week4

October 21, 2017

1 Exploratory Data Analysis

1.1 The act of making sense of data by converting raw data into actionable information

Myatt, Glenn J.; Johnson, Wayne P.. Making Sense of Data I: A Practical Guide to Exploratory Data Analysis and Data Mining. Wiley.

2 Steps In Exploratory Data Analysis

1. Problem definition and planning
2. Data preperation
3. Data analysis
4. Deployment

3 Problem Definition

- Identify the problem to be solved
  - Problem to explore? Question to answer? System to build?
- List project deliverables
  - Report vs System
- Identify required resources/skills and success factors
  - Including data sources
- Assemble team
- Prepare plan

4 Data Preperation

- Access and combine data
- Summarize data
- Look for errors
- Transform data
- Segment data

5 Data Analysis

- Exploring relationships between variables
- Group summaries and comparisons
- Visualization **(Our focus)**
- Other advanced topics include:
  - Discovering non-trivial patterns
  - Building regression and classification models
  - … etc

6 Deployment

- Generate report
- Deploy decision-support tool/system
- Measure business impact

7 Notes On The Steps

- They apply to any other advanced type of analysis
- Because the process involves discovery, it is iterative
  - Experience is key
  - Multiple perspective and critical thinking is useful
  - Patience and Perseverance is required

8 Skills we learned so far focus on

8.1 Data Preperation

- Loading and discovering data
- Plotting and describing variables
- Sorting and filteration
- Preliminary manipulation

8.2 Presenting data (Part of Deployment)

- Using Jupyter Notebook

9 Where are we headed?

Week	Step	Topic
4 - Current	Data Prep.	Joining & Aggregating data

Week	Step	Topic
5 & 6	Data Analysis	Visualizing Data - Groups & Time Series
7	Review	Example case - in class
7	**Midterm Project**	Problem handed out for individual analysis, due end of week 8
8	Data Analysis - Midterm	Advanced topics, Guest speakers
9 & 10	Data Prep.	Internet data collection (APIs and Scraping)
9	Final project	Announcement, team and problem selection
11	Final project - Phase 1	Present proposal - Problem definition
13	Final project - Phase 2	Data preparation report due
15	Final project - Phase 3	Data analysis report due
Final Exam	Final project - Phase 4	Result presentation

10 Joining Data

- Analysis is typically done a single dataframe
- Sometime the data might be in two different files/dataframes
- joining combines the data into a single dataframe

11 Concatination operation

- Easiest form of joining data
- Dataframes must have identical columns
- Rows from one dataframe are added to another
  - End result is a dataframe containing all the rows from combined dataframes

12 Join Operation

- Combines columns from two different dataframe into a single dataframe
- This is what we typically mean by joining data
- In pandas, you use
  - join() if you are joining on dataframe indecies
  - merge() if you are joining on columns

13 Things to consider when joining data

13.1 But first, let's learn how to connect to fetch data from databases

Dataset can be found at European Soccer Kaggle Dataset
by Hugo Mathien
You can download the sqlite db for this exercise from here

14 But first, working with Sqlite3 DBs

```
In [1]: # import libraries
        import pandas as pd
        import sqlite3

        # connect to database
        # database.sqlite is the name of the database
        db = sqlite3.connect("database.sqlite")
```

