Mohammad Rasool (mrasool@lsu.edu)
Parimal Kashireddy (pkashi1@lsu.edu)
April 29, 2023

Premier League Players & Club Performance Analysis

Table Of Contents

1.1 Introduction 👋

The latest FIFA World Cup hosted in Qatar gained coverage from billions of people, and it has led to an exponential boom in FIFA's popularity. Due to this boost in coverage from around the world, there is a need for sophisticated data analysis techniques to extract useful and meaningful insights from the incredibly huge amount of data generated by the sport. The use of data analytics in soccer has become increasingly prevalent in recent years, with clubs and national teams investing heavily in data analysis to gain a competitive edge in tournaments and matches. Data analysis is used to identify trends, patterns, and insights that can help teams make informed decisions about player recruitment, team strategy, and performance optimization.

1.2 Motivation

We are longtime fans of this global sport. We have enjoyed watching and playing it on various occasions through our childhood and are still very invested in it. Our interests range from the national level all the way down to individual clubs and players. This fascination leads us to some curiosity and inspiration to try and understand the patterns of the sport. Was it possible to predict the outcome of individual games? Was it possible to aggregate attributes of individual players and use them to foretell the results of matchups? These questions are the reason we decided to delve into this project and we have made some discoveries that may hold the answers to those questions.

1.3 Project Description

The main objective to complete was making predictions from data that we were given and some data we were able to extract manually. Our project aims to leverage big data analytics and machine learning algorithms to predict the outcomes of soccer games. The project will involve collecting and processing large amounts of data, including club historical data and most

importantly player performance data from a video game (FIFA 23) that is very precise with its analysis.This data will then be used to train and test a few machine learning algorithms such as poisson distribution and linear regression models.

The ultimate goal of our project is to try and develop a predictive machine learning model that can accurately predict the outcomes of football games with relatively high precision. This model can be used by coaches, players, and fans alike to make informed decisions ,backed by real world data, about team strategies, player selection, and betting odds.

1.4 Division Of Roles

We split the roles of this project evenly. We both helped each other research and debug code and give new ideas for further improvement.

- Mohammad: Preprocessing, Cleaning, Storing, Transferring Data, Installing PySpark on Personal Laptops, Researching ML Algorithms, Converting PANDAS Algorithm to PySpark, Final Report
- Parimal: Researching ML Algorithms, Writing Pandas ML Algorithm , Data Visualization, Documentation, Final Report

2.1 Big Data Frameworks ⌨️

The Big Data frameworks used for the entire duration and successful completion of this project are:

- Hadoop MapReduce
- Hadoop HDFS
- Apache Spark (PySpark)

Some notable softwares and libraries that aided in data collection and processing are:

- Jupyter Notebook
- Koalas API
- PySpark SQL and ML libraries
- Kaggle & Official Premier League Website

Hadoop MapReduce:

Hadoop MapReduce is a framework designed for analyzing and processing large data sets in a distributed manner. It achieves this by breaking data into smaller parts, processing them in parallel on a cluster of computers, and merging the results. The framework is very efficient, scalable, and fault-tolerant, which makes it a perfect choice for our needs.

Hadoop HDFS:

Hadoop Distributed File System (HDFS) is an essential part of the Hadoop ecosystem we used in our project. Datasets are stored in HDFS in the form of comma-separated files, with each row starting with a newline character. Here are some key points that we learned in class:

- HDFS also uses jar files to compile the code directly through terminal commands.
- HDFS is a file system that is closely attached to the Hadoop MapReduce framework.
- HDFS is used to store data on a cluster or any configured environment.

Apache Spark (PySpark):

Apache Spark is a framework that lets you process lots of data. It's faster than other tools like Hadoop MapReduce, and it can handle different kinds of tasks like machine learning or working with graphs.

PySpark is the Python API used to utilize Apache Spark with the Python programming language. Python is a popular language for data science, and PySpark makes it easy to work with big datasets. It's also compatible with other Python tools like Pandas and NumPy, which is great for us to make machine learning algorithm creation easier.

Jupyter Notebook: Our course VM was not able to support our dataset, so we manually installed PySpark onto our laptops. This installation process was the hardest aspect of our project.

