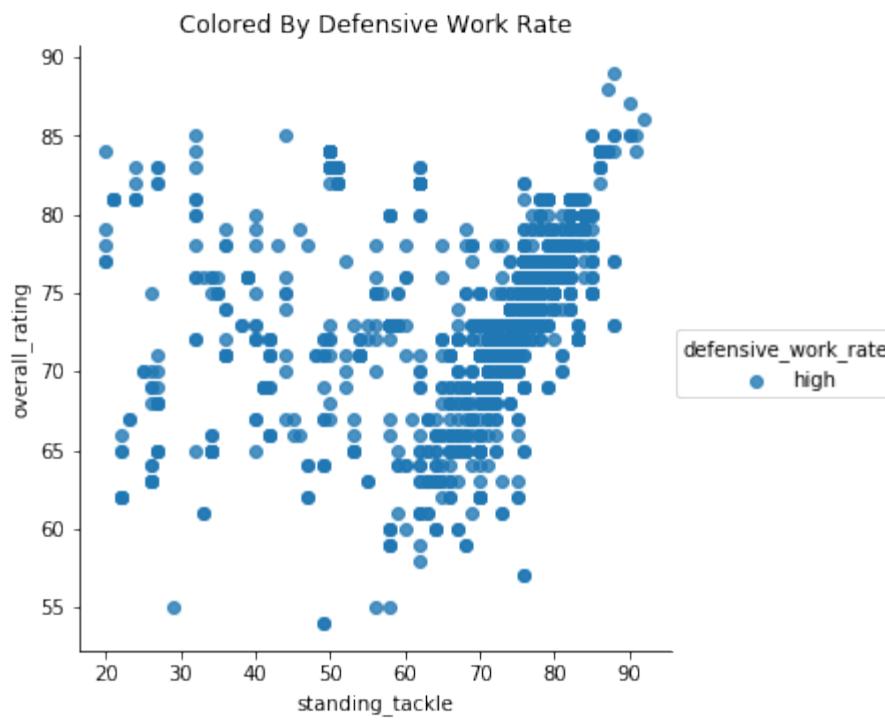
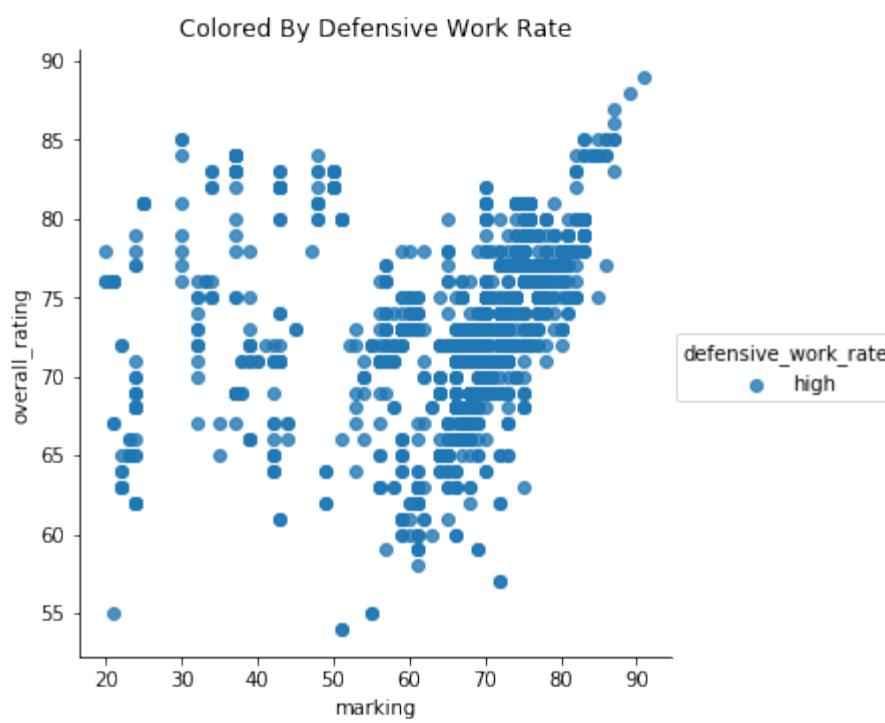
    vis=sns.lmplot(x='interceptions', y='overall_rating', hue='defensive_work_rate', sharex=False, data=df, \
                    scatter=True, fit_reg=False, units=None, order=1, legend=True)
    plt.title('Colored By Defensive Work Rate')
    plt.show()
    plt.close()

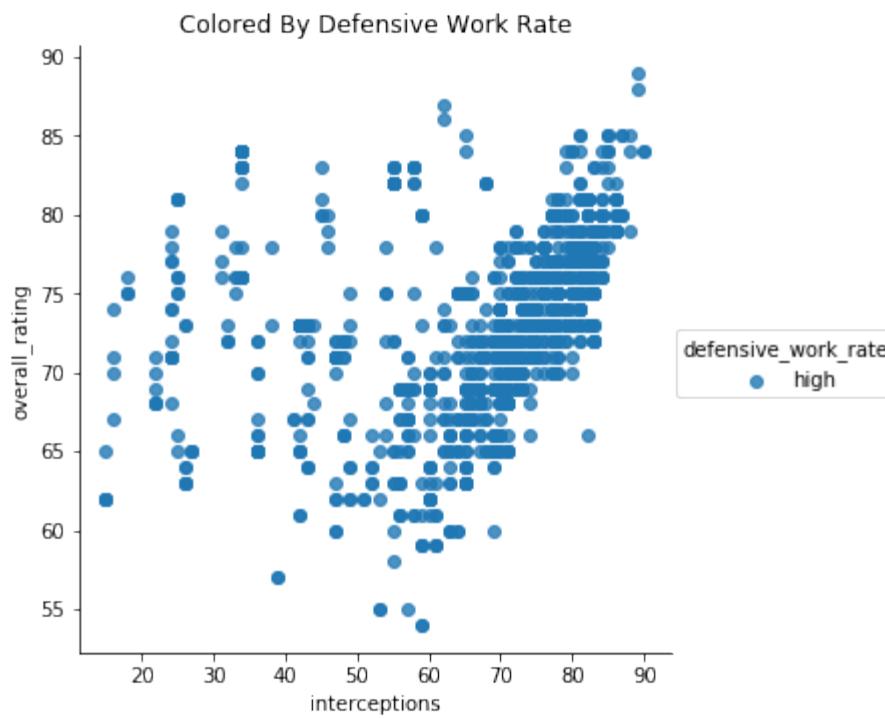
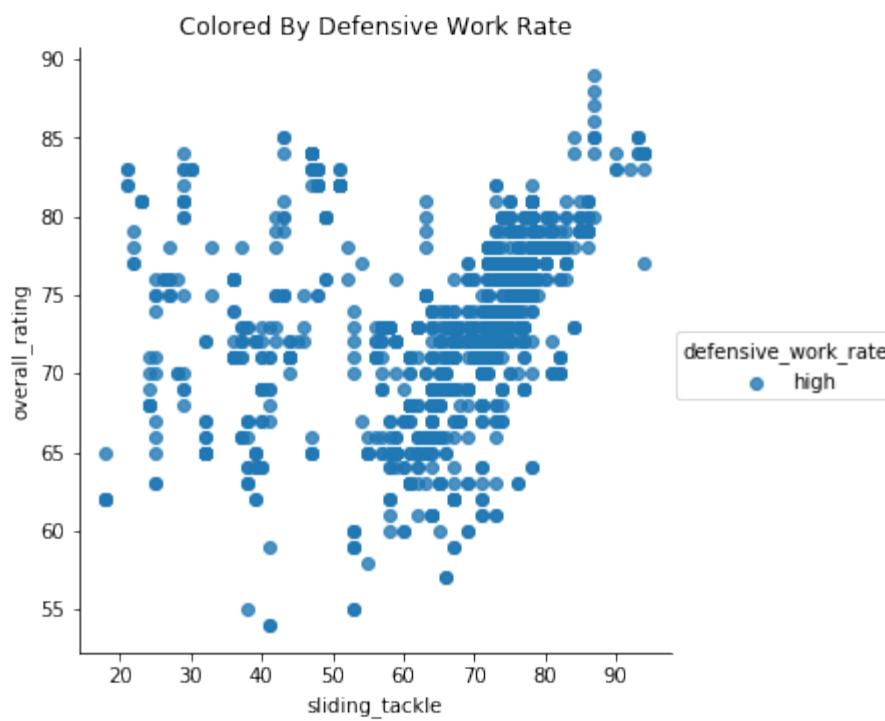
lmplot(df1) #color by 'defensive_work_rate'].isin(['low','medium','high'])
)
lmplot(df2) #color by 'defensive_work_rate'].isin(['high'])
lmplot(df3) #color by 'defensive_work_rate'].isin(['medium'])
lmplot(df4) #color by 'defensive_work_rate'].isin(['low'])

