CPS803 / CPS8318 Assignment 4 (Due: 30-Nov-2025)

Ruel Gutierrez

1: Background

The dataset I used for this assignment, which I obtained from Kaggle, contains detailed information about 167 champions, playable characters, in the online multiplayer game, League of Legends (LoL) **[1]**. Specifically, it contains information about their starting in-game stats and how much they gain of said stat as they level up throughout the game. It includes numerical variables such as a champion's health, attack range, mana, armor, magic resistance, and other combat related attributes. These features influence how a champion is played and what role they typically serve within the game.

League of Legends categorizes their champions into roles such as Controller, Fighter, Mage, Marksman, Slayer, Tank, and Specialist. The goal of this model is to see how the machine would group champions based solely on their different stat distributions. My initial expected outcome was that it would group these champions into their specified roles, or by the position they're often played within the game. On the map shown below, these positions are the three lanes, the jungle in between the lanes, and the support that typically plays in the bottom lane.

[2]

2: Methods

First, the dataset was loaded from the CSV file. Champions who had missing values were removed, these were typically champions who used a resource other than mana. Because of this, those who did not use mana were excluded to maintain consistency across any mana-related attributes. Binarization was applied to the "Range type" attribute, to convert the categorical attribute into a binary numerical value: (0 = Melee, 1 = Ranged). Other categorical attributes such as Name, Tags, Role, and Resource Type were also dropped as they were not needed for clustering.

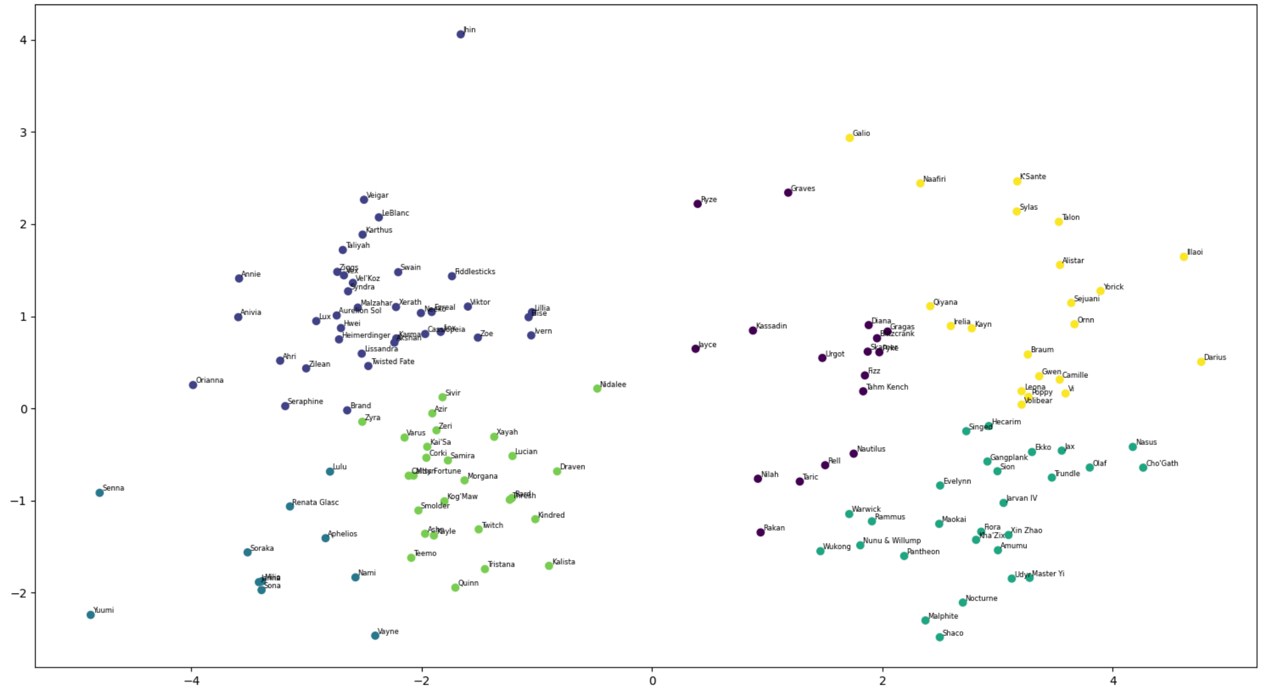
Because the different stats of a champion exist on different scales, (e.g health being in the thousands, while attack speed being in the decimals), the dataset needs to be standardized so that the machine doesn't skew heavily towards a particular attribute like health. The dataset was standardized using StandardScaler, a function within one of Sci-kit Learn's modules. The standardization scaler brings all features to a common scale by, "transform[ing] each numeric column so that it has a mean of 0 and a standard deviation of 1" **[3]**.