Koalas API: Our Ml algorithms were originally written in Pandas which was not completely compatible with PySpark, so to make it compatible we used Koalas API dataframes to be able to use PySpark and Pandas simultaneously.

PySpark SQL & ML Libraries: Multiple PySpark libraries used to train and implement prediction analysis and data processing.

2.2 Datasets 💾

We used 2 primary datasets for our analysis:

- FIFA 23 Complete Player Dataset : https://www.kaggle.com/datasets/stefanoleone992/fifa-23-complete-player-dataset
    - This dataset has 110 attributes for over 20k players, making it a very informational data package.
    - This data contains multiple file:
        - **female_coaches.csv**
        - **female_players (legacy).csv**
        - **female_players.csv**
        - **female_teams.csv**
        - **male_coaches.csv**
        - **male_players (legacy).csv**
        - **male_players.csv**
        - **male_teams.csv**
    - We used the largest dataset out of these files which was the male_players.csv = ~ **5.24GB** in size. This CSV file has 110 columns, with each column being an attribute of a player, and over 20k rows, with each being a separate player.

- Previous Game Data from the Official Premier League Website
    - We collected this data manually and extracted it into a CSV file so we could use it to predict the future using historical outcomes.
    - For good measure, we decided to search for losses, wins, and draws for each of the Premier League teams to increase the accuracy of the prediction algorithms.

3.1 Detailed Description of Components

We had many complex components/steps in our projects timeline:

- **Preprocessing Data:**
    - The VM used in VMware was limited in resources such as processing speed and storage space. Hadoop is very complicated to install on our own laptops. Due to these restrictions, we made a decision to split our 5.24 GB dataset into smaller chunks, specifically **200 MB** files. We tried to do this in Hadoop but since we could not even transfer the bigger dataset over to HDFS for processing, we wrote a python program that would do this task for us. The python program was then run on our local machines. After that we would write and run MapReduce jobs on each individual 200MB file.
        - The python program written was: **splitFiles200.py**
- **Storing Data:**
    - **$ hadoop fs -put players_X.csv /user/training/data** ***(****Here the X variable is a number 1-27 indicating 1/27 part of the splitted dataset****)***
- **Cleaning Data:**
    - To clean the dataset we used Hadoop MapReduce programs as described in the course.
        - Delete null values
            - NullReducer, NullMapper, NullDriver

        - **hadoop jar null.jar null.NullDriver players_X.csv outputNULL**
    - Deleting unnecessary columns
        - CsvReducer, CsvMapper, CsvDriver
        - **hadoop jar deleteColumns.jar columns.CsvDriver players_X.csv outputColumns**

```
String[] columnsToDrop = {"player_url", "fifa_version", "fifa_update", "fifa_update_date",
    "dob", "club_loaned_from", "club_joined_date", "club_contract_valid_until_year",
    "real_face", "release_clause_eur", "player_face_url"};
```

- **Extract Meaningful Data:**
    - Get Attributes for ALL Premier League Players
        - CsvFilterReducer, CsvFilterMapper, CsvFilterDriver
        - **hadoop jar AllPLPlayers.jar columns.CsvFilterDriver players_X.csv output**
        - Output File: **Extracted.csv** (Contains ALL Premier League players & their attributes)
    - Get Top 11 Players from each Premier League Team
        - PlayerFilterReducer, PlayerFilterMapper, PlayerFilterDriver
        - **hadoop jar top11.jar top11.PlayerFilterDriver players_X.csv outputTOP11**
        - Output File: **player_attributes.csv** (Contains only the top 11 players of each club)
- **Write Machine Learning Algorithms (Hardest Step)** 😢
    - We are both experienced in the use of Python and the Pandas library but these tools did not count as contributions towards our project. To begin with we downloaded **Jupyter Notebook** using **Anaconda.** These tools were crucial in our development of the ML algorithms. We downloaded the