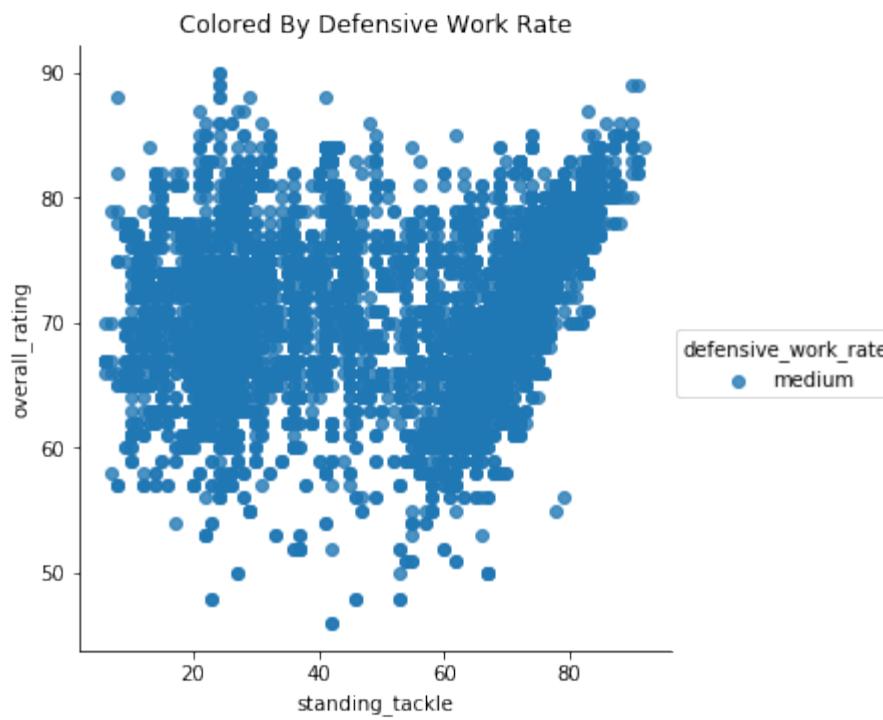
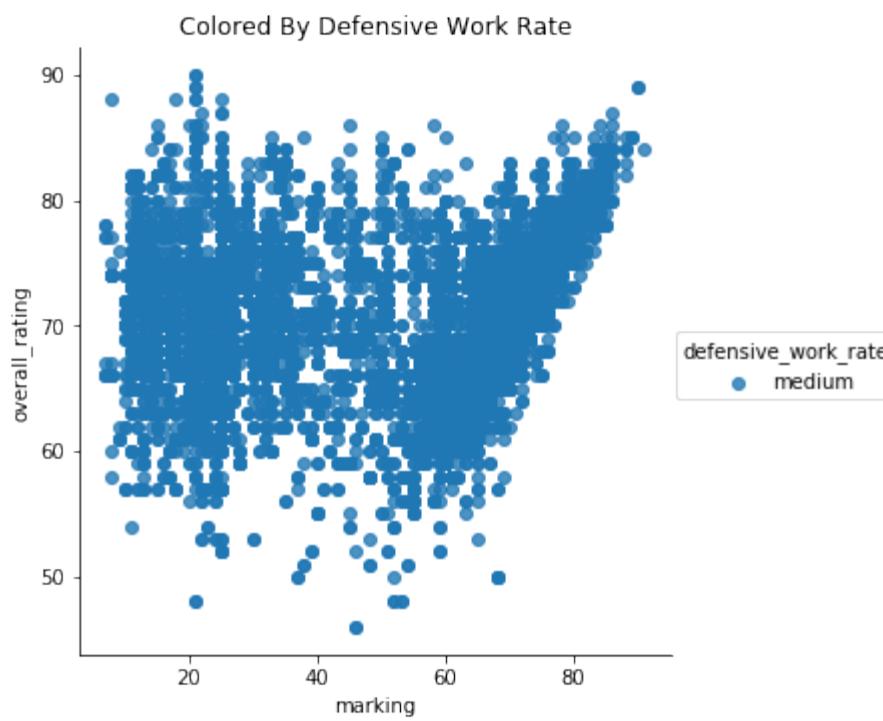
```

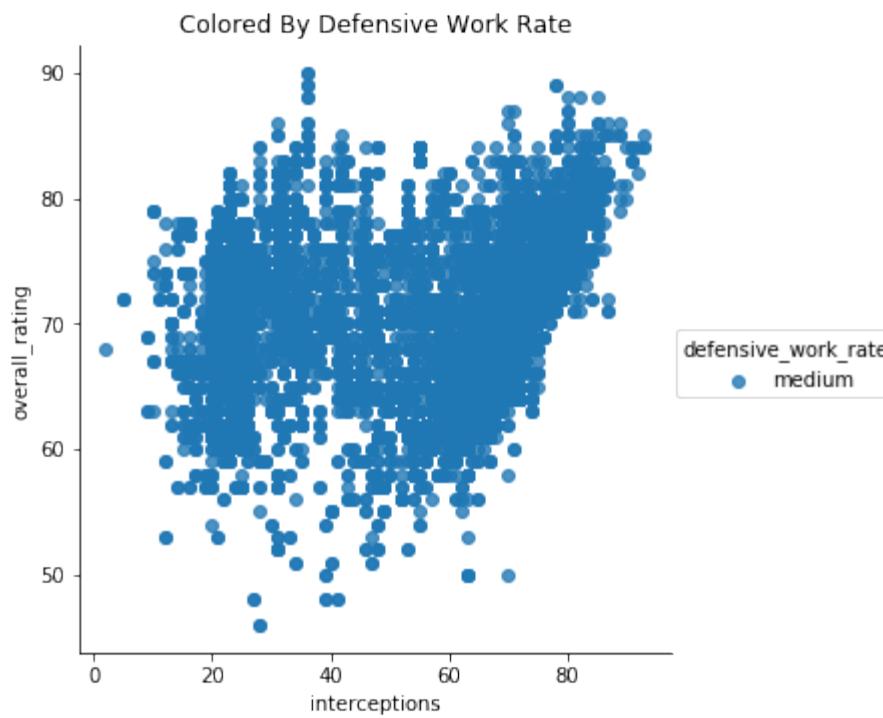
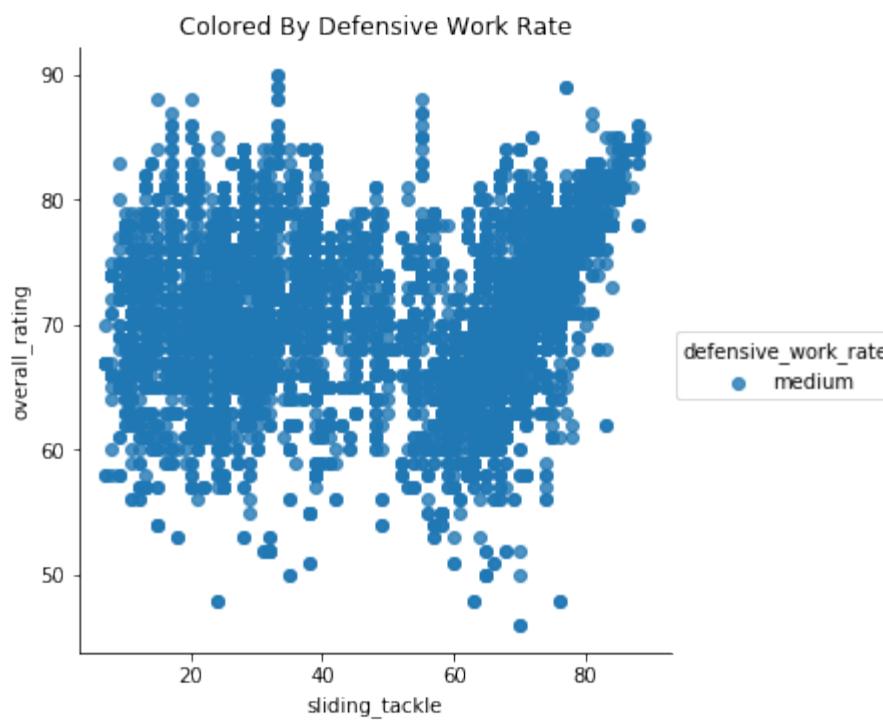


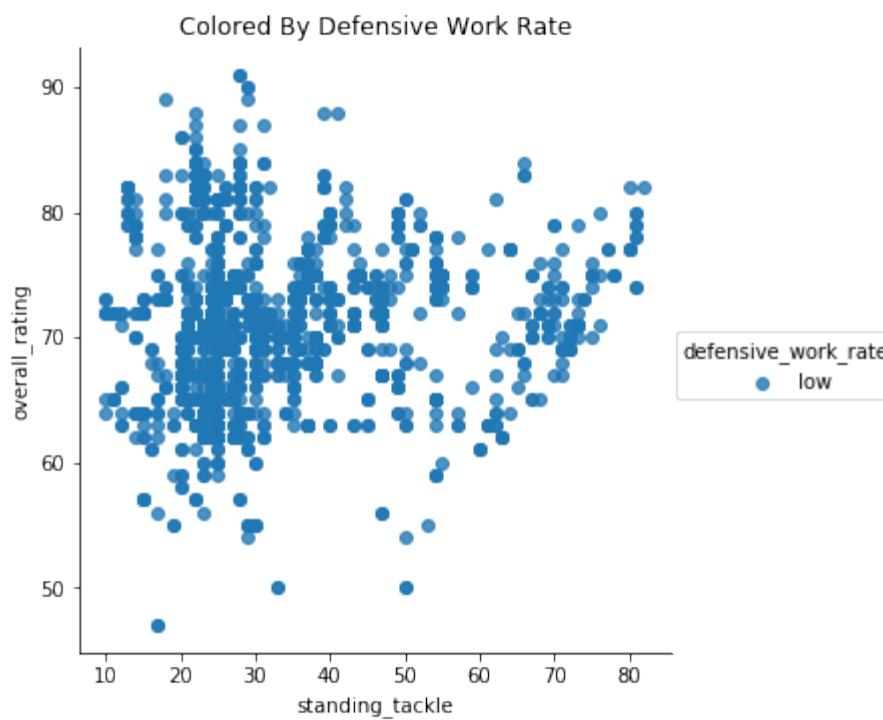
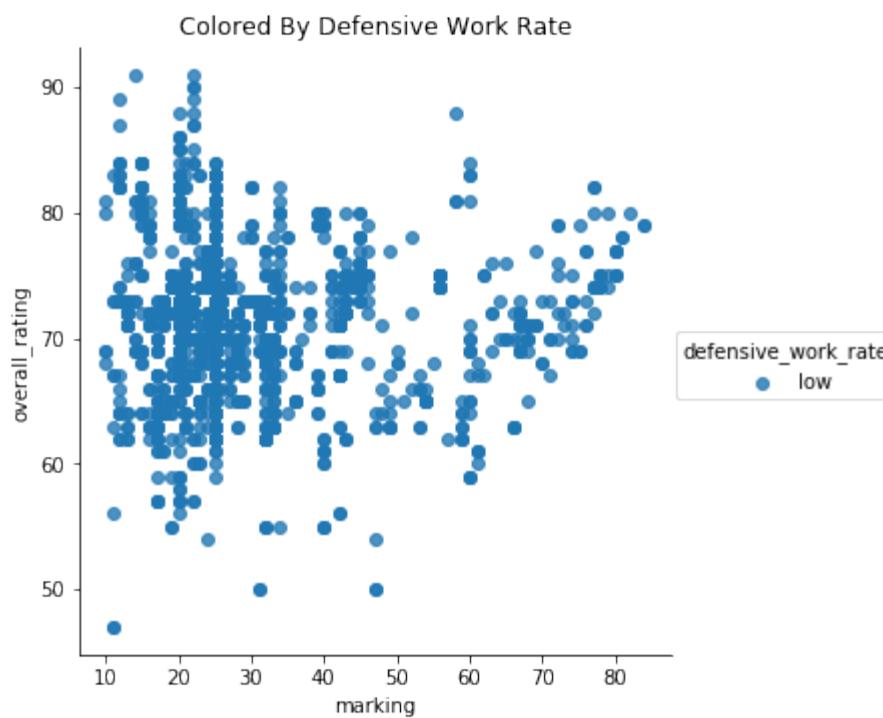


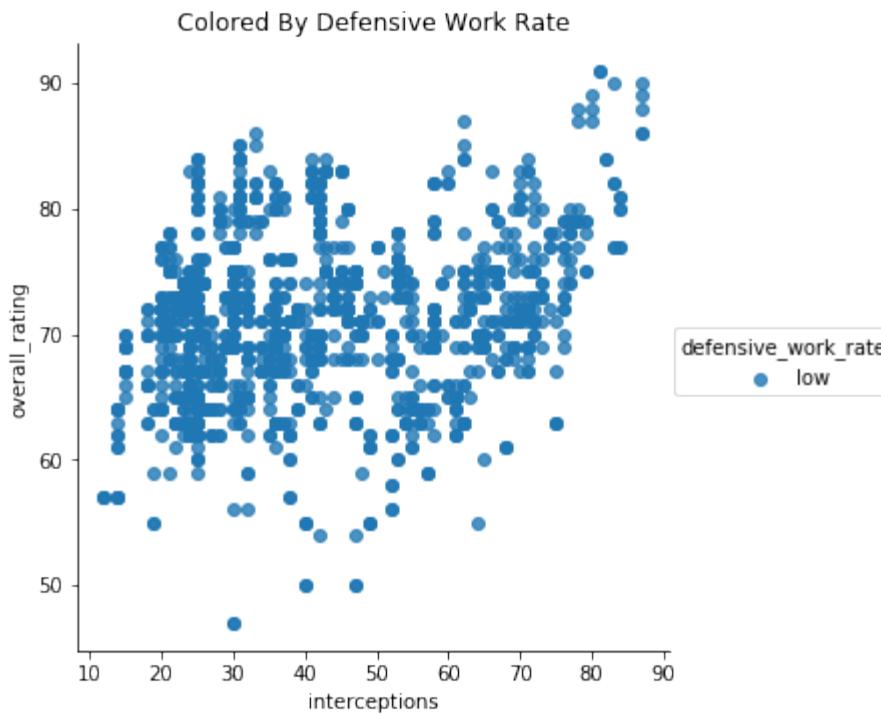
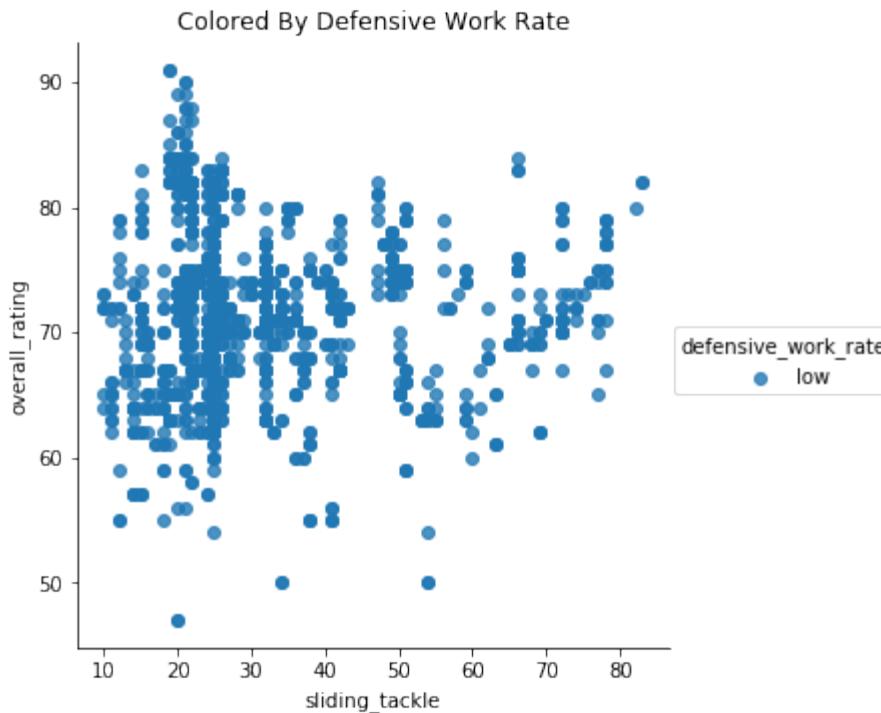






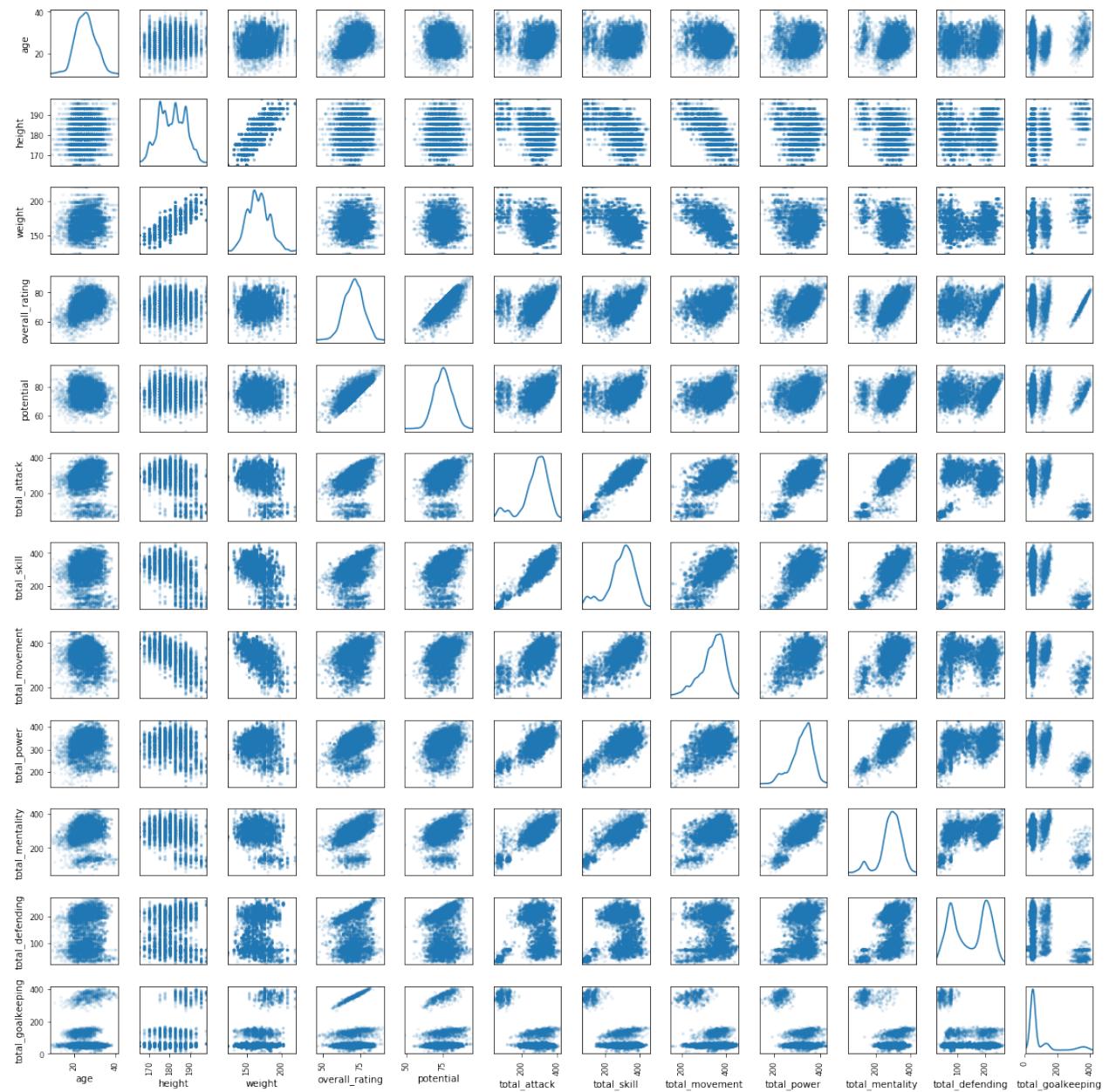






```
In [4]: df_totals=df_all_col[numeric_few_col]
pd.plotting.scatter_matrix(df_totals, alpha=0.1, figsize=(16, 16), diagonal='kde', range_padding =0.01)
plt.tight_layout()
plt.show()
plt.close()

total_cols = numeric_few_col + ['player_fifa_api_id', 'player_name']
df_t=df_all_col[total_cols]
df_t.to_csv('player_total_score_per_attributes_category.csv')
print(df_t.shape)
print (df_t.head())
```



(10898, 15)

	age	height	weight	overall_rating	potential	total_attack	\
1045	31.0	177.8	165	73.0	75.0	311.0	
1046	31.0	177.8	165	72.0	75.0	307.0	
1047	30.0	177.8	165	73.0	75.0	305.0	
1048	28.0	177.8	165	73.0	75.0	298.0	
1049	28.0	177.8	165	70.0	72.0	288.0	

	total_skill	total_movement	total_power	total_mentality	\
1045	319.0	375.0	355.0	351.0	
1046	318.0	375.0	355.0	372.0	
1047	316.0	377.0	357.0	372.0	
1048	311.0	372.0	355.0	370.0	
1049	335.0	366.0	346.0	356.0	

	total_defending	total_goalkeeping	player_fifa_api_id	\
1045	219.0	42.0	17880	
1046	216.0	143.0	17880	
1047	221.0	143.0	17880	

1048	221.0	141.0	17880
1049	212.0	102.0	17880

	player_fifa_api_id	player_name
1045	17880	Abel
1046	17880	Abel
1047	17880	Abel
1048	17880	Abel
1049	17880	Abel

```
In [16]: # Total Goalkeeping
vis=sns.lmplot(x='total_goalkeeping', y='overall_rating', hue = None, sharex=False, data=df_totals, \
                scatter=True, fit_reg=False, units=None, order=1, legend=True)
plt.title('Total Goalkeeping Versus Overall Rating')
plt.show()
plt.close()

df_totals_gk = df_totals[df_totals['total_goalkeeping'] > 250]
corr_gk=df_totals_gk[['total_goalkeeping','overall_rating']].corr()
print ('A Closer Look at the Goalkeeper Subgroup on the Far Right')
print(' total_goalkeeping > 250 ')
print(' ')
print('Strong Positive Linear Correlation: ')
print(corr_gk)

vis=sns.lmplot(x='total_goalkeeping', y='overall_rating', hue = None, sharex=False, data=df_totals_gk, \
                scatter=True, fit_reg=True, units=None, order=1, legend=True)
plt.title('Goalkeeper Subgroup: Total Goalkeeping Versus Overall Rating')
plt.show()
plt.close()

# Total Mentality
vis=sns.lmplot(x='total_mentality', y='overall_rating', hue = None, sharex=False, data=df_totals, \
                scatter=True, fit_reg=False, units=None, order=1, legend=True)
plt.title('Total Mentality Versus Overall Rating')
plt.show()
plt.close()

df_totals_non_gk = df_totals[df_totals['total_goalkeeping'] < 250]
corr_mentality=df_totals_non_gk[['total_mentality','overall_rating']].corr()
print ('A Closer Look at the Non_goalkeeper Subgroup on the Far Right')
print(' total_goalkeeping < 250 ')
print(' ')
print('Moderate Positive Linear Correlation: ')
print(corr_mentality)

vis=sns.lmplot(x='total_mentality', y='overall_rating', hue = None, sharex=False, data=df_totals_non_gk, \
                scatter=True, fit_reg=True, units=None, order=1, legend=True)
```

