Big Data 18CS322

Class Project

FPL: Real time analysis of streamed EPL Data

Fantasy Premier League

Assignment Objectives and Outcomes

a. The objective of this project is for the students to use streaming spark and to become familiar with the real time processing and Spark Mllib.

b. At the end of this assignment, the student will be able to write and debug code for Spark and use Spark Mllib.

Ethical practices

Please submit original code only. You can discuss your approach with your friends but you must write original code. All solutions must be submitted through the portal. We will perform a plagiarism check on the code.

Overview

The aim of this project is to analyse events occurring during football matches of the English Premier League (EPL). A system has to be developed that will process the streamed events data and rate players. Chemistry coefficients between each pair of players has to be calculated as well, these signify how well a pair of players can interact with each other on the field. The system must also maintain profiles of each player and details of each match. It has to be developed on Spark and PySpark library can be used but all code must run on Spark, no native python code will be accepted. Streaming Spark is supposed to be used for accepting the streamed events data and Spark MLlib can be used for clustering and regression.

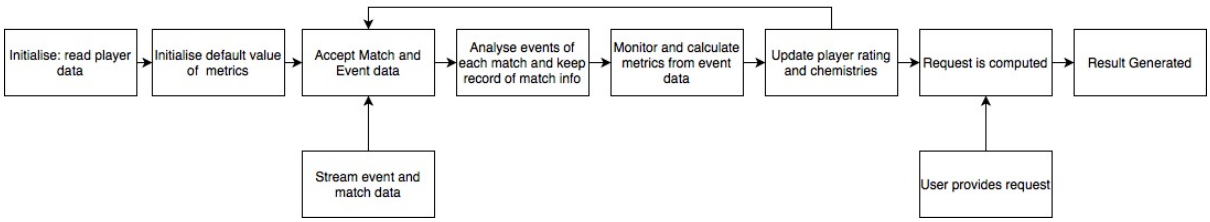

Figure 1: Flow of the System

The Dataset:

The players.csv and teams.csv file should be first read to initialise the player and team data. Then the provided executable can be run to stream data to port 6100, the system should read data from 6100 using streaming Spark. The data streamed is in JSON format and for every match first a match JSON object is sent followed by event JSON objects.

The two JSON objects have the following data:

Match JSON

Field	Description
competitionId	the identifier of the competition to which the match belongs to. It is a integer and refers to the field "wyId" of the *competition* document;
date and **dateutc**	the former specifies date and time when the match starts in explicit format (e.g., May 20, 2018 at 8:45:00 PM GMT+2), the latter contains the same information but in the compact format YYYY-MM-DD hh:mm:ss;
duration	the duration of the match. It can be "Regular", "ExtraTime", "Penalties"
gameweek	the week of the league, starting from the beginning of the league;
roundID	indicates the match-day of the competition to which the match belongs to.
seasonId	indicates the season of the match;
status	it can be "Played", "Cancelled", "Postponed" or "Suspended";
venue	the stadium where the match was held (e.g., "Stadio Olimpico");
winner	the identifier of the team which won the game, or 0 if the match ended with a draw;
wyId	the identifier of the match
teamsData	it contains several subfields describing information about each team that is playing that match: such as lineup, bench composition, list of substitutions, coach and scores:

Sub-field	description
hasFormation	it has value 0 if no formation (lineups and benches) is present, and 1 otherwise;
score	the number of goals scored by the team during the match (not counting penalties);
scoreET	the number of goals scored by the team during the match, including the extra time (not counting penalties);
scoreHT	the number of goals scored by the team during the first half of the match;
scoreP	the total number of goals scored by the team after the penalties;
side	the team side in the match (it can be "home" or "away");
teamId	the identifier of the team;
coachId	the identifier of the team's coach;
bench	the list of the team's players that started the match in the bench and some basic statistics about their performance during the match (goals, own goals, cards);
lineup	the list of the team's players in the starting lineup and some basic statistics about their performance during the match (goals, own goals, cards);
substitutions	the list of team's substitutions during the match, describing the players involved and the minute of the substitution.