correct versions of Java and Python that were needed to run our ML Algorithms.
    - We used two models: LinearRegression and Poisson Distribution.
    - After this we coded the algorithms in **PANDAS** using straight **PYTHON** code.
    - **We were later informed by Professor Kisung Lee that this was not sufficient for the project.**We had to download and install PySpark onto our personal laptops in order to convert the Pandas code into PySpark code. We achieved this by following this tutorial on YouTube: **https://www.youtube.com/watch?v=Rv4NOAn2m-g**
    - We were tasked with the most difficult section of this project converting our already written algorithms into PySpark code using the Koalas API. This unfortunately also did not work so we turned this algorithm directly into PySpark over the span of a few days.
- **1st Algorithm (Converted from PANDAS to PySpark)**
    - This algorithm uses historical data we obtained from the Premier League Website to predict future games and then using the poisson distribution algorithm. The algorithm reads in **historical data from a CSV file** and creates a DataFrame called df_history_data. It then creates two new DataFrames, df_home and df_away, from df_history_data and renames their columns. A new DataFrame called df_team_strength is created by aggregating the df_home and df_away DataFrames for each team. A function called predict_points is defined to calculate the expected number of points for two teams using the Poisson distribution. The function takes in two arguments, home and away, and returns the predicted points for each team. The user is prompted to enter two team names and the predict_points

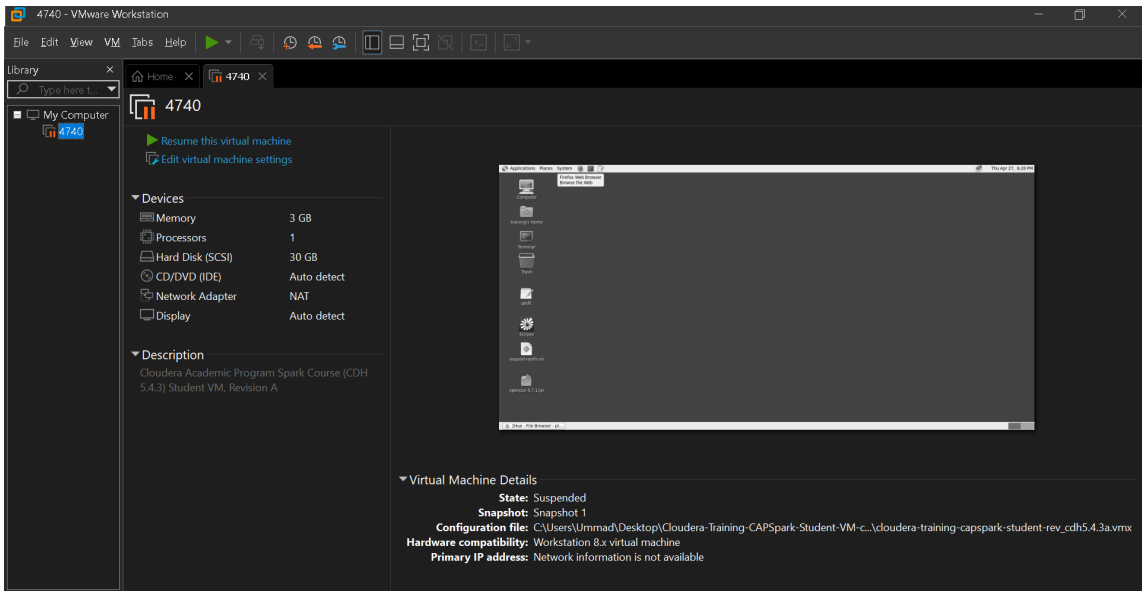
function is called with these names. The predicted winner or draw is printed to the console.
    - This algorithm is demoed in my video: **https://youtu.be/YgqQAc-qz9Y**
- **2nd Algorithm (Written In Pandas, Too Complicated for PySpark conversion)**
    - The **first part** of this algorithm was to read data from a CSV file, filter the data to find the top 11 players based on their attributes, group these players by their club name, and print the names of these players for each club that has at least 11 top players. This is similar to the function of the **Get Top 11 Players from each Premier League Team** MapReduce job mentioned above.
    - The **second part** is the Linear Regression model that predicts the winners and scores of potential games between football clubs using logistic regression. It loads data from a CSV file, filters it by user input, groups it by club, and normalizes it. It then creates a new DataFrame with each row representing a game between two clubs in the filtered list, trains a logistic regression model on this data, and predicts the winners and scores of potential games using the trained model. In case of faulty data, it chooses the team with higher rating as the winner and predicts the score accordingly.
        - This algorithm is demoed in my video: **https://youtu.be/waoNPUdPk1w**