```

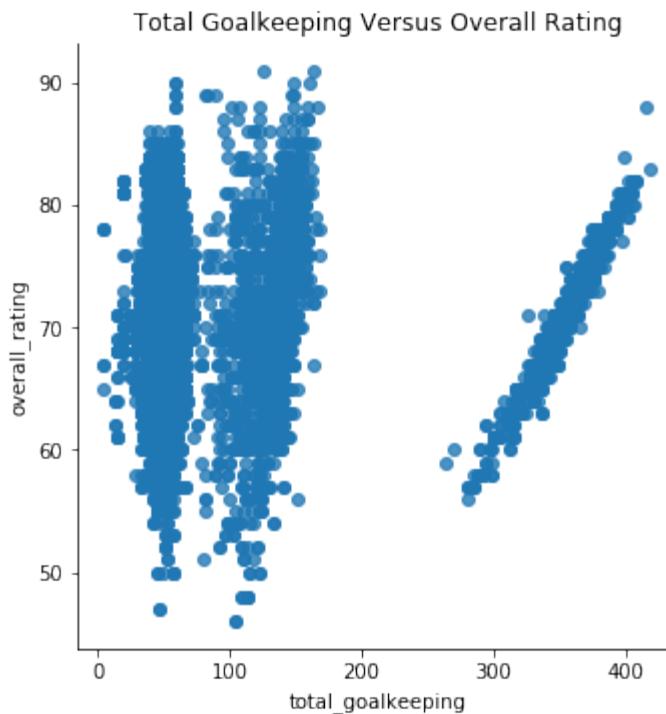
plt.title('Non_Goalkeeper Subgroup: Total Mentality Versus Overall Rating')
)
plt.show()
plt.close()

# Total Attack
vis=sns.lmplot(x='total_attack', y='overall_rating', hue = None, sharex=False,
                data=df_totals, \
                scatter=True, fit_reg=False, units=None, order=1, legend=True)
plt.title('Total Attack Versus Overall Rating')
plt.show()
plt.close()

df_totals_non_gk = df_totals[df_totals['total_goalkeeping'] < 250]
corr_attack=df_totals_non_gk[['total_attack','overall_rating']].corr()
print ('A Closer Look at the Non_goalkeeper Subgroup on the Far Right')
print(' total_goalkeeping < 250')
print(' ')
print ('Moderate Positive Linear Correlation: ')
print(corr_attack)

vis=sns.lmplot(x='total_attack', y='overall_rating', hue = None, sharex=False,
                data=df_totals_non_gk, \
                scatter=True, fit_reg=True, units=None, order=1, legend=True)
plt.title('Non_Goalkeeper Subgroup: Total Attack Versus Overall Rating')
plt.show()
plt.close()

```

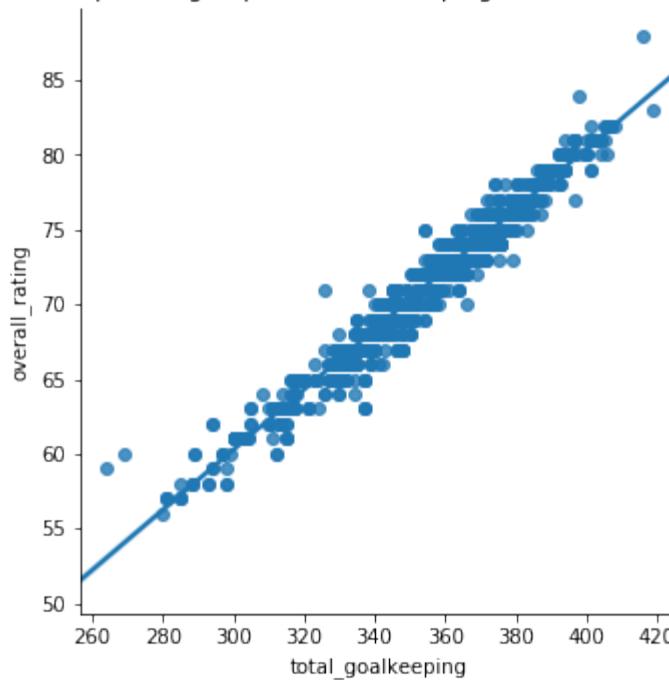


A Closer Look at the Goalkeeper Subgroup on the Far Right
 total_goalkeeping > 250

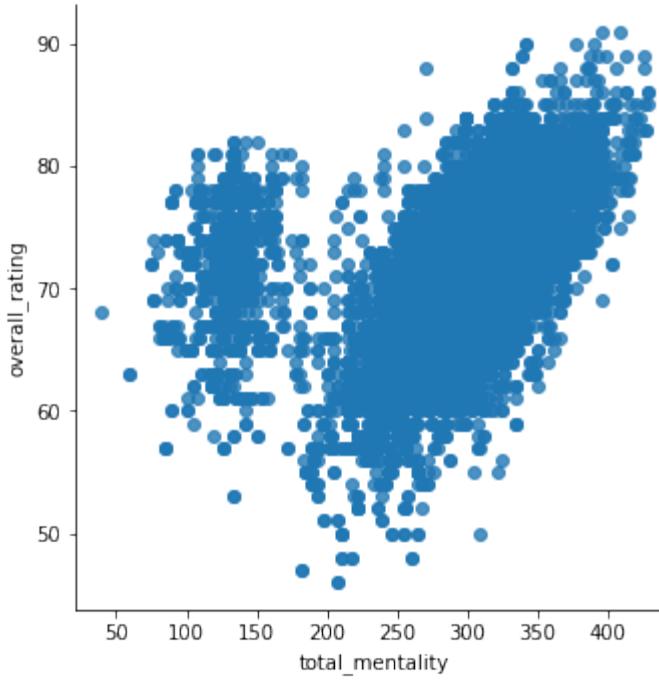
Strong Positive Linear Correlation:
 total_goalkeeping overall_rating

total_goalkeeping	1.000000	0.978269
overall_rating	0.978269	1.000000

Goalkeeper Subgroup: Total Goalkeeping Versus Overall Rating



Total Mentality Versus Overall Rating

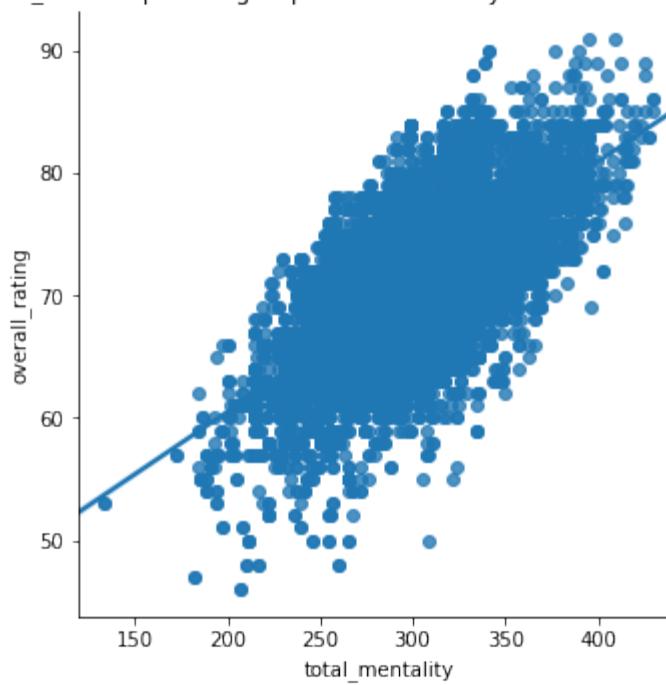


A Closer Look at the Non_goalkeeper Subgroup on the Far Right
 $\text{total_goalkeeping} < 250$

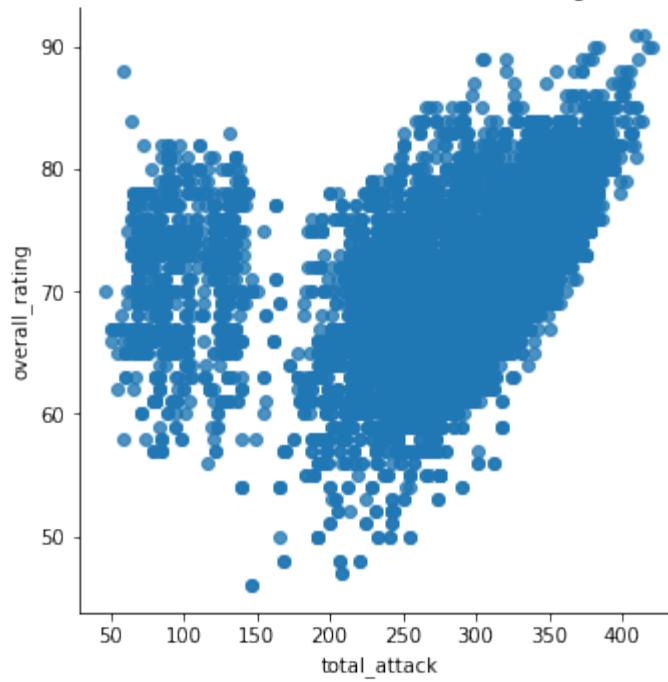
Moderate Positive Linear Correlation:

	total_mentality	overall_rating
total_mentality	1.00000	0.67607
overall_rating	0.67607	1.00000

Non_Goalkeeper Subgroup: Total Mentality Versus Overall Rating



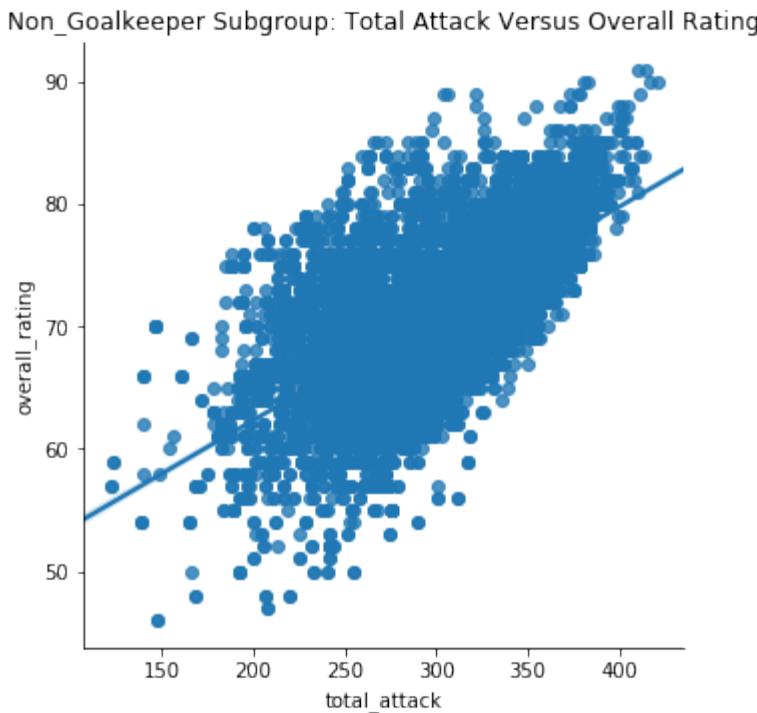
Total Attack Versus Overall Rating



A Closer Look at the Non_goalkeeper Subgroup on the Far Right
 $\text{total_goalkeeping} < 250$

Moderate Positive Linear Correlation:

	total_attack	overall_rating
total_attack	1.000000	0.628757
overall_rating	0.628757	1.000000



Interpret correlation coefficient

Exactly -1. A perfect downhill (negative) linear relationship

- -0.70. A strong downhill (negative) linear relationship
- -0.50. A moderate downhill (negative) relationship
- -0.30. A weak downhill (negative) linear relationship
- 0. No linear relationship

```
In [9]: # save correlation coefficient for dataset to csv
df_corr = df_unscaled_data.corr()
df_corr.to_csv('df_corr.csv')

df_gk=df_unscaled_data.loc[df_unscaled_data['gk_diving']>50]
df_gk_corr = df_gk.corr()
df_gk_corr.to_csv('df_gk_corr.csv')

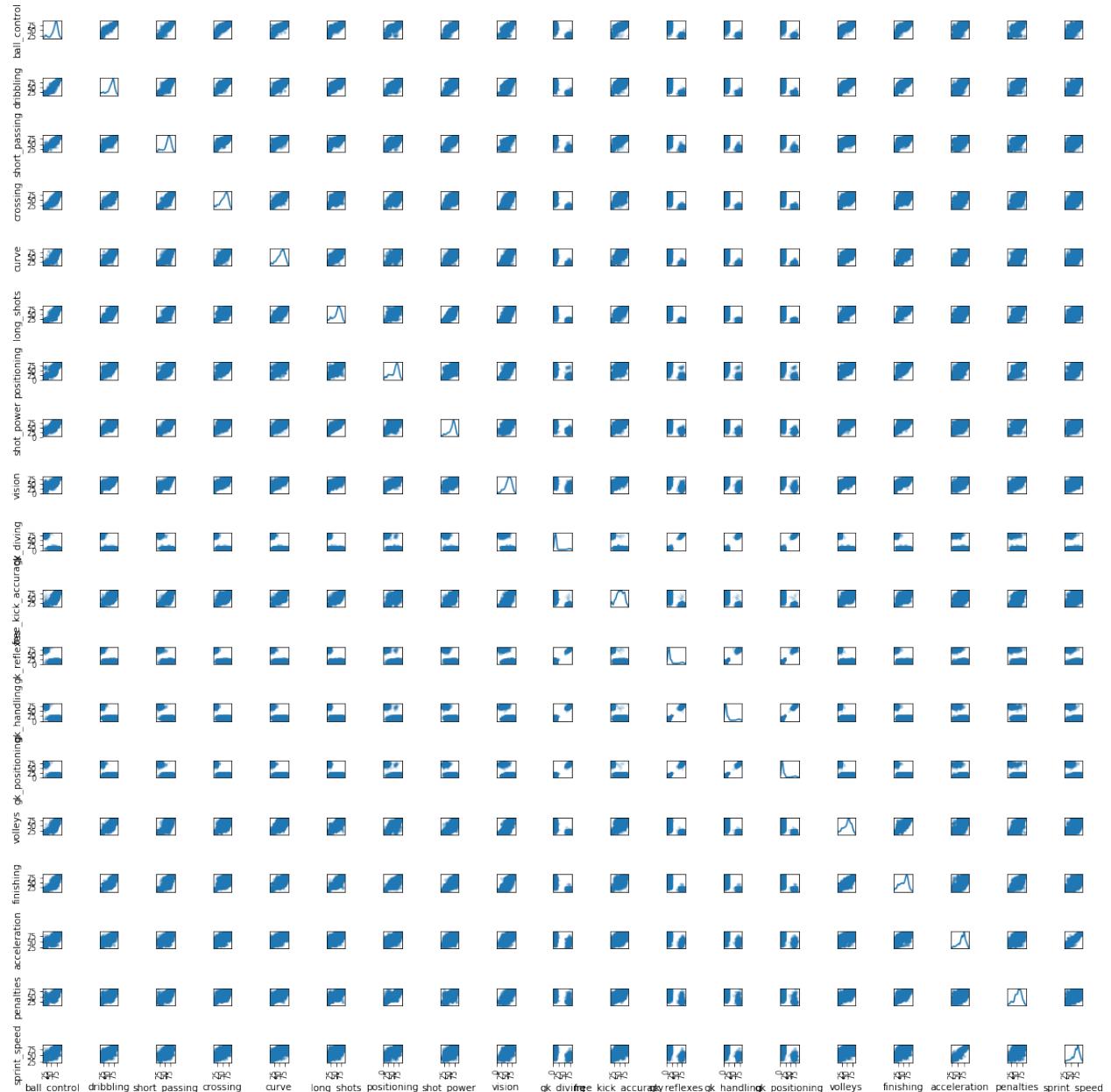
df_non_gk=df_unscaled_data.loc[df_unscaled_data['gk_diving']<50]
df_non_gk_corr = df_non_gk.corr()
df_non_gk_corr.to_csv('df_non_gk_corr.csv')
```

```
In [31]: print('Unscaled Data Scatter Matrix 1')
print('PC1 sorted components: ', loading_scores_PC1_sorted.index[0:19])
#col_of_interest = ['ball_control', 'dribbling', 'short_passing', 'crossing',
#                   'curve', 'long_shots', 'positioning', 'shot_power', 'vision', 'gk_diving',
#                   'free_kick_accuracy', 'gk_reflexes', 'gk_handling']
col_of_interest = loading_scores_PC1_sorted.index[0:19]
df_col_of_interest= df_unscaled_data[col_of_interest] # scatter matrix for
# columns of interest

pd.plotting.scatter_matrix(df_col_of_interest, alpha=0.1, figsize=(16, 16))
```

```
, diagonal='kde', range_padding = 0.01)
plt.tight_layout()
plt.show()
plt.close()
```

Unscaled Data Scatter Matrix 1
PC1 sorted components: Index(['ball_control', 'dribbling', 'short_passing',
'crossing', 'curve',
'long_shots', 'positioning', 'shot_power', 'vision', 'gk_diving',
'free_kick_accuracy', 'gk_reflexes', 'gk_handling', 'gk_positioning',
',
'velleys', 'finishing', 'acceleration', 'penalties', 'sprint_speed'],
dtype='object')



In [39]:

```
print('Unscaled Data Scatter Matrix 2')
print('PC1 sorted components: ', loading_scores_PC1_sorted.index[19:38])
#col_of_interest = ['ball_control', 'dribbling', 'short_passing', 'crossin
g', 'curve', 'long_shots', 'positioning', 'shot_power', 'vision', 'gk_divin
```

```

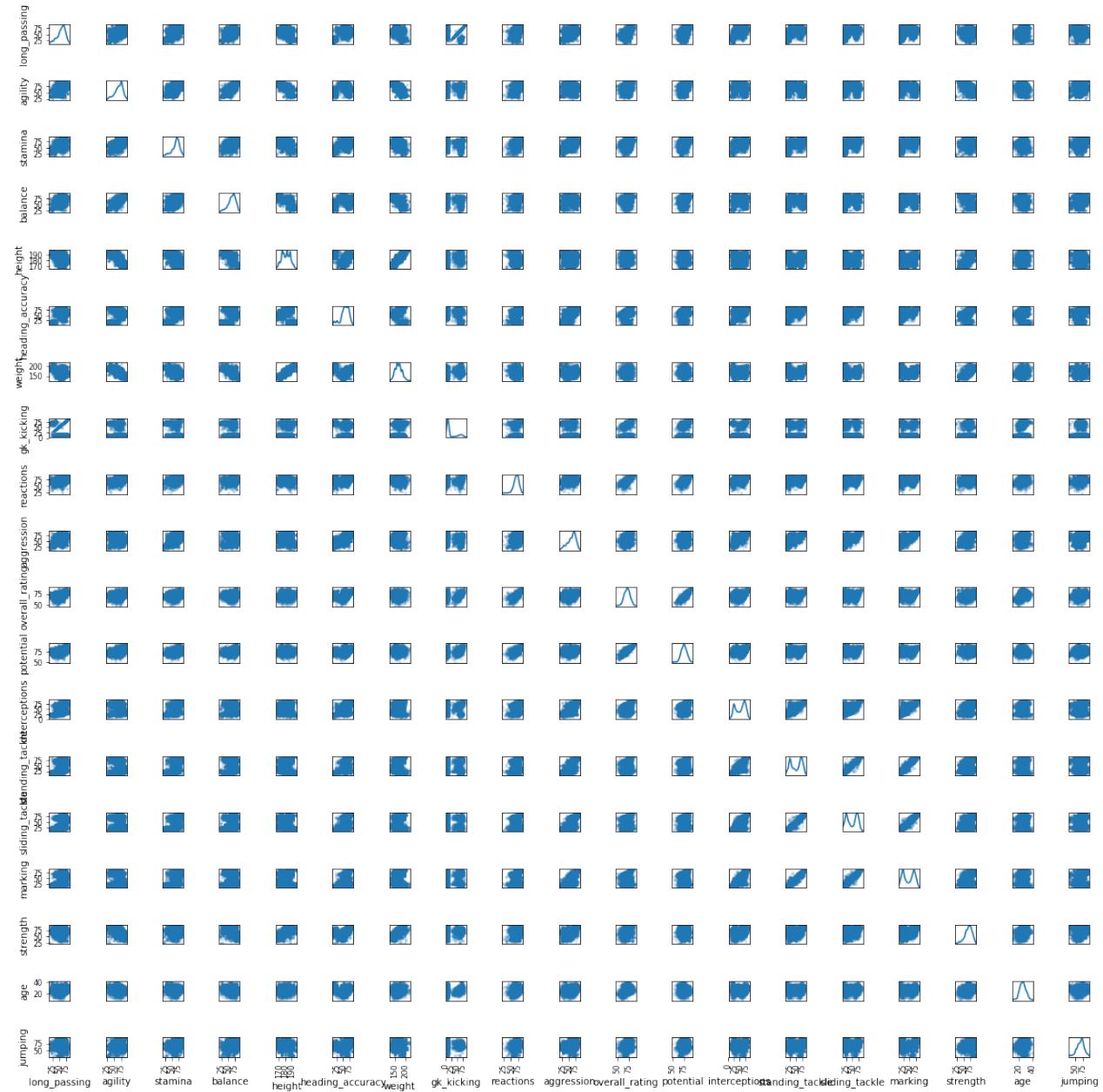
g', 'free_kick_accuracy', 'gk_reflexes', 'gk_handling']
col_of_interest = loading_scores_PC1_sorted.index[19:38]
df_col_of_interest= df_unscaled_data[col_of_interest] # scatter matrix for
# columns of interest

pd.plotting.scatter_matrix(df_col_of_interest, alpha=0.1, figsize=(16, 16)
, diagonal='kde',range_padding = 0.01)
plt.tight_layout()
plt.show()
plt.close()

```

Unscaled Data Scatter Matrix 2

PC1 sorted components: Index(['long_passing', 'agility', 'stamina', 'balance', 'height',
'heading_accuracy', 'weight', 'gk_kicking', 'reactions', 'aggression',
'overall_rating', 'potential', 'interceptions', 'standing_tackle',
'sliding_tackle', 'marking', 'strength', 'age', 'jumping'],
dtype='object')



```
In [36]: # create distribution plot for all features

final_col = ['player_fifa_api_id', 'preferred_foot', 'attacking_work_rate',
'defensive_work_rate'] + numeric_col
print(final_col)
df_final=df_all_col[final_col]
...
df_final=df_all_col['player_fifa_api_id', 'preferred_foot', 'attacking_work_rate',
'defensive_work_rate', 'age', \
'height', 'weight', 'overall_rating', 'potential', 'crossing', 'finishing', 'heading_accuracy', \
'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_accuracy', 'long_passing', \
'ball_control', 'acceleration', 'sprint_speed', 'agility', 'reactions', 'balance', 'shot_power', \
'jumping', 'stamina', 'strength', 'long_shots', 'aggression', 'interceptions', 'positioning', \
'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle', 'gk_diving', 'gk_handling', \
'gk_kicking', 'gk_positioning', 'gk_reflexes']
...
print(df_final.head())
print(len(final_col))
df_final.to_csv("df_final.csv")

# distplot for goalkeeping attributes
fig = plt.figure(figsize=(24,18))
ax1 = fig.add_subplot(321)
ax2 = fig.add_subplot(322)
ax3 = fig.add_subplot(323)
ax4 = fig.add_subplot(324)
ax5 = fig.add_subplot(325)
vis1=sns.distplot (df_all_col['gk_diving'], bins=30, ax=ax1)
vis2=sns.distplot (df_all_col['gk_handling'], bins=30, ax=ax2)
vis3=sns.distplot (df_all_col['gk_kicking'], bins=30, ax=ax3)
vis4=sns.distplot (df_all_col['gk_positioning'], bins=30, ax=ax4)
vis5=sns.distplot (df_all_col['gk_reflexes'], bins=30, ax=ax5)
plt.show()
plt.close()

# distplot for defending attributes
fig = plt.figure(figsize=(18,5))
ax6 = fig.add_subplot(131)
ax7 = fig.add_subplot(132)
ax8 = fig.add_subplot(133)
vis6=sns.distplot (df_all_col['marking'], bins=30, ax=ax6)
vis7=sns.distplot (df_all_col['standing_tackle'], bins=30, ax=ax7)
vis8=sns.distplot (df_all_col['sliding_tackle'], bins=30, ax=ax8)
plt.show()
plt.close()

# distplot
for i in range (4,42) :
    sns.distplot (df_all_col[final_col[i]], bins=30)
    plt.title('Distribution Plot')
    plt.show()
```

```

plt.close()

['player_fifa_api_id', 'preferred_foot', 'attacking_work_rate', 'defensive_work_rate', 'age', 'height', 'weight', 'overall_rating', 'potential', 'crossing', 'finishing', 'heading_accuracy', 'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_accuracy', 'long_passing', 'ball_control', 'acceleration', 'sprint_speed', 'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'stamina', 'strength', 'long_shots', 'aggression', 'interceptions', 'positioning', 'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle', 'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning', 'gk_reflexes']

      player_fifa_api_id  player_fifa_api_id preferred_foot \
1045              17880                  17880        right
1046              17880                  17880        right
1047              17880                  17880        right
1048              17880                  17880        right
1049              17880                  17880        right

      attacking_work_rate  defensive_work_rate    age   height   weight \
1045             None                   o  31.0  177.8   165
1046             None                   o  31.0  177.8   165
1047             None                   o  30.0  177.8   165
1048             None                   o  28.0  177.8   165
1049             None                   o  28.0  177.8   165

      overall_rating  potential     ...  vision  penalties  marking \
\ 1045          73.0       75.0     ...    75.0      66.0     73.0
1046          72.0       75.0     ...    75.0      75.0     72.0
1047          73.0       75.0     ...    75.0      75.0     74.0
1048          73.0       75.0     ...    75.0      76.0     74.0
1049          70.0       72.0     ...    75.0      83.0     70.0

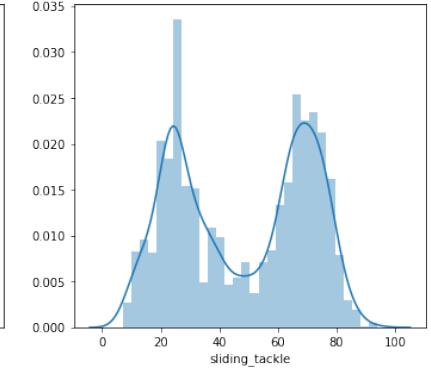
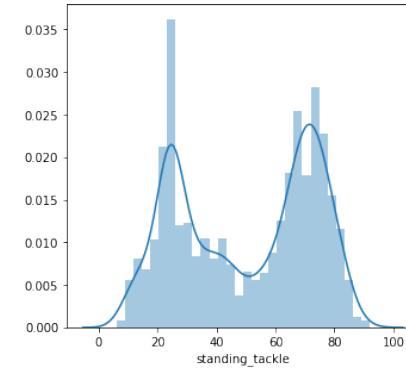
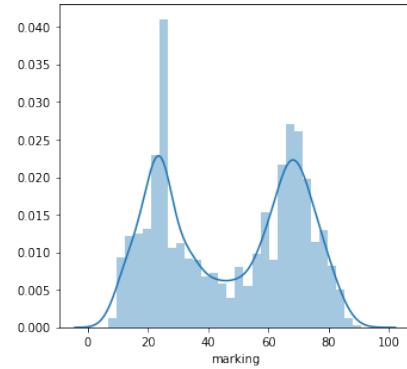
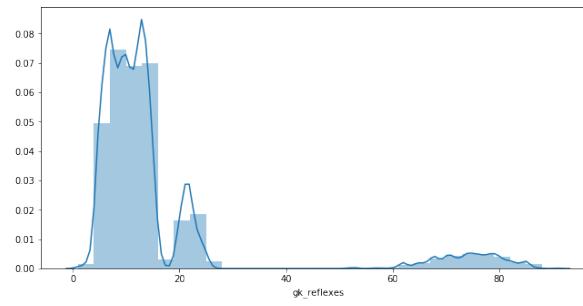
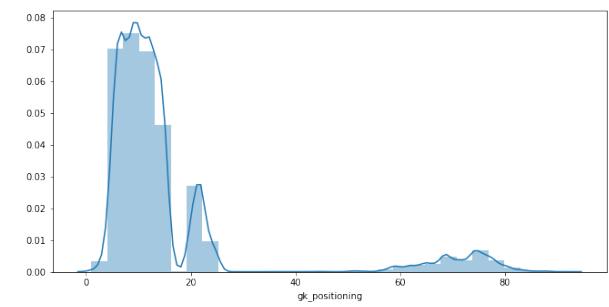
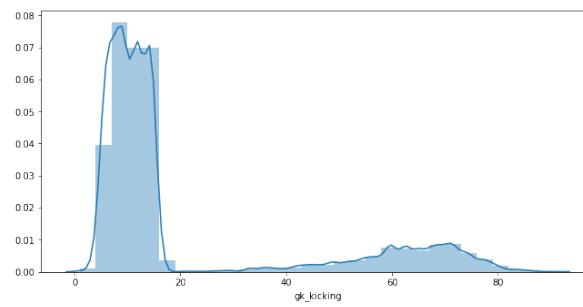
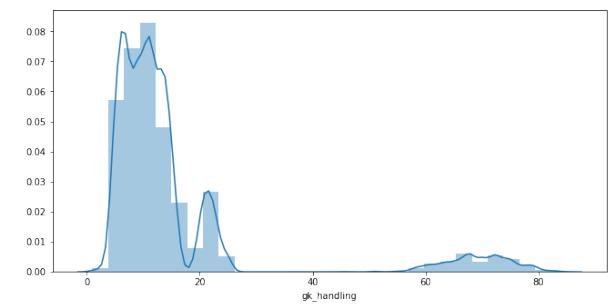
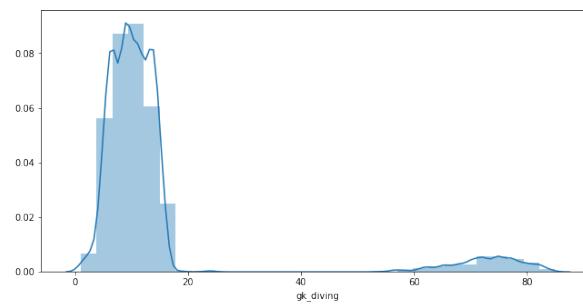
      standing_tackle  sliding_tackle  gk_diving  gk_handling  gk_kicking \
\ 1045          74.0           72.0      7.0        5.0        7.0
1046          72.0           72.0      9.0       20.0       74.0
1047          75.0           72.0      9.0       20.0       74.0
1048          75.0           72.0      9.0       20.0       72.0
1049          70.0           72.0      9.0       10.0       63.0

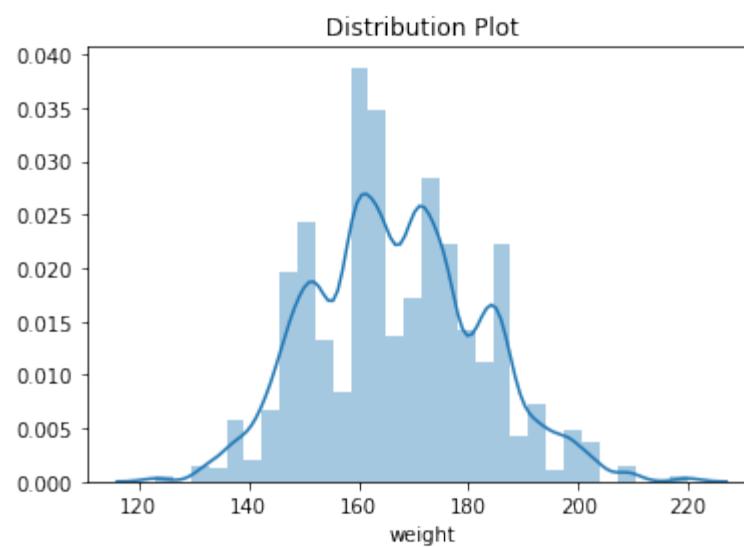
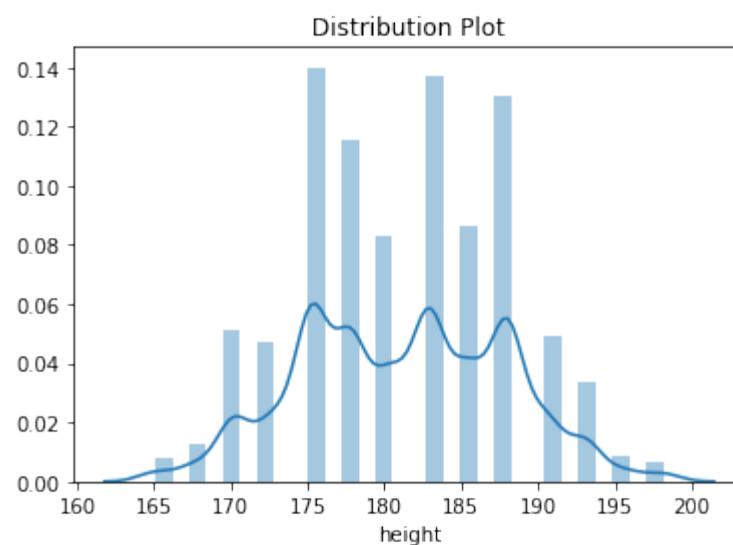
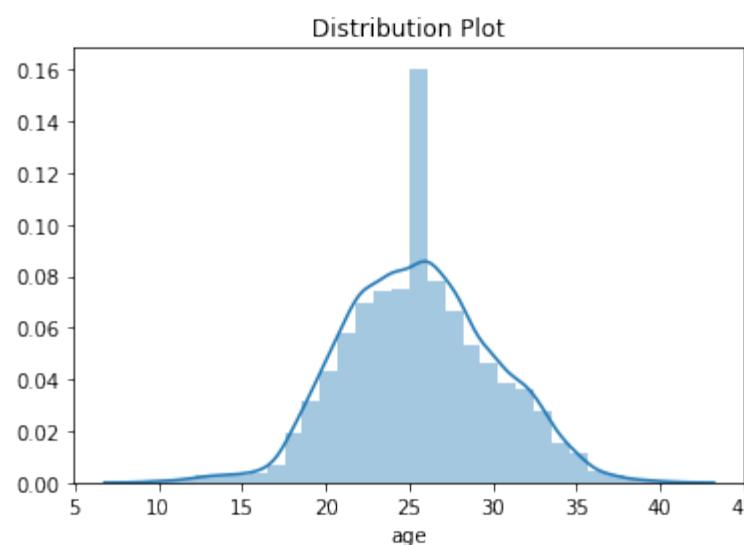
      gk_positioning  gk_reflexes
1045          13.0         10.0
1046          20.0         20.0
1047          20.0         20.0
1048          20.0         20.0
1049           8.0         12.0