15 Fetching data from the database connection

This involves writing SQL
This page describes how the data looks like.
If you look to the left of the page, you will see the following tables: - Player - Player_Attributes
Let's load 500 entries from them

```
In [78]: # prepare the sql statement
         sql = "SELECT * FROM Player limit 5000"

         # execute it on the database
         player_df = pd.read_sql(sql, db)

         # let's view the data
         player_df.head()
```

```
Out[78]:    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
```

```
In [79]: # Now it is your turn to fetch 500 entries from Player_Attributes
         sql = "SELECT * FROM Player_Attributes limit 5000"
         atts_df = pd.read_sql(sql, db)
         atts_df.head()
```

```
Out[79]:    id  player_fifa_api_id  player_api_id                 date  overall_rat
         0   1              218353         505942  2016-02-18 00:00:00
         1   2              218353         505942  2015-11-19 00:00:00
         2   3              218353         505942  2015-09-21 00:00:00
         3   4              218353         505942  2015-03-20 00:00:00
         4   5              218353         505942  2007-02-22 00:00:00

            potential preferred_foot attacking_work_rate defensive_work_rate  cross
         0         71          right              medium              medium
         1         71          right              medium              medium
         2         66          right              medium              medium
         3         65          right              medium              medium
         4         65          right              medium              medium

                ...       vision  penalties  marking  standing_tackle  sliding_tackl
         0      ...           54         48       65               69             6
         1      ...           54         48       65               69             6
         2      ...           54         48       65               66             6
         3      ...           53         47       62               63             6
         4      ...           53         47       62               63             6

            gk_diving  gk_handling  gk_kicking  gk_positioning  gk_reflexes
         0          6           11          10               8            8
         1          6           11          10               8            8
         2          6           11          10               8            8
         3          5           10           9               7            7
         4          5           10           9               7            7

         [5 rows x 42 columns]
```

16 Things to consider when joining data

- Is there a key to combine data on? How will rows be matched to one another?

Examine the two data frames and suggest a key to use to combine

17 Things to consider when joining data

- What about entries that do not have a match in the other dataframe? do we include them?
  - Inner means to include in the result only the records that have matching entries in both dataframes
  - Outer means to include all entries, including ones without matching entries
    - The values for columns with non-matching entries will be NaN

```
In [35]: # Let's try inner join


         player_df.merge(atts_df,how="inner",on="player_api_id").head()
```

```
Out[35]:    id_x  player_api_id         player_name  player_fifa_api_id_x  \
         0     1         505942  Aaron Appindangoye                218353
         1     1         505942  Aaron Appindangoye                218353
         2     1         505942  Aaron Appindangoye                218353
         3     1         505942  Aaron Appindangoye                218353
         4     1         505942  Aaron Appindangoye                218353

                       birthday  height  weight  id_y  player_fifa_api_id_y  \
         0  1992-02-29 00:00:00  182.88     187     1                218353
         1  1992-02-29 00:00:00  182.88     187     2                218353
         2  1992-02-29 00:00:00  182.88     187     3                218353
         3  1992-02-29 00:00:00  182.88     187     4                218353
         4  1992-02-29 00:00:00  182.88     187     5                218353

                           date      ...       vision  penalties marking  \
         0  2016-02-18 00:00:00      ...           54         48      65
         1  2015-11-19 00:00:00      ...           54         48      65
         2  2015-09-21 00:00:00      ...           54         48      65
         3  2015-03-20 00:00:00      ...           53         47      62
         4  2007-02-22 00:00:00      ...           53         47      62

           standing_tackle sliding_tackle  gk_diving  gk_handling  gk_kicking  \
         0              69             69          6           11          10
         1              69             69          6           11          10
         2              66             69          6           11          10
         3              63             66          5           10           9
         4              63             66          5           10           9

            gk_positioning  gk_reflexes
         0               8            8
         1               8            8
         2               8            8
         3               7            7
         4               7            7
```

```
[5 rows x 48 columns]
```

```
In [37]: # Outer join will give us an idea of where the data went

         player_df.merge(atts_df,how="outer",on="player_api_id").head()
```

```
Out[37]:    id_x  player_api_id         player_name  player_fifa_api_id_x  \
         0     1         505942  Aaron Appindangoye                218353
         1     1         505942  Aaron Appindangoye                218353
         2     1         505942  Aaron Appindangoye                218353
         3     1         505942  Aaron Appindangoye                218353
         4     1         505942  Aaron Appindangoye                218353

                       birthday  height  weight  id_y  player_fifa_api_id_y  \
         0  1992-02-29 00:00:00  182.88     187     1                218353
         1  1992-02-29 00:00:00  182.88     187     2                218353
         2  1992-02-29 00:00:00  182.88     187     3                218353
         3  1992-02-29 00:00:00  182.88     187     4                218353
         4  1992-02-29 00:00:00  182.88     187     5                218353

                           date     ...        vision  penalties marking  \
         0  2016-02-18 00:00:00     ...            54         48      65
         1  2015-11-19 00:00:00     ...            54         48      65
         2  2015-09-21 00:00:00     ...            54         48      65
         3  2015-03-20 00:00:00     ...            53         47      62
         4  2007-02-22 00:00:00     ...            53         47      62

           standing_tackle sliding_tackle  gk_diving  gk_handling  gk_kicking  \
         0              69             69          6           11          10
         1              69             69          6           11          10
         2              66             69          6           11          10
         3              63             66          5           10           9
         4              63             66          5           10           9

            gk_positioning  gk_reflexes
         0               8            8
         1               8            8
         2               8            8
         3               7            7
         4               7            7

         [5 rows x 48 columns]
```

```
In [38]: # Try to compare both operations by check counts, and null values
         # can you find differences?
         # Can you explain why these differences exist?
```