4.1 Software Manual 📖

The first thing to do is to download VmWare Workstation Pro by requesting an allotment from LSU ITservices. After downloading Workstation Pro, it is necessary to download the CourseVM accessible at this link: CourseVMLink. After that step, it is required that you download the

dataset from Kaggle available at this link: Dataset. Next, setting up Hadoop MapReduce, HDFS, Hue, and PySpark is required to be able to properly run the programs and Jar files. **These tutorials are available on Prof. Kisung Lee's moodle page for CSC 4740**. After these are all set up, you must separate the data using the splitFiles200.py program and upload all of it to HDFS using the command provided in the detailed component section above. Next, you must execute each of the Hadoop MapReduce files in order as we did using all of the commands given. After this, you will have an exponentially smaller dataset to work with and can be transferred easily. Download Jupyter Notebook and PySpark following the tutorial at this link: **Tutorial**. Then you may copy our ML Algorithms and correct all of the file paths to ensure that the correct files are being accessed. After this you will be set up to test and run our ML Algorithms and predict football matches for yourself.

Homepage Of VMWare Workstation Pro:



Example of Running One Of The Hadoop MapReduce Jobs (Command may be different):

```
                Map output bytes=227114875
                Map output materialized bytes=229302581
                Input split bytes=228
                Combine input records=0
                Combine output records=0
                Reduce input groups=729231
                Reduce shuffle bytes=229302581
                Reduce input records=729231
                Reduce output records=729231
                Spilled Records=2187693
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=987
                CPU time spent (ms)=40510
                Physical memory (bytes) snapshot=640679936
                Virtual memory (bytes) snapshot=2734354432
                Total committed heap usage (bytes)=437125120
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=209719646
        File Output Format Counters
                Bytes Written=227457703
[training@localhost ~]$ hadoop jar deleteNull.jar deleteNull.NullDriver players_1.csv playerNull
```

The below image show the MapReduce Job running which will delete unnecessary columns from the dataset:

```
                Failed reduce tasks=4
                Launched map tasks=2
                Launched reduce tasks=4
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=0
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=49264
                Total time spent by all reduce tasks (ms)=68491
                Total vcore-seconds taken by all map tasks=49264
                Total vcore-seconds taken by all reduce tasks=68491
                Total megabyte-seconds taken by all map tasks=12611584
                Total megabyte-seconds taken by all reduce tasks=35067392
        Map-Reduce Framework
                Map input records=729231
                Map output records=729231
                Map output bytes=220654016
                Map output materialized bytes=222841722
                Input split bytes=228
                Combine input records=0
                Spilled Records=1196116
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=1155
                CPU time spent (ms)=27010
                Physical memory (bytes) snapshot=442515456
                Virtual memory (bytes) snapshot=1682960384
                Total committed heap usage (bytes)=298319872
        File Input Format Counters
                Bytes Read=209719646
[training@localhost ~]$ hadoop jar deleteColumns.jar deleteColumns.CsvDriver players_1.csv playerDeleteColumn
```

4.2 Evaluation/Test Environment 🧪

Our Hadoop MapReduce jobs were run directly on the course VM and from there they were transferred to our personal laptops. Machine Learning algorithms and PySpark were installed and executed on our own laptops.

