Mini Project 2 & 3 - Apache Spark

Overview

League of Legends (abbreviated **LoL** or **League**) is a multiplayer online battle arena video game developed and published by Riot Games. Originally inspired by Valve's Defense of the Ancients (DotA), the game has followed a freemium[1] model since its release on October 27, 2009. League of Legends is often cited as the world's largest e-sport, with an international competitive scene. The 2019 League of Legends World Championship had over 100 million unique viewers, peaking at a concurrent viewership of 44 million, with a minimum prize pool of US$2.5 million. Riot Games' 2019 revenue has been recorded at slightly more than 2 billion dollars.

As stated above, League of Legends is a free-to-play, free-to-win game, i.e. paying money won't affect your competitive skills or increase your chances of winning the game. The available purchases are only aesthetic changes and doesn't change the game at all. Riot Games' success can be attributed to the continuous analytical analysis of the player bases to update and refresh the game to be more engaging and addictive.

As in other multiplayer online battle arena (MOBA) games, each player in League of Legends controls a character ("champion") with a set of unique abilities. Most games involve two teams of five players, with each player using a different champion. The two teams compete to be the first to destroy the Nexus structure within the opposing base.

Project Goal

Inspired by Riot Games' success, this project tries to simulate the lifecycle of a real-life analytics project. The project aims to deliver useful information to players or developers that will help make profitable business decisions.

Data

You are given two forms (offline, and online) of data to use to deliver meaningful, and realistic business insights that will be translated to real-life business decisions. You will be given extra credit the more you rely on the online form, but will receive you full credit if you use the offline form. You do not need to run your code to use the online form directly, you can create your own "cache" to run the code off. The online form tests your ability to extract, clean, and summarize real-life data to fit your business needs.

Offline Form

The Dataset consists of ~180K ranked games from all regions from patch 10.9

- The dataset is JSON format, pretty-printed.

[1] "/dev: On League's Business Model – League of Legends." 25 Jan. 2017, https://nexus.leagueoflegends.com/en-us/2017/01/dev-on-leagues-business-model/

- Compression Type: zip (1,720,602,506 bytes compressed | 14,663,416,884 bytes uncompressed)
- Dataset can be found here (Google Drive), or here (Kaggle)
- Do NOT open the dataset file on an editor as it may crash or freeze your editor or your system
- zipinfo of the file is

```
Zip file size: 1720602668 bytes, number of entries: 1
-rw---- 5.1 fat 14663416884 bx defN 20-Jun-17 04:10 matches.json
1 file, 14663416884 bytes uncompressed, 1720602506 bytes
compressed:  88.3%
```

Online Form

Riot Games has match information publicly available through their online API. It has docs to explain each endpoint, and how to use it. Please be aware of the rate limits which can be found here.

Requirements

There are three categories of requirements. For each requirement, it is marked as "Batch", "Streaming", or both. The batch requirements will either use the offline dataset or the online dataset as a bonus. The streaming requirements will use the online version only or a "cached" version (i.e. download data first with code, then stream data into framework from file)

1. Basic Requirements:
    a. Write "CLEAN", well-documented code & produce indicative visualisations to:
        i. [Batch] Champion win, pick, and ban rates (bonus if per patch)
        ii. [Batch] Champion Synergies or duos (bonus if per patch)
        iii. [Batch] Item win, pick rates (bonus if per patch)
        iv. [Batch] Item Synergies (item with champion, item with class) (bonus if per patch)
        v. [Batch] Item suggestion (has to be for at least 2 champions) (bonus if per patch)
        vi. [Streaming] Win prediction for live matches
        vii. [Streaming] Threat prediction for live matches
    b. Write a "CLEAN", organized "TECHNICAL" document detailing the process needed to reach the results with focus on these points
        i. Data analysis (data problems, patterns, noise, outliers)
        ii. Challenges faced & how they were solved
        iii. Optimizations
        iv. Final design of the code detailing each part of the pipeline
        v. Any approaches should be written in this document

   c. Write a "CLEAN", organized "BUSINESS" document that is to be delivered to a business user with charts and visualizations.
2. Creative/Innovative Requirements to get more insights, information and/or suggestions (This is completely up to the students)
3. Discussions & Executions with the following process:
   a. Do experiments, trial runs, and/or section runs
   b. Discuss findings

Performance

Performance is quite important for business needs. Delivering information as soon as requested is considered a must at times. To simulate this, a percentage of the rubric is set for performance. The fastest team will get the full percentage, the slowest team will get none of the percentage. The rest of the teams will be distributed accordingly

Bonuses & Competitions

1. Having your code as the most efficient performance relative to the other teams
2. Using the online API instead of the offline form
3. Interacting with the TA usefully if you have questions, suggestions, etc.
4. Finding the TA's summoner name. More info will be provided in the video attached.

Rubric

- 50% - Basic Requirements
- 20% - Creative/innovative requirements
- 20% - Documentation
- 5% - Batch & Streaming Performance
- bonus 5% - Usage of online API
- bonus 5% - Useful interactions with TA (questions, suggestions, etc.)
- bonus 5% - Finding TA's summoner's name (This is for ALL teams, i.e. the 5% will be distributed to teams who find the summoner's name)
- - 5% per instruction broken, for any submission/delivery criteria not followed
- AN INSTANT ZERO (if you are lucky) for plagiarism/cheating/copying for all students involved

Restrictions

1. You have to use Apache Spark
2. You are not restricted to the supplied dataset, however, any other datasets to be used have to be approved by the TA first
3. NO restriction on language
4. NO restriction on utility libraries[2]

[2] "Riot API Libraries - Read the Docs." https://riot-api-libraries.readthedocs.io/en/latest/

5. NO restriction on size or architecture of the code, however, efficiency is will be taken into account

Delivering Process

Process milestones:

1. Team assignment (teams of 2)
2. First approach discussion with proposal document
3. Colab run (may be the cluster)

Deliverables:

1. Code
2. Script to run/setup the job requirements to be run on the master
3. Visualisations[3]
4. Results
5. Proposal document
6. Technical document
7. Business document

[3] "Data Is Beautiful - Reddit." 14 Feb. 2012, https://www.reddit.com/r/dataisbeautiful/.