Events JSON

Field	Description
eventId	the identifier of the event's type. Each eventId is associated with an event name (see next point);
eventName	name of the event's type. There are seven types of events: pass, foul, shot, duel, free kick, offside and touch;
subEventId	the identifier of the subevent's type. Each subEventId is associated with a subevent name (see next point);
subEventName	the name of the subevent's type. Each event type is associated with a different set of subevent types;
tags	a list of event tags, each one describes additional information about the event (e.g., accurate). Each event type is associated with a different set of tags;
eventSec	the time when the event occurs (in seconds since the beginning of the current half of the match);
id	a unique identifier of the event;
matchId	the identifier of the match the event refers to
matchPeriod	the period of the match. It can be "1H" (first half of the match), "2H" (second half of the match), "E1" (first extra time), "E2" (second extra time) or "P" (penalties time);
playerId	the identifier of the player who generated the event.
positions	the origin and destination positions associated with the event.
teamId	the identifier of the player's team.

Software/Languages to be used:

- Use only Spark to do this assignment
- PySpark library can be used
- Use Spark Mllib for clustering and regression
- No native python code will be allowed, all code must be compatible with PySpark and executed on Spark
- Use Streaming spark to accept the streamed match and event data

For running the streaming service, python3 and the following libraries will be required:

- socket
- sys

- requests
- json
- time
- csv
- cryptography

Marks:

Task	Marks
Accepting Streamed data on streaming spark	2
Compute metrics for each match	2
Maintaining player profile	2
Clustering	2
Regression	2
Prediction of winning chances	1
Processing request and giving response	3 (1 Each)
Report	2
Viva and Demo	4

Submission Date:

- 01/12/2020

Tasks Overview

- Read the play.csv and teams.csv to initialise player and team data
- Run the python script to stream data on 6100 and read it using streaming spark
- Calculate Pass accuracy, duel effectiveness, shots on target, free kick effectiveness, fouls and own goals for each player playing as the event is received
- At the end of each match compute player contribution of each player and update player rating and chemistry between each player.
- After processing all data, save player profiles, match data, ratings and ML models.
- CUI must read JSON file with request and write JSON files with the response.

Tasks

From the streamed events data, the following metrics need to be calculated for each player who played the match as the data is received (players who were on the field at any time during the match and were not benched for the entire duration):

1. Pass Accuracy:
    a. A pass is signified by eventId = 8
    b. Pass Accuracy must be bound between 0 and 1
    c. If the “tags” field in the events JSON has:
        i. ‘id’ = 1801 it is an accurate pass
        ii. ‘id’ = 1802 it not an accurate pass
        iii. ‘id’ = 302, it is a key pass

d. Formula:

$$Pass\ Accuracy = \frac{(number\ of\ accurate\ normal\ passes + (number\ of\ accurate\ key\ passes * 2))}{number\ of\ normal\ passes + (number\ of\ key\ passes * 2)}$$

2. Duel Effectiveness
    a. A duel is signified by eventId = 1
    b. Duel Effectiveness must be bound between 0 and 1
    c. If the "tags" field in the events JSON has:
        i. 'id' = 701, duel is lost
        ii. 'id' = 702 , duel is neutral
        iii. 'id' = 703, duel is won
    d. Formula:

$$Duel\ Effectiveness = \frac{(Number\ of\ duels\ won + (Number\ of\ neutral\ duels * 0.5))}{Total\ number\ of\ duels}$$

3. Free Kick Effectiveness
    a. A free kick is signified by eventId = 3
    b. Free kick effectiveness must be bound between 0 and 1
    c. If the "tags" field in the events JSON has:
        i. 'id' = 1801 it is an accurate pass
        ii. 'id' = 1802 it not an accurate pass
    d. If the subEventId = 35, the free kick is a penalty and in such a case if tags has Id = 101, the penalty was a goal
    e. Some penalties can be effective but may not be a goal.
    f. Formula:

$$Free\ Kick\ Effectiveness = \frac{(Number\ of\ effective\ free\ kicks + number\ of\ penalties\ scored)}{Total\ number\ of\ free\ kicks}$$