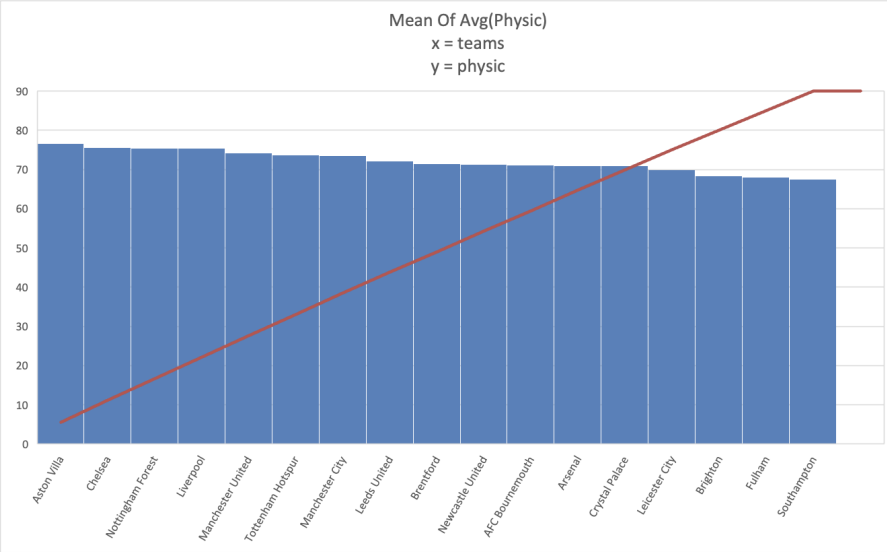
4.3 Evaluation/Test Results

After hosting a tournament where we set up a custom bracket/custom matchup of teams we liked. We found that this tournament was very similar to the results of a previous Premier League season. This was a great confidence booster as we were able to compile that the accuracy of the algorithms was high. Also we noticed that depending on the amount of games played the results of each game could differ because the difference in overall rating of any 2 teams could be very miniscule that the amount of games played could flip the odds of a game into the favor of another team, this was an interesting find.

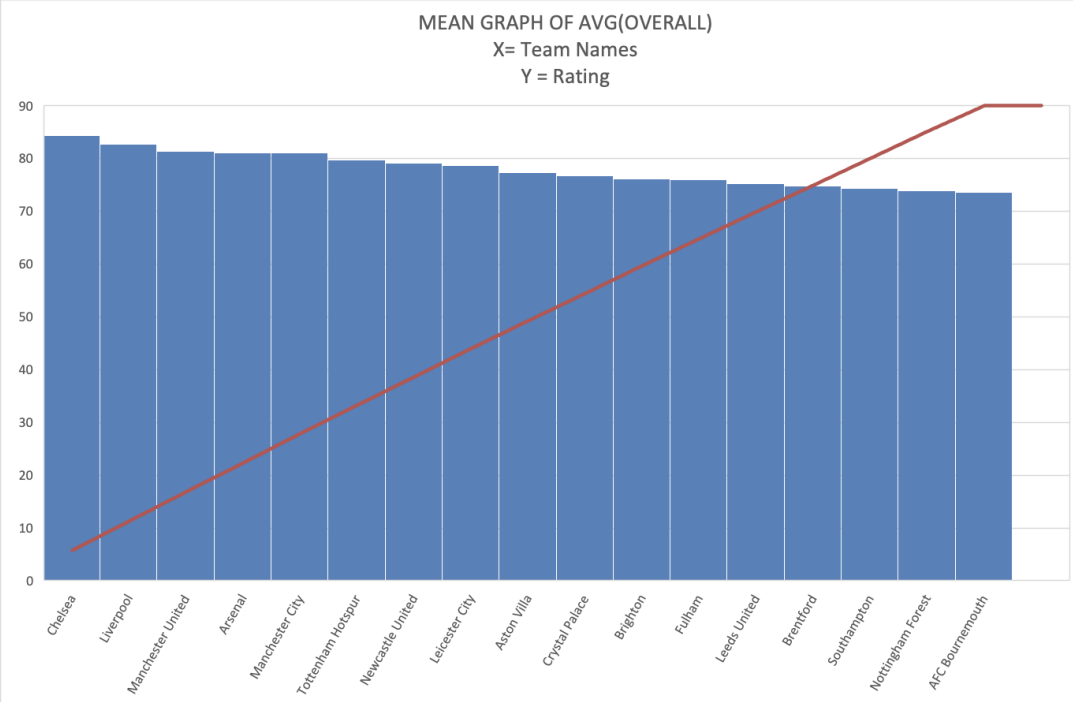
An example of each of the predictions algorithms running can be found at the following YouTube links:

- **Predict By Team History:** https://youtu.be/YgqQAc-qz9Y
- **Predict By Players Attributes:** https://youtu.be/waoNPUdPk1w