18 Things to consider when joining data

- The level of analysis
  - Be careful in your analysis with this!
  - Make sure you have the correct variable
- Consider the dataframes for: Player, Team, Match, League, Country
  - What are the levels of analysis and their relationship to observations in other dataframes?
  - What happens when we combine them?

19 What is Level of Analysis?

- Consider the Euro Soccer data:
  - A league will have many teams
  - A team will have many players
- Analysis can be at:
  - League level, where you compare leagues
  - Team level, where you compare teams
  - Player level, where you compare players
  - This is what we mean by level of analysis (AKA Unit of Analysis)

20 Team Level Analysis

- Do you include the league attributes?
- Do you include the team attributes?
- Do you include the player attributes?

21 Load Data

Load players.csv and teams.csv into **player_df** and **team_df** respectively

```
In [80]: # Load players and teams data here
         player_df =
         team_df =
```

```
In [104]: player_df.head()
```

```
Out[104]:       id_x  player_api_id         player_name  player_fifa_api_id_x  \
          765     48         439366     Abdoulaye Toure                210450
          1050    68          37422  Abella Perez Damia                159580
          2118   129         160447          Adam Smith                190885
          4180   253          32547            Alan Gow                140307
          3081   182         168047      Adrian Stoian                192072
```

```
                  birthday  height  weight  id_y  player_fifa_api_id_y  \
765   1994-03-03 00:00:00  187.96     170   766                210450
1050  1982-04-15 00:00:00  187.96     174  1051                159580
2118  1991-04-29 00:00:00  180.34     179  2119                190885
4180  1982-10-09 00:00:00  182.88     154  4181                140307
3081  1991-02-11 00:00:00  177.80     146  3082                192072

                     date     ...       penalties  marking standing_tackl
765   2016-05-05 00:00:00     ...              39       57               6
1050  2016-01-28 00:00:00     ...              46       65               7
2118  2015-11-12 00:00:00     ...              48       71               7
4180  2015-02-27 00:00:00     ...              62       25               2
3081  2016-02-04 00:00:00     ...              65       28               3

     sliding_tackle gk_diving  gk_handling  gk_kicking  gk_positioning  \
765              58         8           15           9               9
1050             69        13            9          12              19
2118             68         8            9          15               9
4180             25         6            9          10              12
3081             32        10           11           7              11

      gk_reflexes  team_api_id
765             7         8674
1050           12         8674
2118            6         8674
4180            9         8674
3081            6         8674

[5 rows x 49 columns]
```