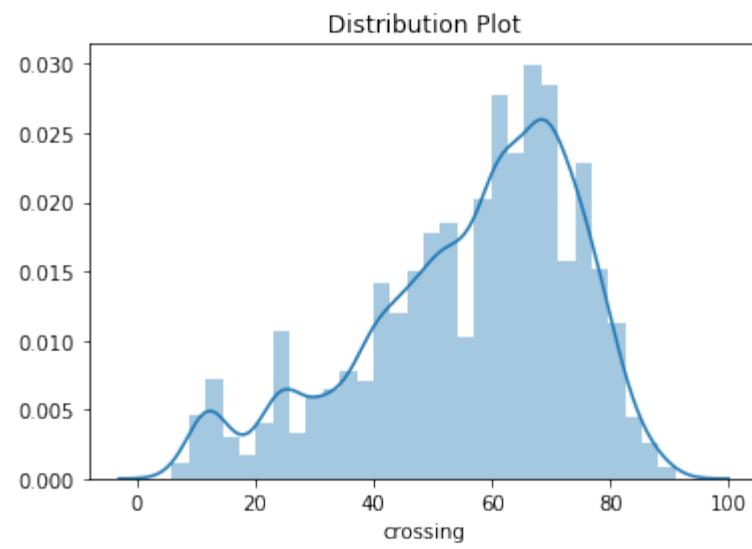
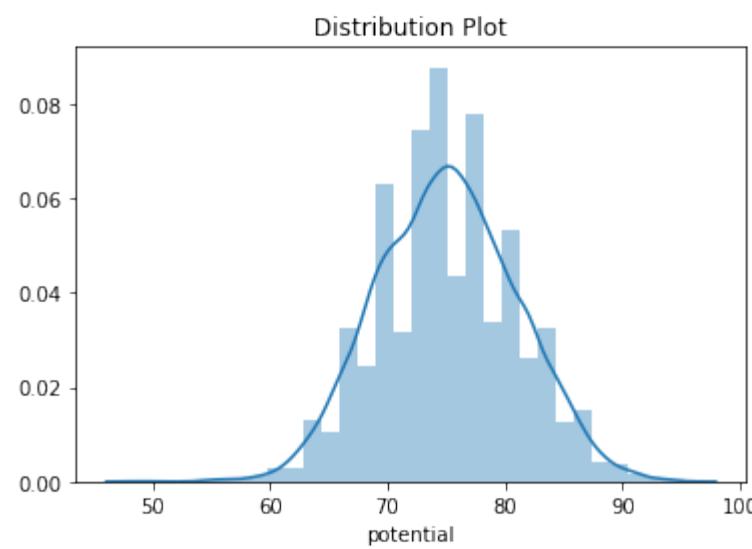
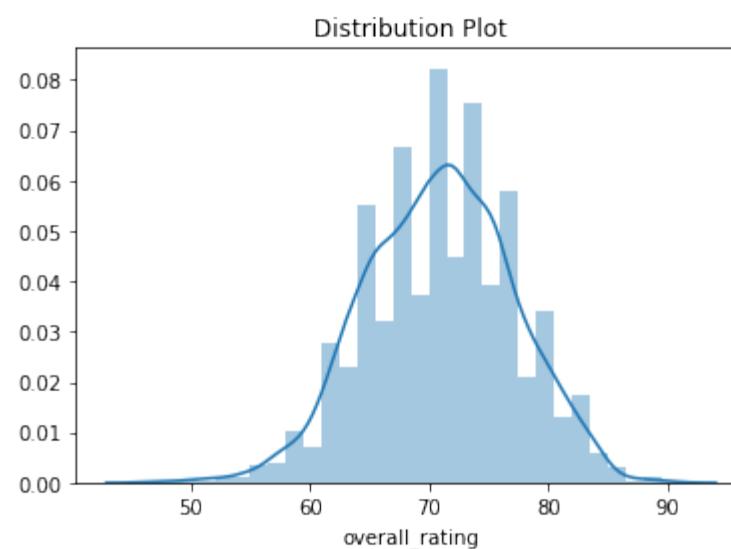
```

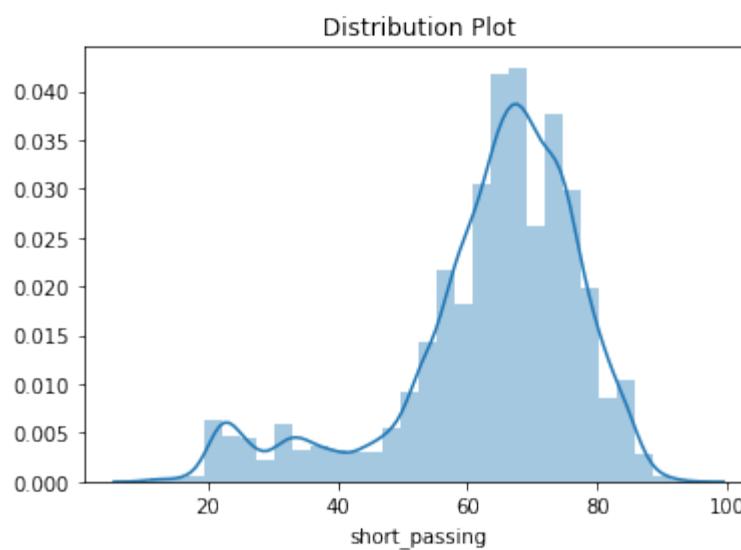
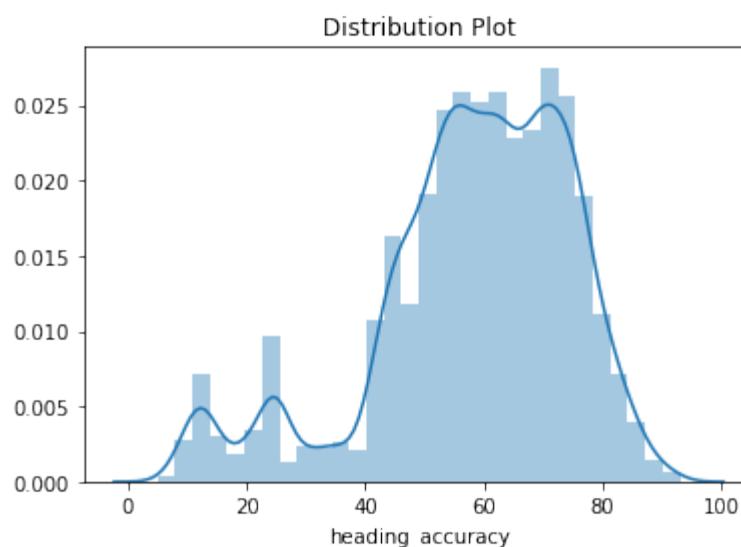
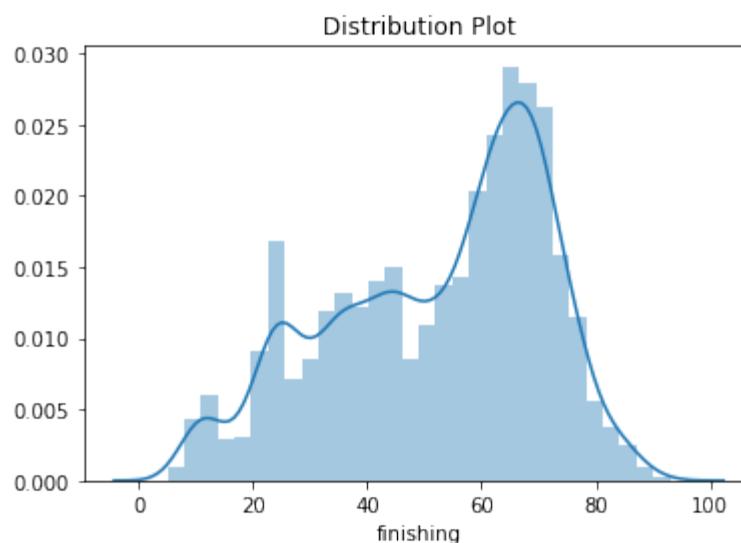
[5 rows x 43 columns]

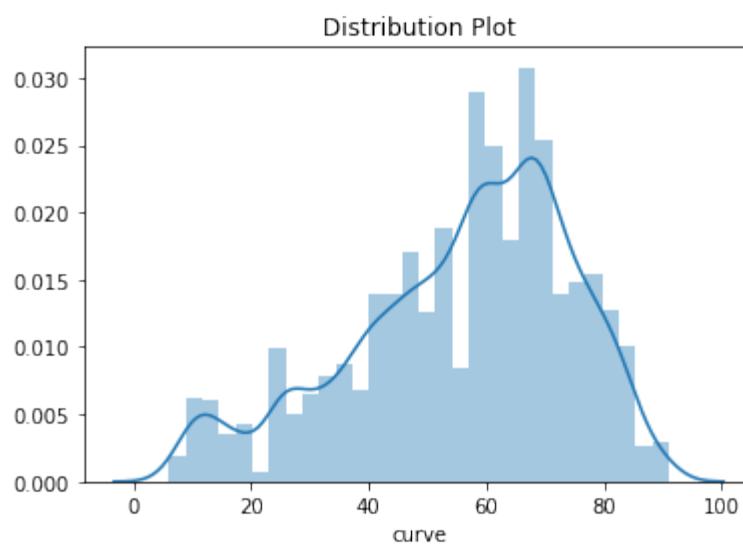
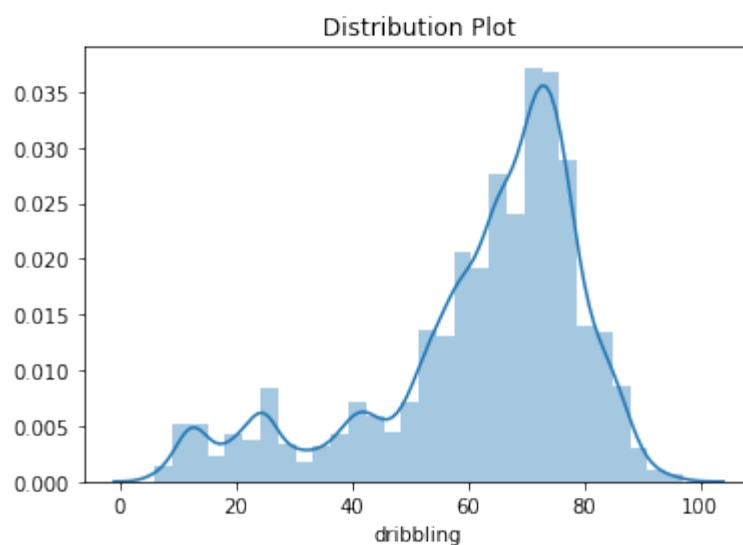
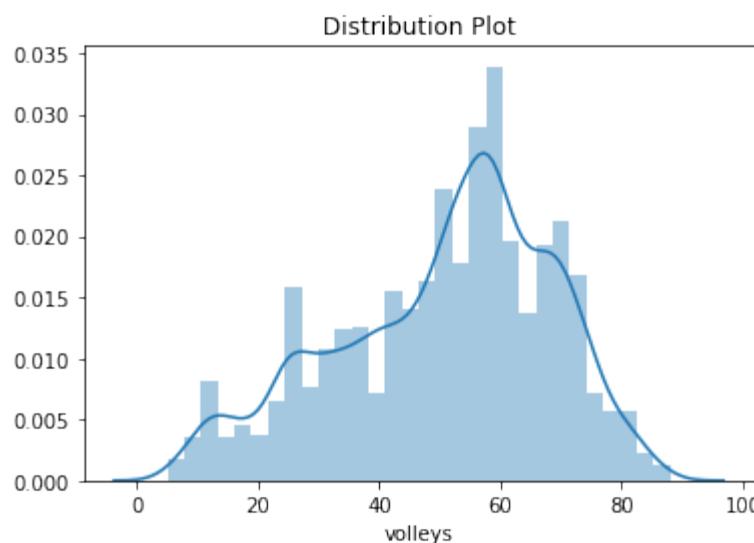
42

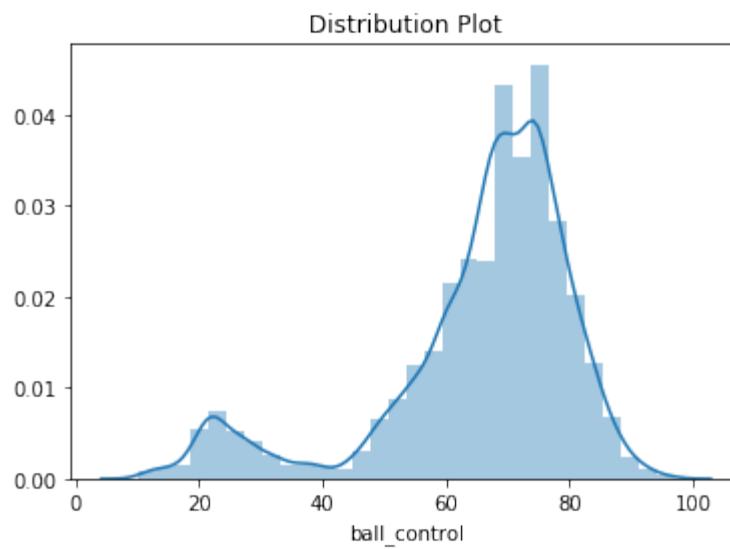
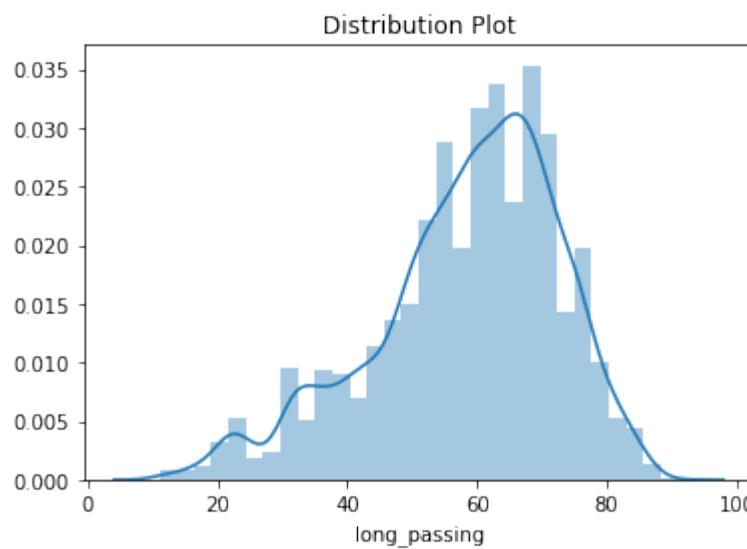
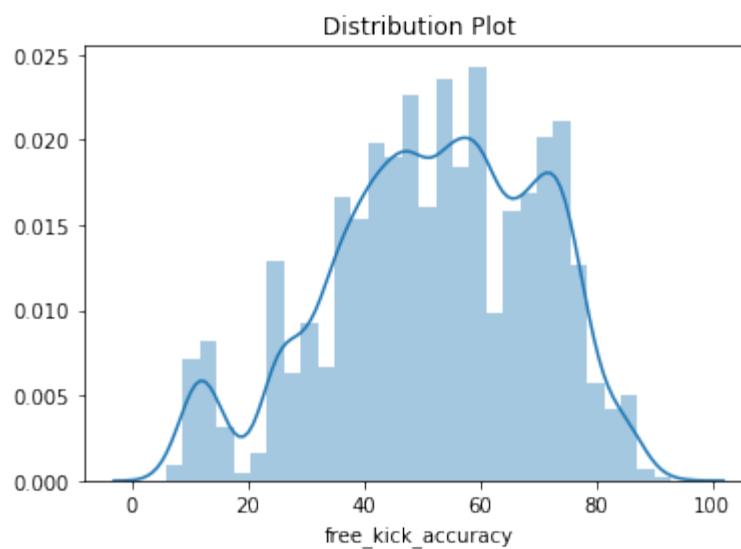


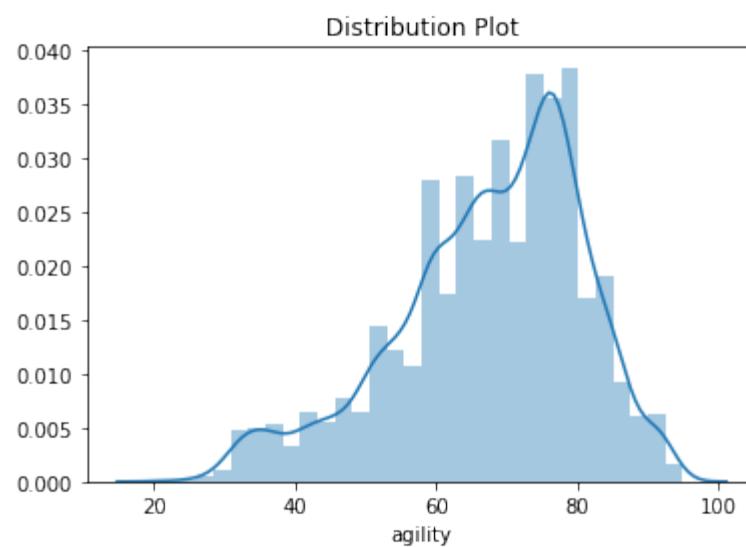
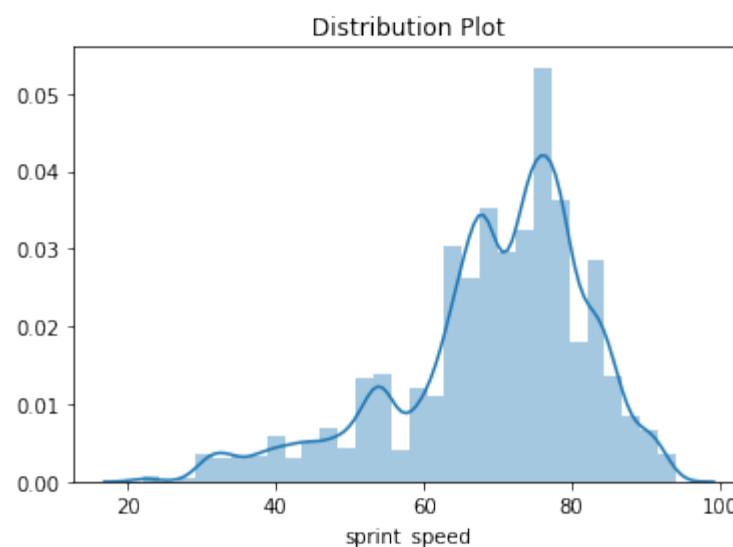
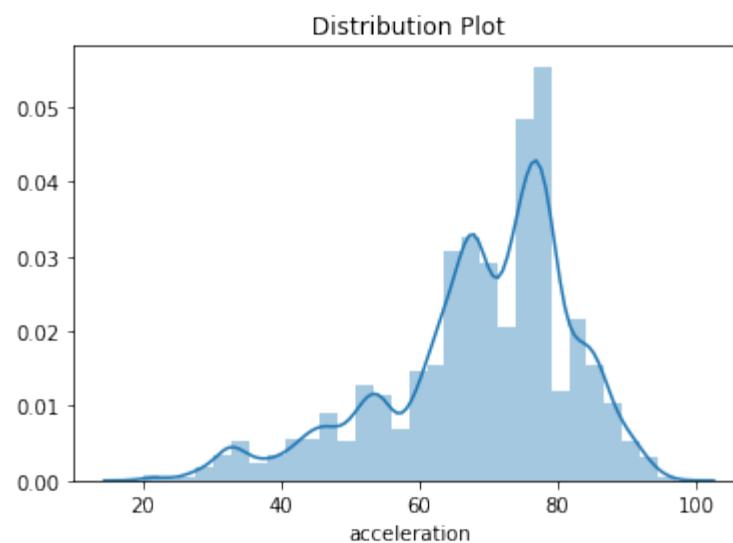


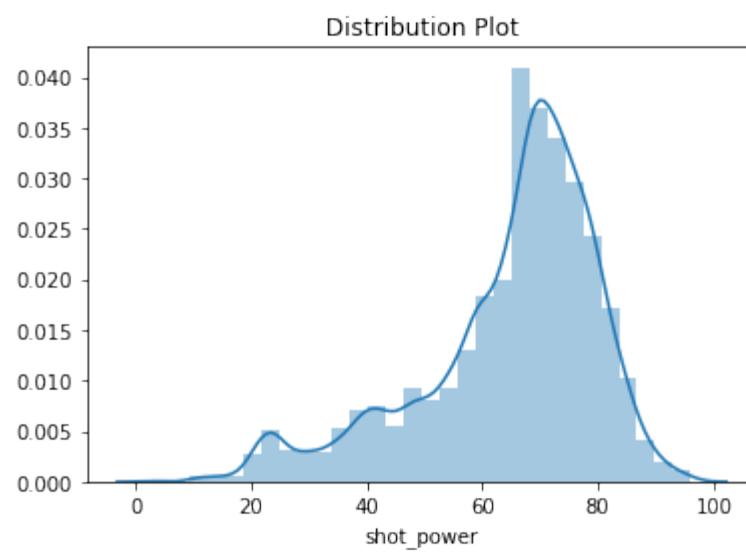
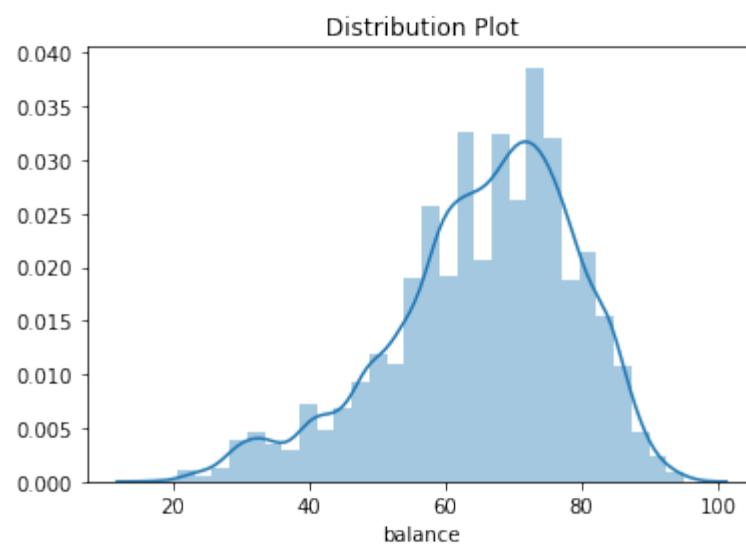
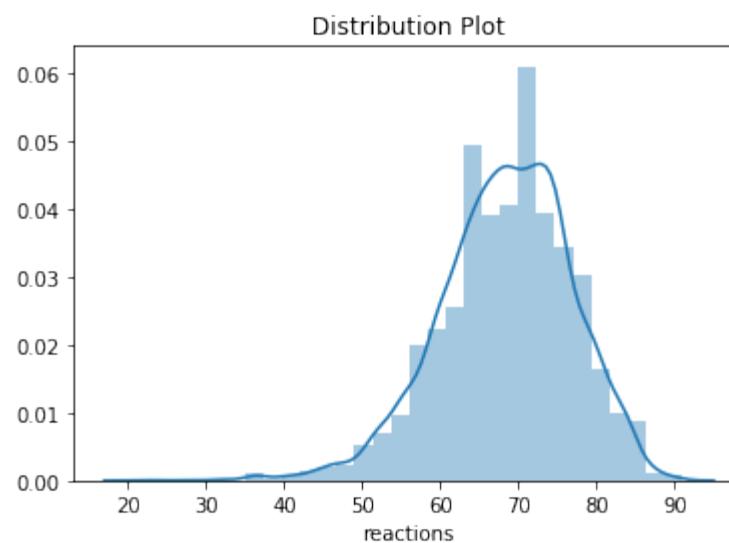