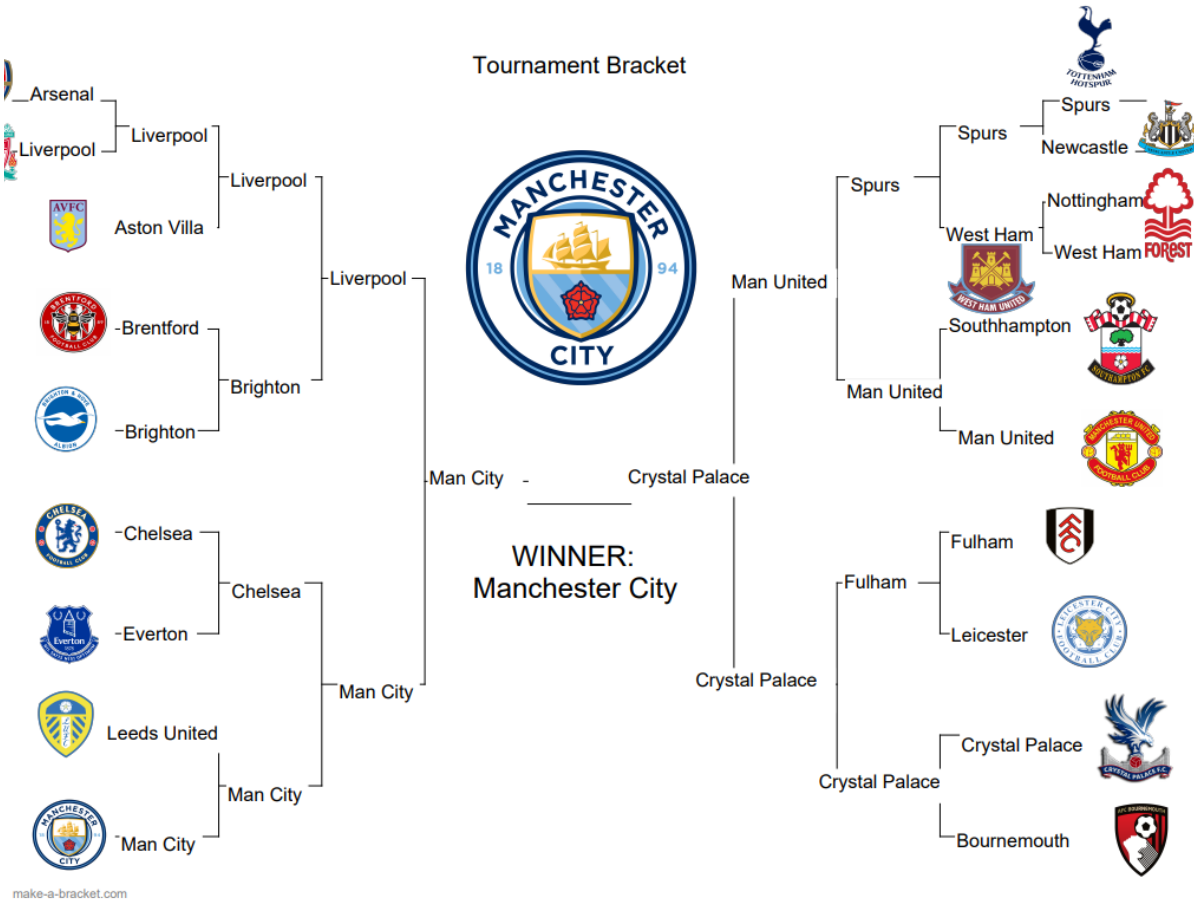
The graph below shows a graph of the Mean Physic of each of the Premier League Teams:



The graph below shows a graph of the Mean Overall Ratings of each of the Premier League Teams:



This is a simulated tournament using our **predict by team history** algorithm:



5.1 Conclusion 👋

File	# Of Lines
NullDriver.java	27
NullMapper.java	28
NullReducer.java	17
CsvMapper.java	45
CsvReducer.java	20
CsvDriver.java	35
CsvFilterMapper.java	29
CsvFilterReducer.java	25
CsvFilterDriver.java	30
splitFiles200.py	43
PlayerFilterMapper.java	90
PlayerFilterReducer.java	24
PlayerFilterDriver.java	25
PredictByPlayers.ipynb	92
PredictByTeamHistoryPYSPARK.ipynb	80
PredictByTeamsHistoryPANDAS.ipynb	54
Predict By Team History PYSPARK Operations	**55**