In [106]: `team_df.head()`

Out[106]:
```
      id_x  team_api_id  team_fifa_api_id_x           team_long_name  \
317  26548         8674                1915             FC Groningen
150  11822         4087              111271  Évian Thonon Gaillard FC
456  43053         9906                 240          Atlético Madrid
374  35284         9807                1889        CF Os Belenenses
355  27780        10218                1971                Excelsior

    team_short_name  id_y  team_fifa_api_id_y                date  \
317             GRO   427                1915  2010-02-22 00:00:00
150             ETG   411              111271  2011-02-22 00:00:00
456             AMA    95                 240  2010-02-22 00:00:00
374             BEL   156                1889  2010-02-22 00:00:00
355             EXC   416                1971  2011-02-22 00:00:00

     buildUpPlaySpeed buildUpPlaySpeedClass           ...             \
317                41              Balanced           ...
```

```
150                  35              Balanced            ...
456                  64              Balanced            ...
374                  30                  Slow            ...
355                  73                  Fast            ...

     chanceCreationShooting chanceCreationShootingClass  \
317                      69                        Lots
150                      65                      Normal
456                      70                        Lots
374                      60                      Normal
355                      52                      Normal

     chanceCreationPositioningClass defencePressure defencePressureClass
317                       Organised              30                 Deep
150                       Organised              45               Medium
456                       Free Form              70                 High
374                       Organised              30                 Deep
355                       Organised              25                 Deep

     defenceAggression defenceAggressionClass  defenceTeamWidth  \
317                 30                Contain                30
150                 55                  Press                65
456                 34                  Press                55
374                 30                Contain                30
355                 47                  Press                33

    defenceTeamWidthClass  defenceDefenderLineClass
317                Narrow                     Cover
150                Normal                     Cover
456                Normal              Offside Trap
374                Narrow              Offside Trap
355                Narrow                     Cover

[5 rows x 29 columns]
```

22 Team Level Analysis

- Do you include the league attributes?
  - Yes you can
- Do you include the team attributes?
  - Yes you can, this is the point of the analysis
- Do you include the player attributes?
  - No! **unless you aggregate!**

23 What is aggregation?

- Combining observations from the same level of analysis into a single observation at a higher level of analysis

24 Match Analysis Example

- **buildUpPlaySpeed** is a team attribute.
- However, **overall_rating** is a player attribute.
  - You cannot include a single player overall_rating in the analysis of a team, because the value describe a single player.
  - However, if you calculate the **average_overall_rating** for all players in that team, you get a value that we can use to describe a team, because a team consists of players.
  - Any operatino to combine the overall_rating for all the players in the team will work:
    - Count, Sum, Min, Max, Std, Var, Mean, Median . . . etc.
- You can include all match attributes in the analysis of matches
- You must aggregate player attribute to include it in match analysis

```
In [110]: team_df[["team_api_id","buildUpPlaySpeed"]].head()
```

```
Out[110]:      team_api_id  buildUpPlaySpeed
          317         8674                41
          150         4087                35
          456         9906                64
          374         9807                30
          355        10218                73
```

```
In [112]: player_df[["player_api_id","team_api_id","overall_rating"]].head()
```

```
Out[112]:       player_api_id  team_api_id  overall_rating
          765          439366         8674              64
          1050          37422         8674              71
          2118         160447         8674              70
          4180          32547         8674              63
          3081         168047         8674              70
```

```
In [113]: # First we have to merge based on team_api_id

          merged_df = player_df.merge(team_df, how="inner", on="team_api_id")
          merged_df.columns
```

```
Out[113]: Index(['id_x_x', 'player_api_id', 'player_name', 'player_fifa_api_id_x',
                 'birthday', 'height', 'weight', 'id_y_x', 'player_fifa_api_id_y',
                 'date_x', 'overall_rating', 'potential', 'preferred_foot',
                 'attacking_work_rate', 'defensive_work_rate', 'crossing', 'finishi
                 'heading_accuracy', 'short_passing', 'volleys', 'dribbling', 'curv
                 'free_kick_accuracy', 'long_passing', 'ball_control', 'acceleratic
```

```
       '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', 'team_api_id', 'id_
       'team_fifa_api_id_x', 'team_long_name', 'team_short_name', 'id_y_y
       'team_fifa_api_id_y', 'date_y', 'buildUpPlaySpeed',
       'buildUpPlaySpeedClass', 'buildUpPlayDribbling',
       'buildUpPlayDribblingClass', 'buildUpPlayPassing',
       'buildUpPlayPassingClass', 'buildUpPlayPositioningClass',
       'chanceCreationPassing', 'chanceCreationPassingClass',
       'chanceCreationCrossing', 'chanceCreationCrossingClass',
       'chanceCreationShooting', 'chanceCreationShootingClass',
       'chanceCreationPositioningClass', 'defencePressure',
       'defencePressureClass', 'defenceAggression', 'defenceAggressionCla
       'defenceTeamWidth', 'defenceTeamWidthClass',
       'defenceDefenderLineClass'],
      dtype='object')
```

```
In [120]: # notice that team entries are duplicated
          # and that we have an entry for every player
          merged_df[["player_name","team_long_name","overall_rating","buildUpPlaySp
```

```
Out[120]:          player_name team_long_name  overall_rating  buildUpPlaySpeed
          0     Abdoulaye Toure   FC Groningen              64                41
          1   Abella Perez Damia   FC Groningen              71                41
          2          Adam Smith   FC Groningen              70                41
          3            Alan Gow   FC Groningen              63                41
          4       Adrian Stoian   FC Groningen              70                41
```

```
In [122]: # To analyze teams, you must aggregate player observations if you want to
          # otherwise, you have to filter on team attributes and remove duplicates

          # let's aggregate overall_rating by calculating the average for the playe
          merged_df[
                      ["player_name","team_api_id","team_long_name","overall_rating
                   ].groupby("team_api_id").agg({"overall_rating":"mean"})
```

```
Out[122]:             overall_rating
          team_api_id
          4087             62.090909
          7788             67.909091
          7819             68.000000
          8262             67.818182
          8322             66.000000
          8342             67.090909
          8526             69.636364
          8535             69.000000
          8559             67.545455
```

```
8674              68.000000
9789              66.727273
9807              69.363636
9810              67.363636
9825              65.454545
9826              70.636364
9880              72.363636
9906              66.272727
9987              69.000000
10218             64.909091
208931            67.454545
```