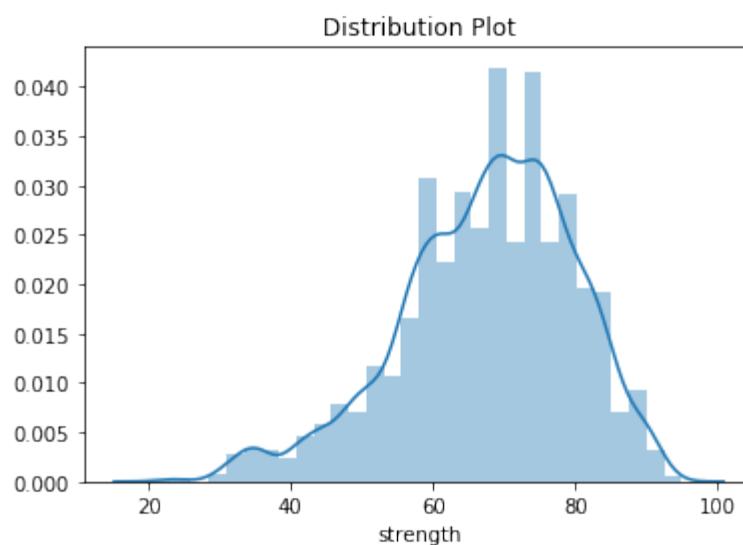
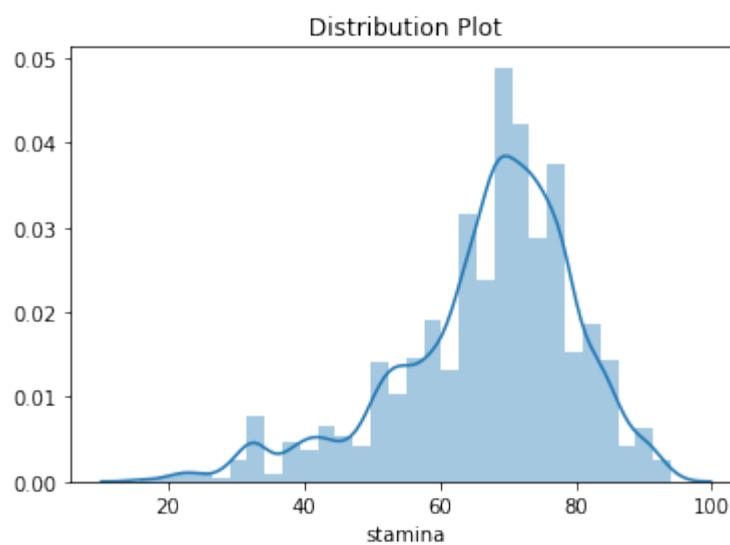
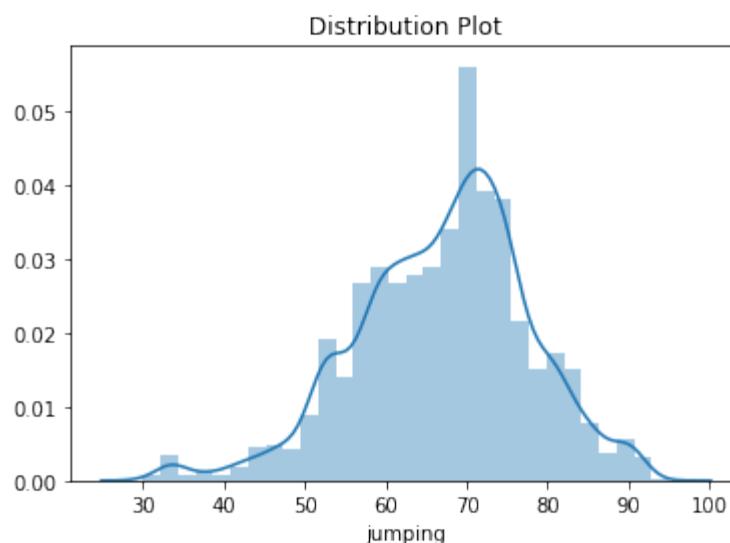


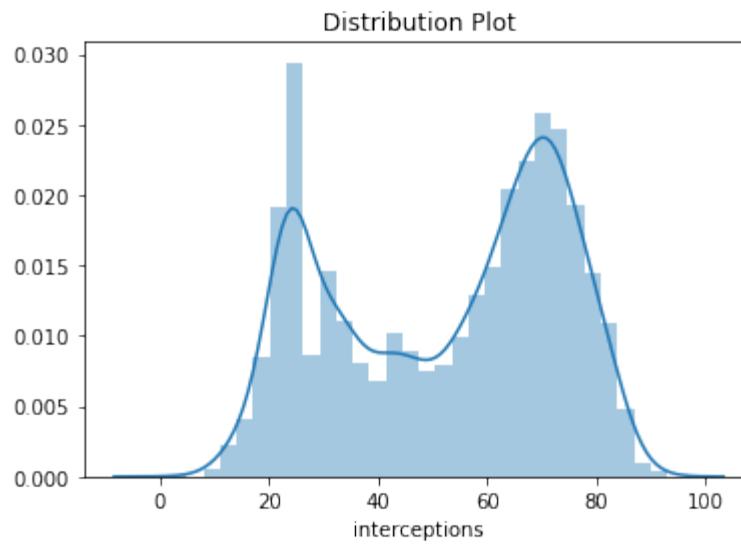
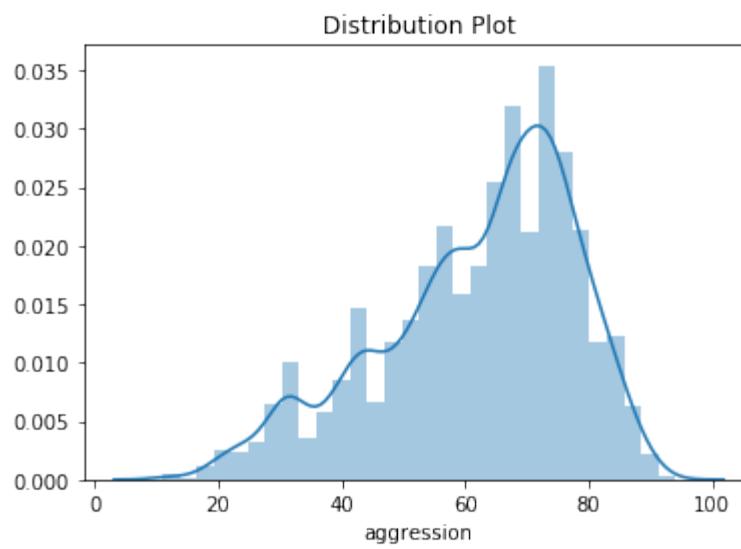
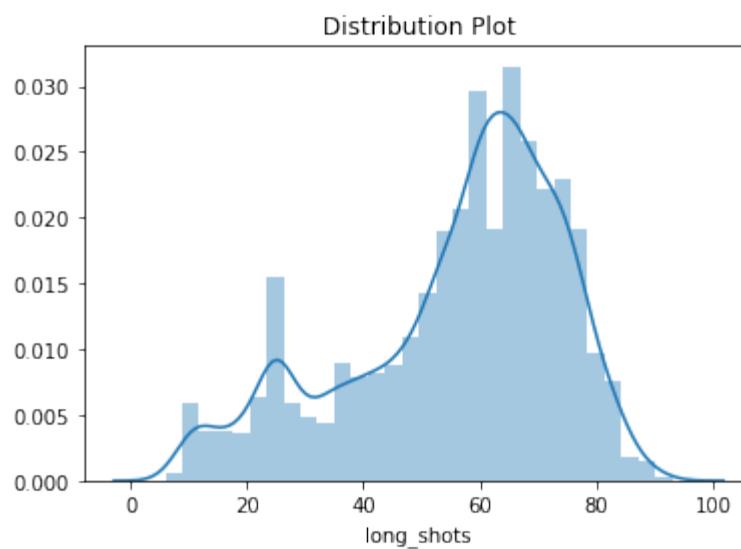


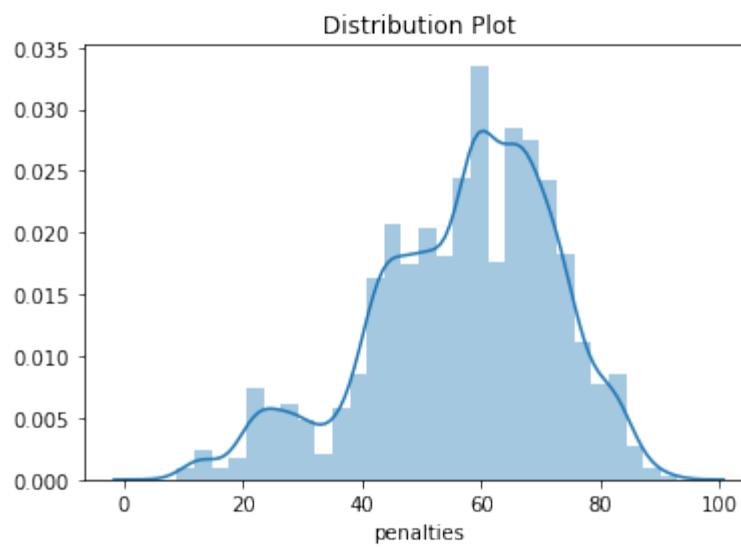
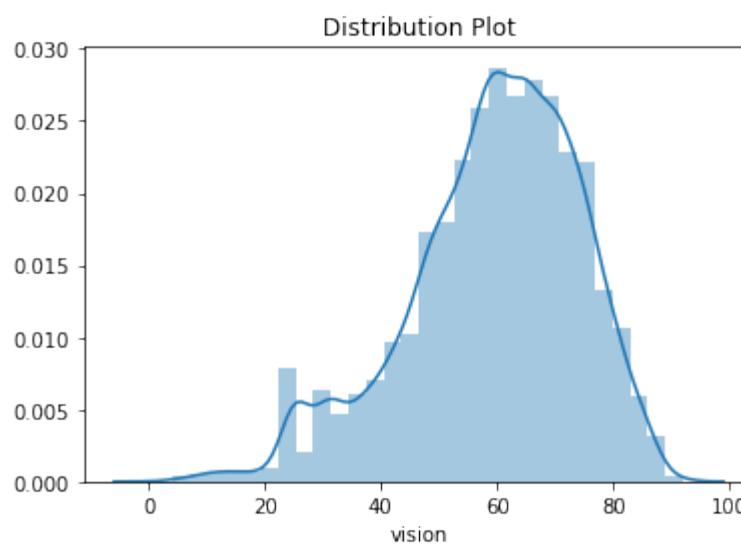
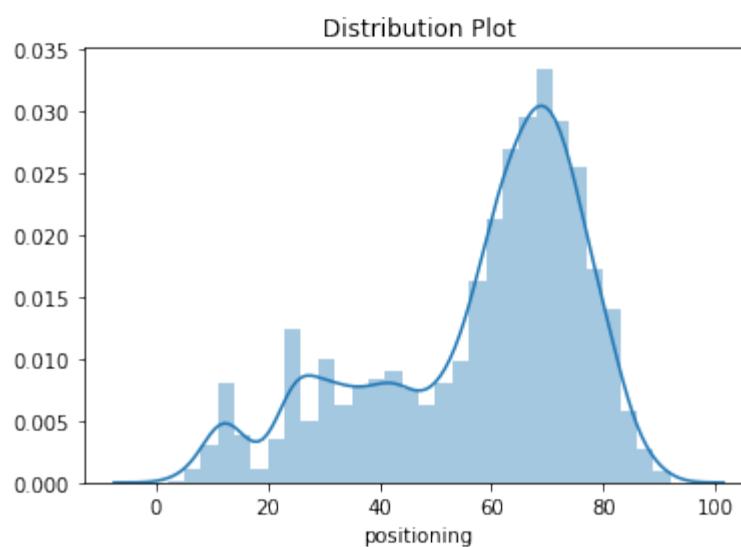


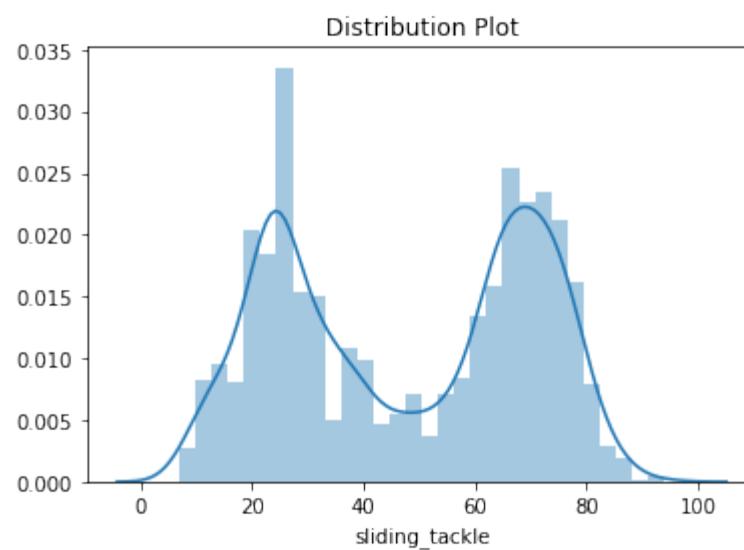
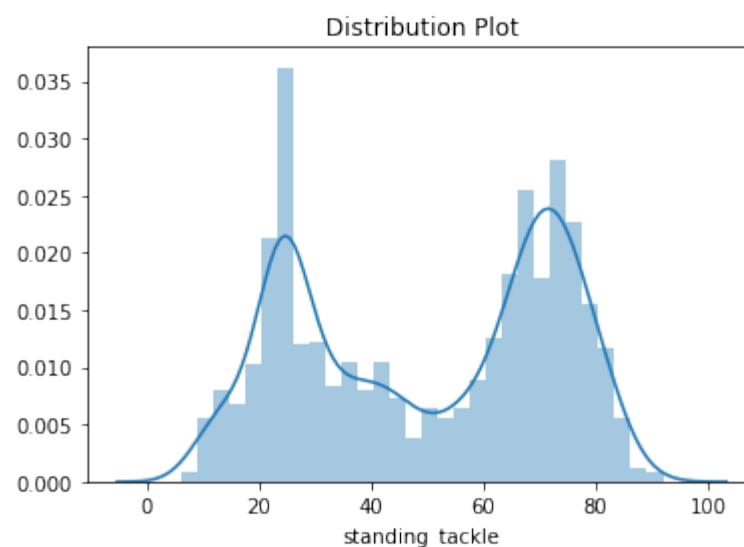
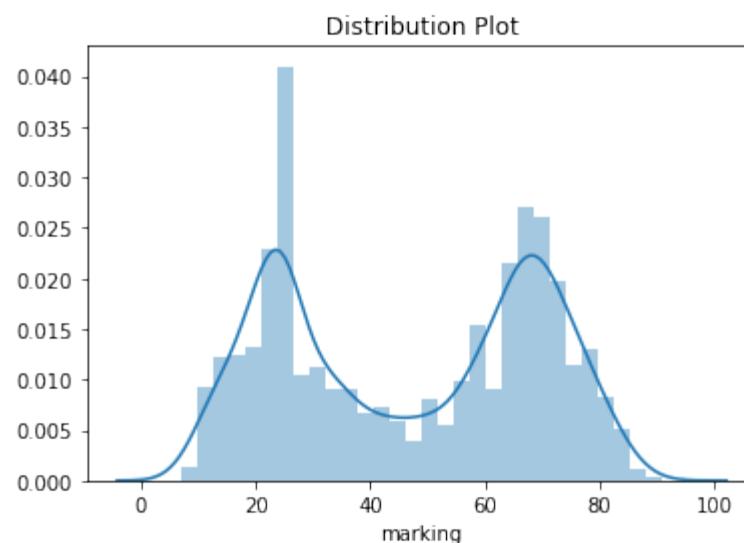




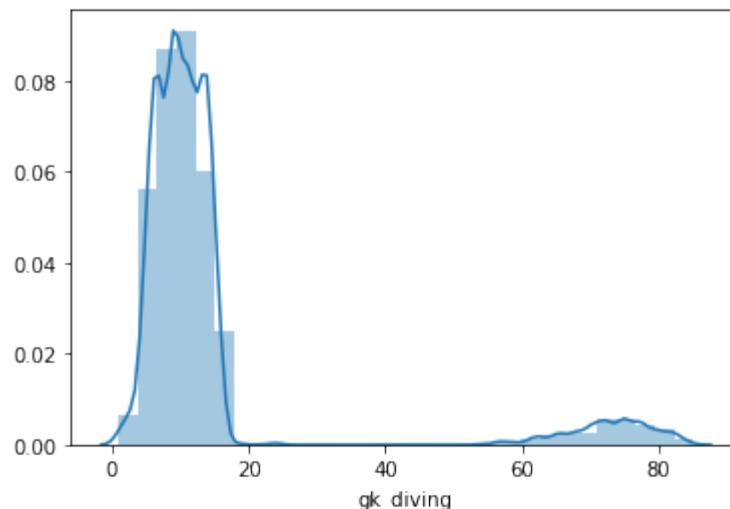




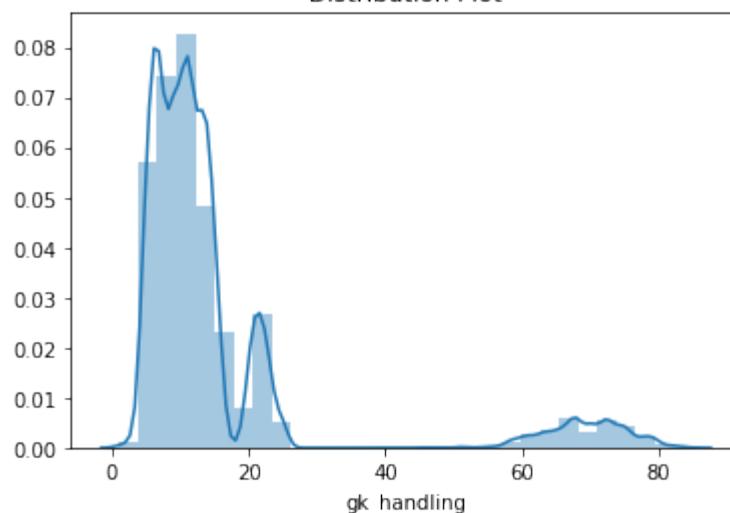




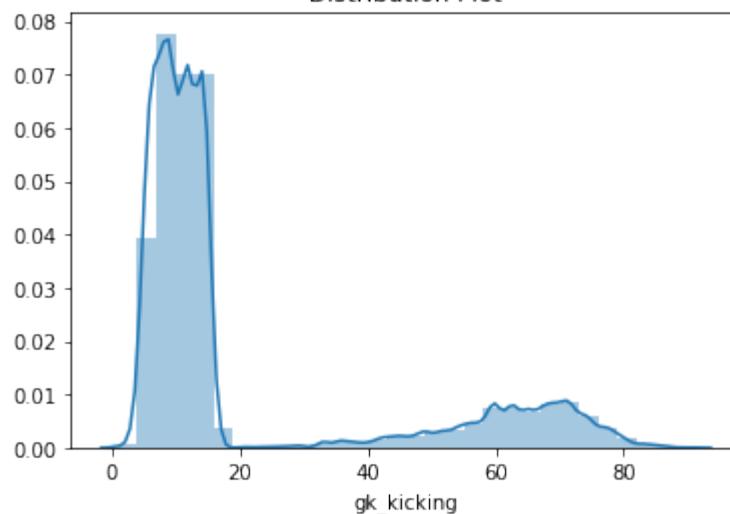
Distribution Plot

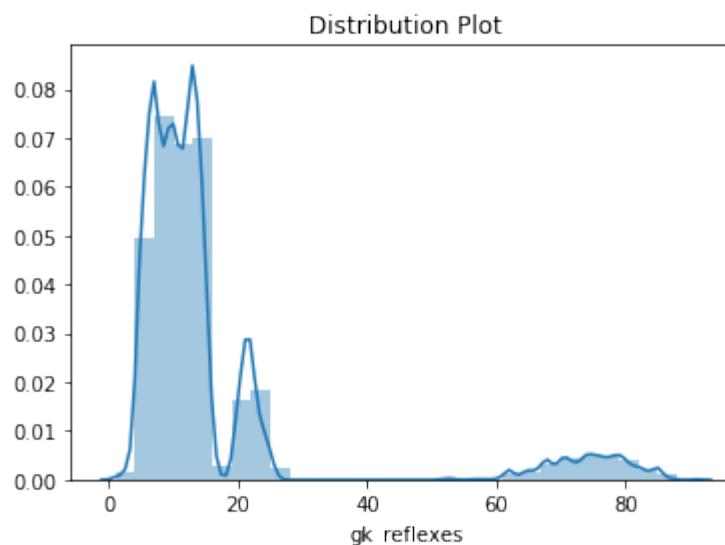
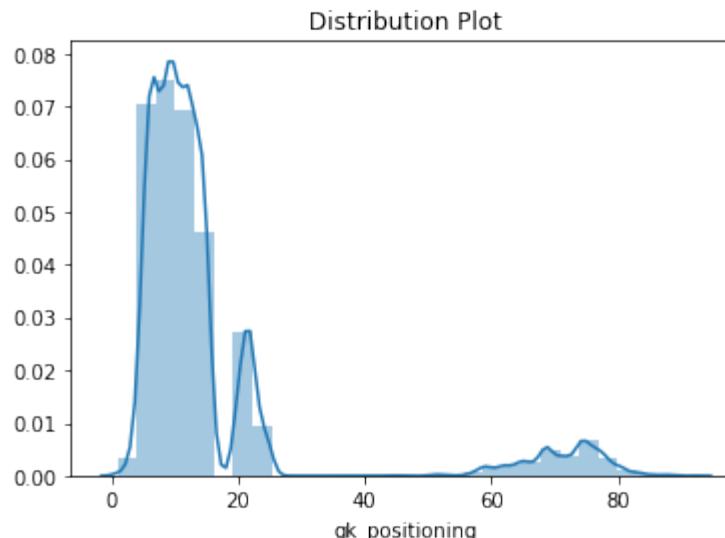


Distribution Plot



Distribution Plot