4. Shots on Target
    a. A shot is signified by eventId = 10
    b. Shots on target must be bound between 0 and 1
    c. If the "tags" field in the events JSON has:
        i. 'id' = 1801, shot is on target
        ii. 'id' = 1802, shot is not on target
        iii. 'id' = 101, shot was a goal

$$Shot\ Effectiveness = \frac{(Shots\ on\ target\ and\ goals + (Shots\ on\ target\ but\ not\ gaols * 0.5))}{Total\ shots}$$

5. Foul Loss
   A foul is signified by eventId = 2. The number of a fouls a player commits during a match should be counted and will be used to penalise the player's contribution for that match.
6. Own Goal
   An own goal is signified by any event having a 'id' = 102 in the 'tags' field. The number of own goals scored by a player should be counted and will be used to penalise the player's contribution for that match.

At the end of every match, compute the following for each player

1. Player Contribution

a. This metric quantifies the contribution of a player towards their team's performance during the match and must be bound between 0 and 1.
b. Formula

$$Player\ contribution = \frac{(pass\ accuracy + duel\ effectiveness + free\ kick\ effectiveness + shots\ on\ target\ )}{4}$$

c. The above player contribution has to be normalised according to the proportion of match the player played:
    i. For players who were never substituted in or out contribution is multiplied by 1.05
    ii. For all other players it is multiplied by $\frac{minutes\ played}{90}$
    iii. For players who were on the bench for the entire match and had 0 field time, the contribution is 0 and the are not considered to have played the match, all computations are to be done only for players who played the match.

2. Player performance
   Player performance is calculated by reducing the contribution by 0.5% for every foul committed by the player and 5% for every own goal. These penalties are not compounded and are calculated on original player contribution.
3. Player Rating
    a. Initially, the rating of every player is 0.5. Before any data has been processed
    b. Formula

$$Player\ rating = \frac{(player\ performance + existing\ player\ rating\ )}{2}$$

4. Chemistry
   Chemistry between a pair of players is a measure of how well they can interact on the field. It is initially 0.5 between all pairs of players, and value must be bound between 0 and 1. At the end of every match the chemistries of each pair of players on the field have to be updated:
    a. for a pair of a players in the same team, the change in chemistry is absolute value of the average of the change in rating of both players. If rating of both players increased or decreased then chemistry increases but if rating of one player increased while that of other decreased, then chemistry decreases by same value.
    b. for a pair of a players in the opposing teams, the change in chemistry is absolute value of the average of the change in rating of both players. If rating of both players increased or decreased then chemistry decreases but if rating of one player increased while that of other decreased, then chemistry increases by same value.

   Examples:
    a. If Rashford and Salah are playing on opposite sides and their ratings go up by 0.02 and 0.06 respectively, then their chemistry will go down by 0.04
    b. If Bale and Ozil are playing on the same team and their ratings go up by 0.07 and down by 0.03 respectively, then their chemistry will go down by 0.02
    c. If Pogba and Kane are playing on opposing teams and their ratings go up by 0.07 and down by 0.03 respectively, then their chemistry will go up by 0.02

Player Profile

For each player maintain a player profile. All background data from the player.csv file must be present in the profile. Some players who were on the bench throughout the season may not have any data about them, such players are to be ignored and if the profile of any such player is requested, return {"error": "Invalid Player"}. The profile should also contain the following metrics as it stands at the end of the training data, across all matches in the training data:

- Number of Fouls
- Number of Goals
- Number of Own Goals
- Pass Accuracy
- Shots on Target

Predict Chances of Winning

For a pair of teams, comprising 11 players each predict the winning chance:

- For each player of a team, find the average of all chemistry coefficients with all other members of that team
- Multiply this consolidated chemistry coefficient with the rating of that player to give player strength
- Find the team's overall strength by averaging the player strength of the 11 players
- Predict winning chance from team overall strength from given formula

$$chance\ of\ A\ winning = \left(0.5 + strength\ of\ A - \frac{(Strength\ of\ A + Strength\ of\ B)}{2}\right) * 100\%$$

$$chance\ of\ B\ winning = 100 - chance\ of\ A\ winning$$

There are 2 problems with this approach to predicting winning chances:

- Some players may not have played enough matches in the given data to accurately assess their ratings
- Ratings of players are a function of their age and thus will change according to when match is held

Solutions:

- For the problem of players not having played enough matches in the test data used clustering:
  - Cluster players into 5 groups to approximate rating and chemistry coefficient for players who have played less than 5 matches (matches for which player was on the squad but was on the bench for entire match does not cout as played) in training data
  - Cluster based on the profile of the players
  - Take the average rating and chemistry coefficient of cluster for players who have played less than 5 matches
- For the second problem, predict the rating of a player on the date of the match using non linear regression:
  - Use quadratic regression to model the change in player's rating with age
  - When a user provides teams for a match and match date, use regression to get the player's rating at that date and the compute winning chance

User Interface tasks

The system must be able to undertake 3 tasks from the user. These tasks will in the form of reading a JSON file with a request and writing a JSON file with the response:

1. Predicting match winning chances between 2 teams:
    a. Users can provide a date of a match and 2 teams of 11 players each
    b. System must ensure that the list of players have, as mentioned in the role field of the player's data:
        1. 1 Goalkeeper (GK)
        2. At least 3 defenders (DF)
        3. At least 2 Mid fielders (MD)
        4. And at least 1 Forward (FW)
    c. Return Invalid team in case the above conditions are not met
    d. After regression make sure that the rating of any player is not below 0.2, if any player has a rating of less than 0.2, return that the player has retired
    e. If all conditions are met, return chance of either team winning the match
    f. Input:

```
{
"req_type": 1,
"date": "date in YYYY-MM-DD"
"team1":
        {
            "name": "team name" (doesn't have to match an existing team),
            "player1": "name of player",
            "player2": "name of player",
            .
            .
            .
            "player11": "name of player",

        }
"team2":
        {
            "name": "team name" (doesn't have to match an existing team),,
            "player1": "name of player",
            "player2": "name of player",
            .
            .
            .
            "player11": "name of player",

        }

}
```

    g. Output:

{
team1:
{
“name”: “team name”,
“winning chance”: CC “chance of winning as a 2-digit integer”
}
team2:
{
“name”: “team name”,
“winning chance”: CC “chance of winning as a 2-digit integer”
}
}

2. Player Profile
    a. For a given player, return their personal data and overall metrics as mentioned above as at the end of training data
    b. If a player has not played any match in training data, metrics other than background information should be 0
    c. Input:
    {
    “req_type”: 2,
    “name”: “”,
    }
    d. Output:
    {
    “name”: “”,
    “birthArea”: “”,
    “birthDate”: “”,
    “foot”: “”,
    “role”: “”,
    “height”: (integer),
    “passportArea”: “”,
    “weight”: (integer),
    “fouls”: (integer),
    “goals”: (integer),
    “own_goals”: (integer),
    “percent_pass_accuracy”: (2-digit integer),
    “percent_shots_on_target”: (2-digit integer)
    }
3. Match Info
    a. For a given match, return match data and overall values
    b. Input
    {
    “date”: “match date in YYYY-MM-DD”,
    “label”: “contains the name of the two clubs and the result of the match (e.g.,’Lazio - Internazionale, 2 – 3’)”
    }

c. Output

```
{
    “date”: ”YYYY-MM-DD”,
    “duration”: “Regular or ExtraTime or Penalties”,
    “winner”: “Team name or null in case of draw”,
    “venue”: “name of stadium”,
    “gameweek”: (Integer),
    “goals”:
        [
            {
                “name”: “player_name”,
                “team”: “name_of_team”,
                “number_of_goals”: (Integer)
            },
            ...
            {
                “name”: “player_name”,
                “team”: “name_of_team”,
                “number_of_goals”: (Integer)
            },
        ],
    “own_goals”:
        [
            {
                “name”: “player_name”,
                “team”: “name_of_team”,
                “number_of_goals”: (Integer)
            },
            ...
            {
                “name”: “player_name”,
                “team”: “name_of_team”,
                “number_of_goals”: (Integer)
            },
        ],
    yellow_cards:
        [
            “player_name”, …., “player_name”
        ],
    red_cards:
        [
            “player_name”, …., “player_name”
        ]
```

Final Report

The submission has to be accompanied by a final report which should outline your inferences and findings from the data and problems and difficulties faced while doing the project.