In [125]:
```
# simply merge it to team_df to start analyzing teams
# but dont forget to reset_index to convert the index into a regular colu
ratings_df = merged_df[
            ["player_name","team_api_id","team_long_name","overall_rating
         ].groupby("team_api_id").agg({"overall_rating":"mean"}).reset_ir

team_df.merge(ratings_df, how="inner", on="team_api_id").head()
```

Out[125]:
```
    id_x  team_api_id  team_fifa_api_id_x            team_long_name  \
0  26548         8674                1915              FC Groningen
1  11822         4087              111271  Évian Thonon Gaillard FC
2  43053         9906                 240           Atlético Madrid
3  35284         9807                1889          CF Os Belenenses
4  27780        10218                1971                 Excelsior

  team_short_name  id_y  team_fifa_api_id_y                 date  \
0             GRO   427                1915  2010-02-22 00:00:00
1             ETG   411              111271  2011-02-22 00:00:00
2             AMA    95                 240  2010-02-22 00:00:00
3             BEL   156                1889  2010-02-22 00:00:00
4             EXC   416                1971  2011-02-22 00:00:00

   buildUpPlaySpeed buildUpPlaySpeedClass       ...         \
0                41              Balanced       ...
1                35              Balanced       ...
2                64              Balanced       ...
3                30                  Slow       ...
4                73                  Fast       ...

   chanceCreationShootingClass chanceCreationPositioningClass  \
0                         Lots                      Organised
1                       Normal                      Organised
2                         Lots                      Free Form
3                       Normal                      Organised
4                       Normal                      Organised
```

```
   defencePressure defencePressureClass defenceAggression  \
0               30                 Deep                30
1               45               Medium                55
2               70                 High                34
3               30                 Deep                30
4               25                 Deep                47

   defenceAggressionClass defenceTeamWidth  defenceTeamWidthClass  \
0                 Contain               30                 Narrow
1                   Press               65                 Normal
2                   Press               55                 Normal
3                 Contain               30                 Narrow
4                   Press               33                 Narrow

  defenceDefenderLineClass  overall_rating
0                    Cover       68.000000
1                    Cover       62.090909
2             Offside Trap       66.272727
3             Offside Trap       69.363636
4                    Cover       64.909091

[5 rows x 30 columns]
```