```
In [15]: #####
def print_corr (x_attribute, y_attribute, df, title):
    corr15=df[[x_attribute, y_attribute]].corr()
    print(title)
    print(corr15)
    vis15=sns.lmplot( x=x_attribute, y=y_attribute, hue=None, sharex=False
, data=df, scatter=True, fit_reg=True, units=None, order=1, legend=True)
    plt.show()
    plt.close()
####

print('A Closer Look at Overall Rating Versus Reactions Attributes')
print(' ')
print_corr('reactions', 'overall_rating', df_all_col, 'All Players')

# hue: attacking work rate
df11=df_all_col[df_all_col['attacking_work_rate'].isin(['low','medium','high'])]
corr11=df11[['reactions','overall_rating']].corr()
print(corr11)
vis11=sns.lmplot( x='reactions', y='overall_rating', hue='attacking_work_r
ate', sharex=False, data=df11, scatter=True, fit_reg=True, units=None, ord
```

```

er=1, legend=True)
plt.show()

# hue: defensive work rate
df12=df_all_col[df_all_col['defensive_work_rate'].isin(['low','medium','high'])]
corr12=df12[['reactions','overall_rating']].corr()
print(corr12)
vis12=sns.lmplot( x='reactions', y='overall_rating', hue='defensive_work_rate', sharex=False, data=df12, scatter=True, fit_reg=True, units=None, order=1, legend=True)
plt.show()

df_goalkeepers=df_all_col[df_all_col['gk_diving'] > 50] # goalkeepers
df_non_goalkeepers=df_all_col[df_all_col['gk_diving'] < 50] # non_goalkeepers

