University of Science and Technology
Communications & Information Engineering Program
CIE 427: Big Data Analytics

# Mini Project 2 - Apache Spark

## Data

Compared to data of mini project 1, data of this mini project was much better with respect to cleanliness, and organization. Data collection was not an easy job, but it also was not so difficult. The API requests made it feasible. Only a **Data Not Found** error came out sometimes when collecting data or processing it. This was due to few empty data entries. Regardless that small problem passed by a simple **try except** statement, data was so good and easy to process.

We have collected a dataset of 120,000 matches of different patches of the game. Then we filtered this dataset and kept the data of only **11.23** patch which were **72,455** matches.

## Project Challenges

Dealing with **Spark** was a challenge itself. It was difficult at first to understand the flow and how it works. A significant difficulty was the difference between **map** and **flatMap** functions. After some research we found a good YouTube video illustrating **Spark** from coding perspective that helped us a lot and coding was very easy job after finding that video. The main challenge in the project was the domain knowledge or can say our background of the game. As students who have no interest in computer games it was very difficult to understand what this data we have collected refer to. There were many terminologies that we had no idea about. But again, we found no way except to understand these terms and do the project requirements. We consulted a **LOL** geek who explained all these terms which seemed algorithms at first. After that consultancy session we were able to start thinking of our methodology and implementing it in **PySpark** codes.

## Methodology

A **PySpark** code is to implement the team methodologies in the project. Each of the requirements is similar in implementing despite some difference among them.

### Win, Pick, and Ban rate

It was calculated for each champion on some steps. First, we read the champion's name of each summoner in each match and read whether the player has won the match or not as pairs in an **RDD** called **champs**. Each element in that **RDD** was a tuple of the champion's name and an int value of **0** or **1** to represent winning status. Second, another **RDD** was generated to contain only the tuples of winning champs called **win_champs**. Each of these two **RDD**s was grouped by key, the champion's name. The list of matches related to each champion in each of the **RDD**s was converted to an integer number refer to the length of the list. Third, the two **RDD**s were joined again together. Fourth, for each champion the number of won matches was divided by the number of played matches to calculate the **win rate** for each champion.

Similarly, the pick and ban rates were calculated. Pick rate represents the number of matches a champion takes part in divided by the number of all matches in the dataset. While ban rate represents the number of times a champion is banned divided by double the number of items, because each player can be banned two times in a game.

## Champion Synergies

The synergy between each two champions was calculated thorough some heuristic that depends on the win rate of these two champions together. When two champions are most likely to win when play in the same team are assumed to have high synergy in between. First, a combination of each two champions was generated. Then, win rate was calculated for these combinations exactly as it was calculated for the champion.

## Item win and pick rate

That requirement was implemented exactly as champion win and pick rate were calculated. The number of times each item was picked is divided by the number of all items could be picked in the matches, which is **70** times the number of all matches in the dataset as each player can have **7** items. In item win rate, we divided the number of times an item was picked, and the champion has won that game divided by **70** times the number of matches in the dataset.

## Item Synergy

Item champion synergy was calculated by comparing two values. First, the percentage of winning using the item. Second, the percentage of losing using that item. If the first value is big comparing to the second one, there is some synchronization between them and hence some synergy. If the two values have small difference between them, they mean nothing. If the second value is much greater than the first, it’s against the synergy.

Item with class synergy was exactly calculated like the champion with champion synergy. The same as each other.

## Item Suggestion

For each champion, two items are suggested. This item suggestion depends on the champion item synergy calculated in the previous requirement.

## More Requirements

### Most Time Spent Living

We calculate the average of the longest time spent living for all the champions and from this statistic we can have some insights about the nature of the champions whether it can endure damage or not. According to these results, we can between champions roles in the game. For example, tank champions are ready to have hit, but assassins always need to run from these fights.

### Physical VS Magical Damage

For each champion we compare between the physical damage and magical damage it causes. Some champions are magically stronger while others are physically stronger. This is calculated through finding the average of physical and magical damage made by every champion and comparing them with each other.

## Team

### 3la Allah 7kaytna

| | |
|---|---|
| Mohammed Abdelazeem | 201700361 |
| Omar Hassan | 201700849 |