25 Player Analysis Example

- **overall_rating** is a player attribute
- **buildUpPlaySpeed** is a team attribute
  - While this is an attribute that describes a team, this is the team that the player is part of
  - The player is affected by the overall performance of the team, and describes the **team that the player is part of**, so in a way, it is a player attribute
  - You will notice that the value of buildUpPlaySpeed does not change for players in the same team
- You can include all player attribute to analyze and compare players
- You can also include team attributes without problems, because they can also be considered player attribute

```
In [127]: # you can perform your analysis directly on marged_df
          # because the level of analysis is the player there
          merged_df[["player_name","team_api_id","team_long_name","overall_rating",
```

```
Out[127]:            player_name  team_api_id team_long_name  overall_rating  \
          0     Abdoulaye Toure         8674    FC Groningen              64
          1  Abella Perez Damia         8674    FC Groningen              71
          2         Adam Smith         8674    FC Groningen              70
          3           Alan Gow         8674    FC Groningen              63
          4      Adrian Stoian         8674    FC Groningen              70
```

```
   buildUpPlaySpeed
0                 41
1                 41
2                 41
3                 41
4                 41
```

26 Aggregating With Transform

If you want to create a column in merged_df that contains the average overall_rating then you use **transform**

```
In [142]: import numpy as np
          merged_df.groupby("team_api_id").transform(np.mean).overall_rating.head(1

Out[142]: 0     68.000000
          1     68.000000
          2     68.000000
          3     68.000000
          4     68.000000
          5     68.000000
          6     68.000000
          7     68.000000
          8     68.000000
          9     68.000000
          10    68.000000
          11    62.090909
          12    62.090909
          13    62.090909
          14    62.090909
          Name: overall_rating, dtype: float64

In [141]: # simply assign this column to merged_df and give it an appropriate name

          merged_df["mean_overall_rating"] = merged_df.groupby("team_api_id").trans
          merged_df.head(15)

Out[141]:     id_x_x  player_api_id              player_name  player_fifa_api_id_x
          0       48         439366        Abdoulaye Toure                 21045(
          1       68          37422      Abella Perez Damia                 15958(
          2      129         160447             Adam Smith                 190885
          3      253          32547               Alan Gow                 140307
          4      182         168047         Adrian Stoian                 192072
          5      246          34268              Alain Nef                  49939
          6       65         302985            Abel Khaled                 207541
          7      206         213366         Afriyie Acquah                 201223
          8       73          80592       Aboubakar Oumarou                 218548
          9      275          37503  Albano Benjamin Bizzarri                  14907
```

```
10      37         173955    Abdoul Razzagui Camara               19395
11     247         182847       Alain Pierre Mendy               20935
12     162         121643          Adrian Chomiuk                18662
13     243         127255          Akwetey Mensah                19878
14      51         419681        Abdul Aziz Tetteh               19019

               birthday  height  weight  id_y_x  player_fifa_api_id_y  \
0   1994-03-03 00:00:00  187.96     170     766                210450
1   1982-04-15 00:00:00  187.96     174    1051                159580
2   1991-04-29 00:00:00  180.34     179    2119                190885
3   1982-10-09 00:00:00  182.88     154    4181                140307
4   1991-02-11 00:00:00  177.80     146    3082                192072
5   1982-02-06 00:00:00  190.50     194    4057                 49939
6   1992-11-09 00:00:00  180.34     148    1023                207541
7   1992-01-05 00:00:00  177.80     154    3487                201223
8   1987-04-01 00:00:00  182.88     168    1129                218548
9   1977-11-09 00:00:00  193.04     196    4532                 14907
10  1990-02-20 00:00:00  177.80     157     555                193953
11  1989-11-17 00:00:00  182.88     159    4088                209352
12  1988-06-23 00:00:00  182.88     179    2749                186629
13  1983-04-15 00:00:00  172.72     163    4025                198781
14  1989-02-10 00:00:00  182.88     190     803                190193

                 date_x         ...           chanceCreationShootingClas
0   2016-05-05 00:00:00         ...                                  Lot
1   2016-01-28 00:00:00         ...                                  Lot
2   2015-11-12 00:00:00         ...                                  Lot
3   2015-02-27 00:00:00         ...                                  Lot
4   2016-02-04 00:00:00         ...                                  Lot
5   2016-03-10 00:00:00         ...                                  Lot
6   2015-03-13 00:00:00         ...                                  Lot
7   2016-05-12 00:00:00         ...                                  Lot
8   2015-04-01 00:00:00         ...                                  Lot
9   2015-11-26 00:00:00         ...                                  Lot
10  2016-04-21 00:00:00         ...                                  Lot
11  2013-03-15 00:00:00         ...                                Norma
12  2010-08-30 00:00:00         ...                                Norma
13  2010-08-30 00:00:00         ...                                Norma
14  2016-05-05 00:00:00         ...                                Norma

    chanceCreationPositioningClass defencePressure defencePressureClass
0                        Organised              30                 Deep
1                        Organised              30                 Deep
2                        Organised              30                 Deep
3                        Organised              30                 Deep
4                        Organised              30                 Deep
5                        Organised              30                 Deep
6                        Organised              30                 Deep
```

```
7                        Organised             30                 Deep
8                        Organised             30                 Deep
9                        Organised             30                 Deep
10                       Organised             30                 Deep
11                       Organised             45               Medium
12                       Organised             45               Medium
13                       Organised             45               Medium
14                       Organised             45               Medium

   defenceAggression  defenceAggressionClass  defenceTeamWidth  \
0                 30                 Contain                30   
1                 30                 Contain                30   
2                 30                 Contain                30   
3                 30                 Contain                30   
4                 30                 Contain                30   
5                 30                 Contain                30   
6                 30                 Contain                30   
7                 30                 Contain                30   
8                 30                 Contain                30   
9                 30                 Contain                30   
10                30                 Contain                30   
11                55                   Press                65   
12                55                   Press                65   
13                55                   Press                65   
14                55                   Press                65   

    defenceTeamWidthClass  defenceDefenderLineClass  mean_overall_rating  
0                  Narrow                     Cover            68.000000  
1                  Narrow                     Cover            68.000000  
2                  Narrow                     Cover            68.000000  
3                  Narrow                     Cover            68.000000  
4                  Narrow                     Cover            68.000000  
5                  Narrow                     Cover            68.000000  
6                  Narrow                     Cover            68.000000  
7                  Narrow                     Cover            68.000000  
8                  Narrow                     Cover            68.000000  
9                  Narrow                     Cover            68.000000  
10                 Narrow                     Cover            68.000000  
11                 Normal                     Cover            62.090909  
12                 Normal                     Cover            62.090909  
13                 Normal                     Cover            62.090909  
14                 Normal                     Cover            62.090909  

[15 rows x 78 columns]
```