#### Subgroups

print ('#####')
print (' ')
print ('Non_Goalkeeper Subgroup')
print (' ')
#df_non_goalkeepers=df_all_col[df_all_col['gk_diving'] < 50] # non_goalkeepers
print_corr ('reactions','overall_rating', df_non_goalkeepers, 'Non_Goalkeepers: gk_diving < 50')

print ('#####')
print (' ')
print ('Goalkeeper Subgroup')
print (' ')
#df_goalkeepers=df_all_col[df_all_col['gk_diving'] > 50] # goalkeepers
# strongest correlation
print_corr ('gk_diving','overall_rating',df_goalkeepers, 'Goalkeepers Only : gk_diving > 50')
# 2nd strongest correlation
print_corr ('gk_reflexes','overall_rating',df_goalkeepers, 'Goalkeepers Only: gk_diving > 50')
# 3rd strongest correlation
print_corr ('gk_positioning','overall_rating',df_goalkeepers, 'Goalkeepers Only: gk_diving > 50')
# 4th strongest correlation
print_corr ('gk_handling','overall_rating',df_goalkeepers, 'Goalkeepers Only: gk_diving > 50')
# 5th strongest correlation
print_corr ('gk_kicking','overall_rating',df_goalkeepers, 'Goalkeepers Only: gk_diving > 50')
# 6th strongest correlation
print_corr ('reactions','overall_rating', df_goalkeepers, 'Goalkeepers Only: gk_diving > 50')

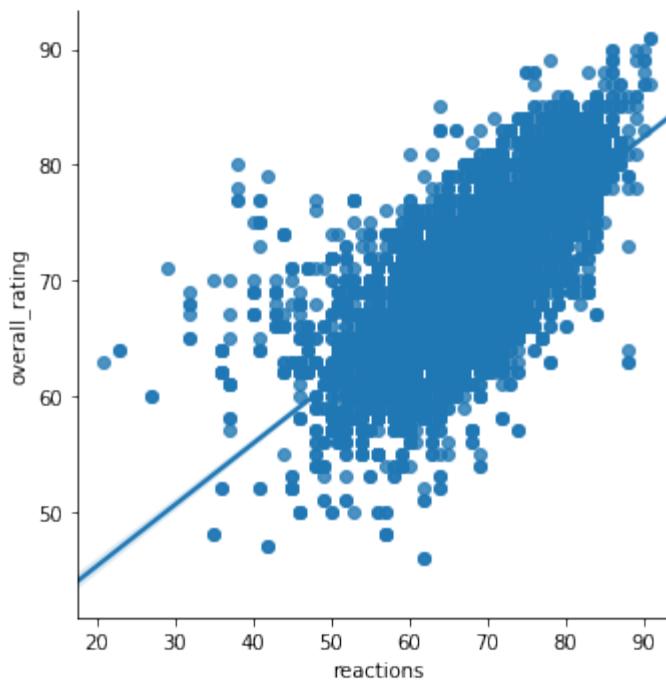
```

A Closer Look at Overall Rating Versus Reactions Attributes

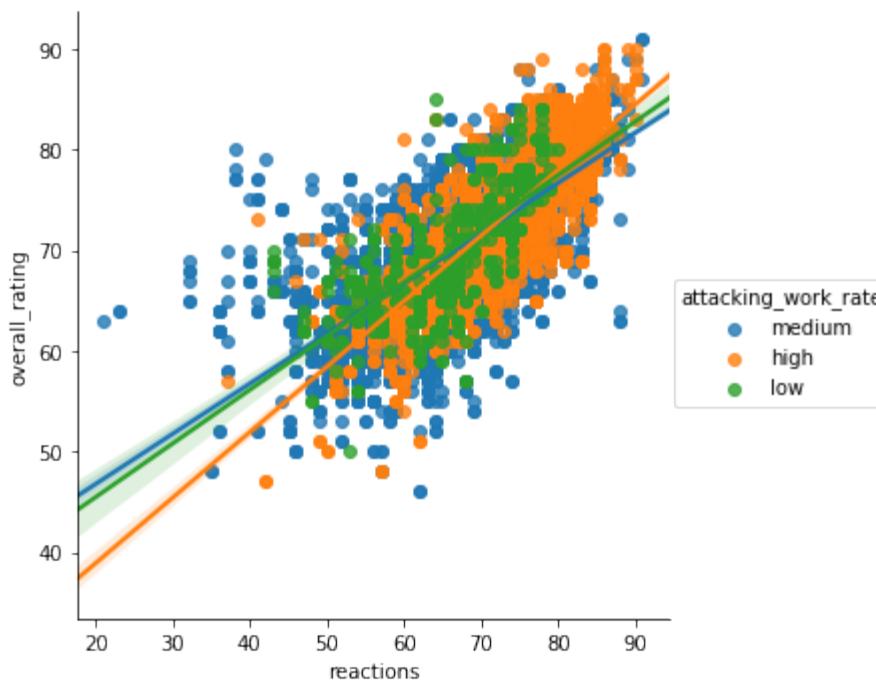
All Players

reactions overall_rating

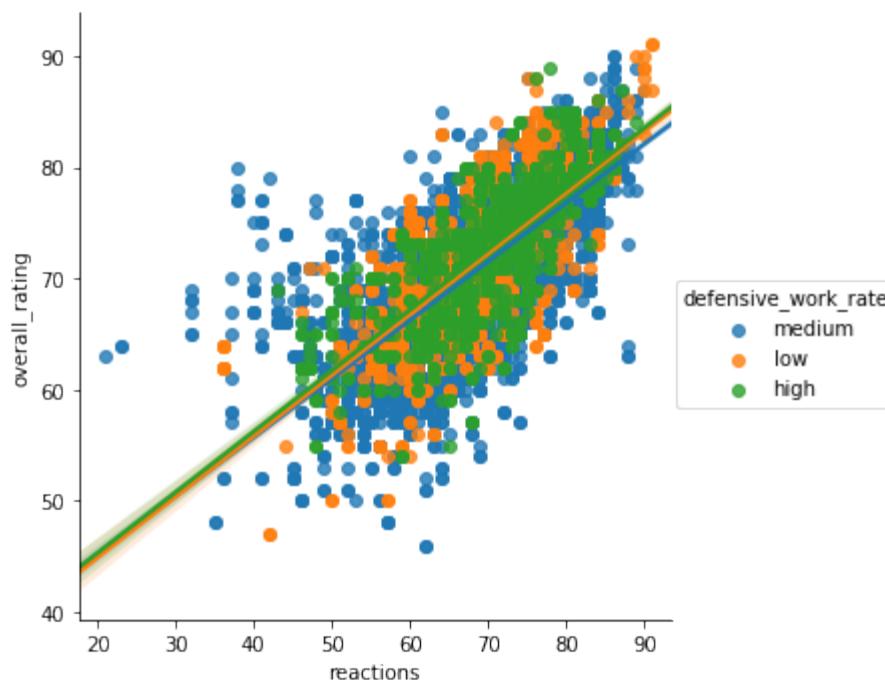
reactions	1.00000	overall_rating	0.72483
overall_rating	0.72483		1.00000



reactions	1.00000	overall_rating	0.72456
overall_rating	0.72456		1.00000



reactions	1.000000	overall_rating	0.725001
overall_rating	0.725001		1.000000

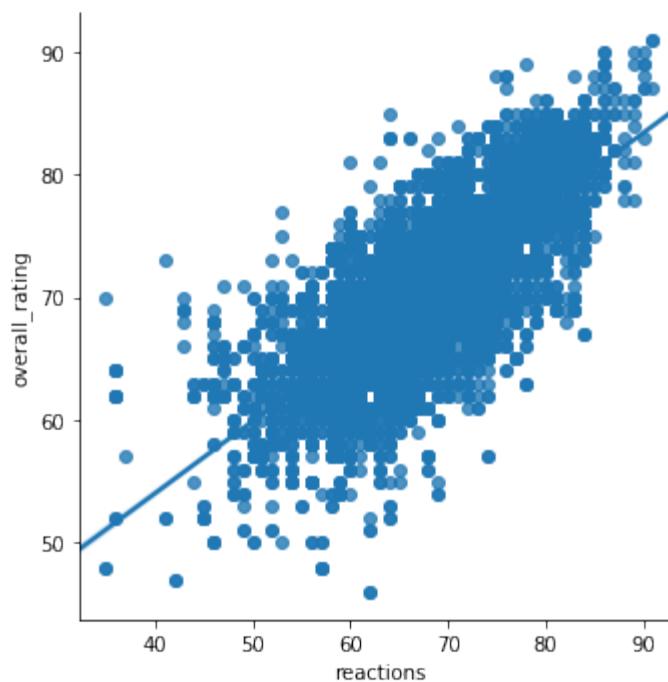


```
#####
```

Non_Goalkeeper Subgroup

Non_Goalkeepers: gk_diving < 50

	reactions	overall_rating
reactions	1.000000	0.759507
overall_rating	0.759507	1.000000



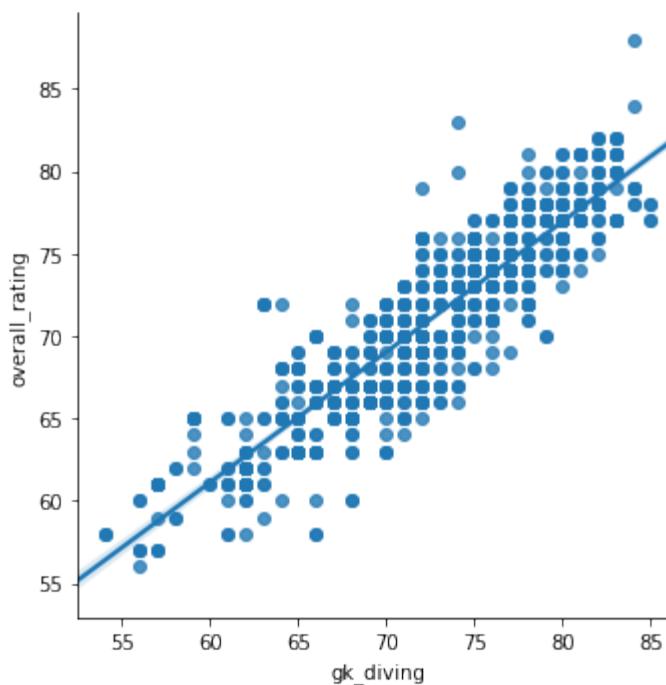
```
#####
```

Goalkeeper Subgroup

Goalkeepers Only: gk_diving > 50

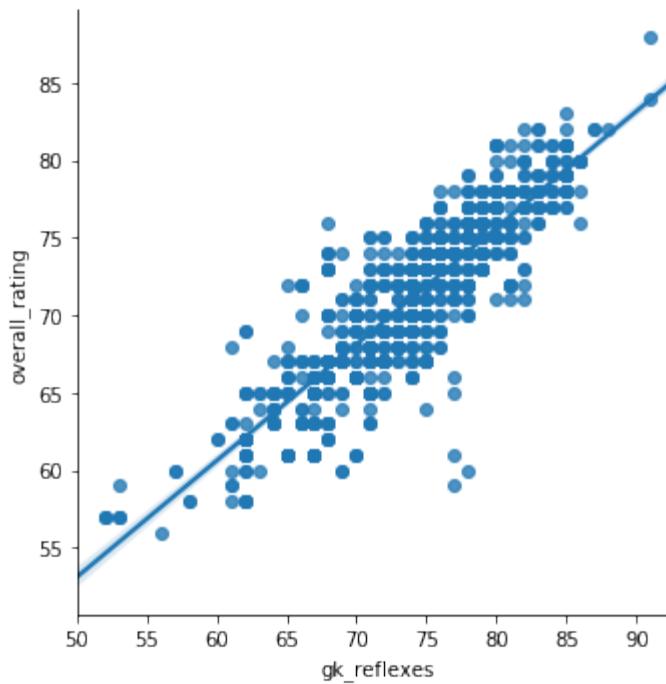
	gk_diving	overall_rating
gk_diving	1.000000	0.759507
overall_rating	0.759507	1.000000

gk_diving	1.000000	0.897919
overall_rating	0.897919	1.000000



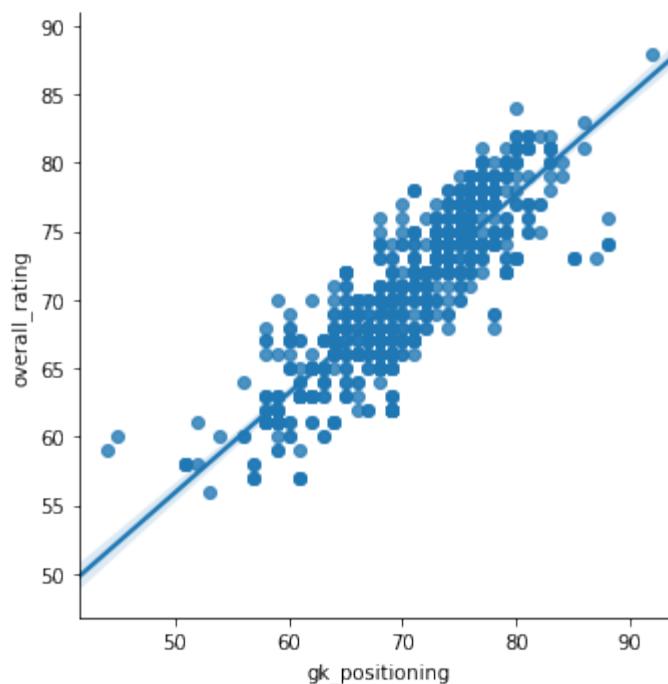
Goalkeepers Only: gk_diving > 50

gk_reflexes	1.000000	0.877756
overall_rating	0.877756	1.000000



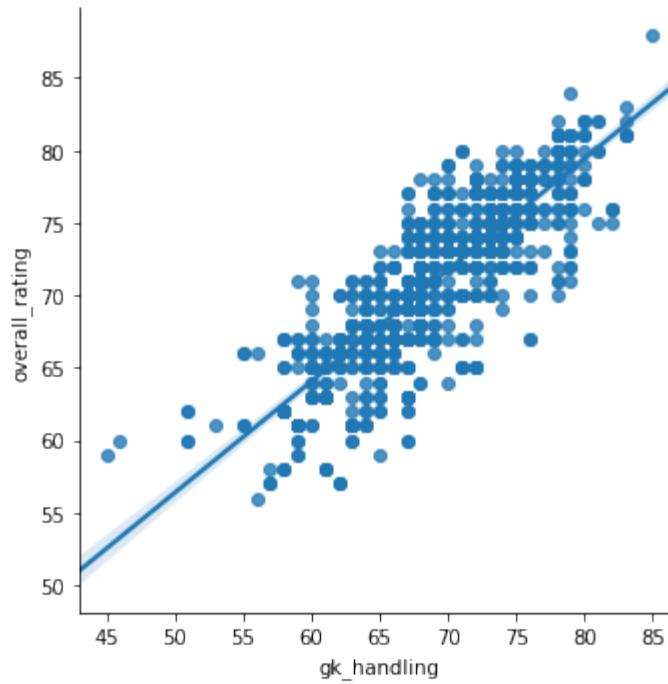
Goalkeepers Only: gk_diving > 50

gk_positioning	1.000000	0.863162
overall_rating	0.863162	1.000000



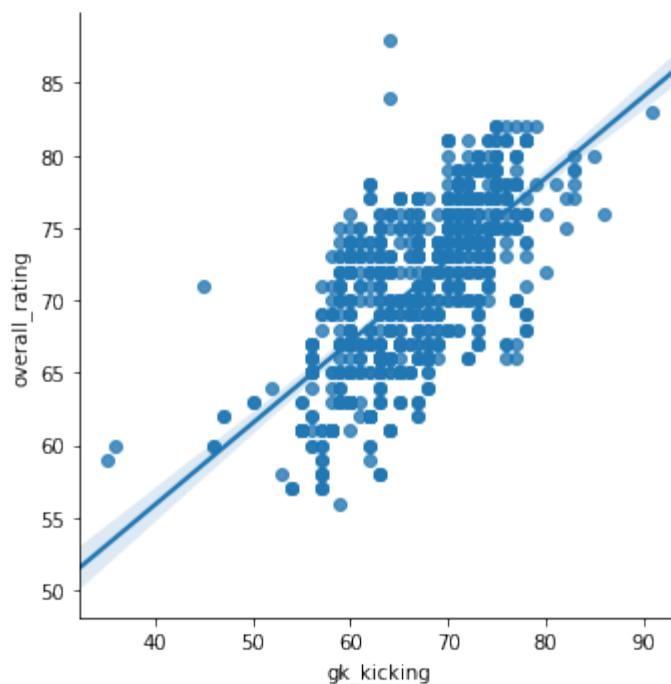
Goalkeepers Only: $\text{gk_diving} > 50$

	gk_handling	overall_rating
gk_handling	1.000000	0.828148
overall_rating	0.828148	1.000000



Goalkeepers Only: $\text{gk_diving} > 50$

	gk_kicking	overall_rating
gk_kicking	1.000000	0.669727
overall_rating	0.669727	1.000000



Goalkeepers Only: gk_diving > 50

	reactions	overall_rating
reactions	1.00000	0.56092
overall_rating	0.56092	1.00000