27 Performing Analysis

- We combine data into single dataframe
- With varying levels of analysis, we have varying degrees of variability because of duplication
  - Remember how all players in the same team share the same value of the team attribute buildUpPlaySpeed
- When we combine data into a single dataframe we have **non-normal form** data with lots of duplicated values
  - Normal form is a database term, not stats
  - Data stored in a data is usually in normal form
  - While some values might be duplicated, the records are not

28 Summary

- Level of analysis is important
- You can include variables from higher levels of analysis without issues
  - Be aware that you might not have variability
- Including variables from lower levels of analysis requires aggregation
  - You aggregate in many different ways: Sums, Counts, Min, Max, Mean, Median, Mode ..etc
  - Aggregation is to produce a single scalar value from a group of values
- Represent many observations at a lower level into a single value at a higher level

29 Final Note About Groupby

- You don't have to have different levels of analysis to use groupby and aggregations
- You can use agg() and transform() with group by to analyze subgroups
  - Just group by the variable you want to create subgroups from
  - groupby should be given a categorical or discrete variable
  - subgroups can be created from a combination of variables

```
In [38]: # At the team level of analysis
         # create 4 new columns from player sprint_speed data:
         # mean_sprint_speed, max_sprint_speed, min_sprint_speed, and std_sprint_sp
```

```
In [147]: # at the player unit of analysis
          # create 4 new columns from player sprint_speed data:
          # mean_sprint_speed, max_sprint_speed, min_sprint_speed, and std_sprint_s
```