Because the dataset contains many features, twenty of them being relevant to the model, Principal Component Analysis (PCA) was used to reduce the dimensionality of the dataset to two components so that it could be plotted on a scatter plot. For this model, the PCA function from the Sci-kit Learn decomposition module was used to implement PCA. PCA works by "reduc[ing] the number of features in a dataset while keeping the most important information" **[4]**, which effectively reduced the twenty feature champion dataset into a two feature dataset while retaining the important information.

Now that the data has been pre-processed, the dataset is ready to be clustered. For this dataset, the K-Means clustering algorithm was favored as a general idea of how many different clusters needed was already known. Because there are six main roles in LoL, omitting Specialist as it is meant to be an outlier niche role, the K-means clustering algorithm was set to search for six clusters. The K-Means algorithm was implemented using the KMeans function under Sci-kit Learn's cluster module. It works by first randomly assigning each data point to one of the 6 clusters, then computing the centroid of each cluster and reassigning each data point to the cluster with the closest centroid. The process is repeated until the cluster assignments no longer change **[5]**.

The evaluation of the model's clustering was determined qualitatively by examining whether champions of a similar role were grouped together. For example, since many mage champions share the same stat characteristics, high mana and low durability, it is expected that these champions would be grouped together in a cluster. This role-based consistency would indicate whether the model's clustering was meaningful or not.

3: Results



The above scatter plot was the result of running the preprocessed dataset through the K-Means algorithm. While some clusters showed tight grouping, like the top left, others were more scattered like the top right and middle group.

Six clusters were produced to represent the six primary LoL roles, although not all these roles were specifically defined in the scatter plot. Certain roles, such as mages, marksmen, and supports, were more coherent and closely grouped on the left, while other roles such as tanks, slayers, and fighters were more interleaved within their clusters on the right. This suggests that many of the champion's stats overlap across roles. This is consistent with LoL's champion design, as many champions are designed to be hybrids of two or more roles, able to be played in multiple positions of the game, this is the case for many of the champions on the right side of the scatter plot.

4: Conclusions

In conclusion, the model was able to group and cluster champion roles with a decent amount of accuracy, but struggled with hybrid champions whose roles overlapped in stats like tanks, fighters, and slayers.

The K-Means algorithm used was able to capture the general role-based groupings, but can be improved on in future works. Perhaps with additional features, such as champion ability information, gameplay statistics, or common items built by a champion, the clusters may become more clear and separated. In future works, other clustering methods, such as hierarchical clustering to capture hybrid roles or DBSCAN, can be used to yield better clustering, as K-Means struggles with clustering non-spherical cluster shapes.

5: References

1. https://www.kaggle.com/datasets/cutedango/league-of-legends-champions

2. https://www.esportstalk.com/blog/eli5-league-of-legends-guide-23576/

3. https://www.geeksforgeeks.org/python/how-to-standardize-data-in-a-pandas-dataframe/

4. https://www.geeksforgeeks.org/data-analysis/principal-component-analysis-pca/

5. https://www.w3schools.com/python/python_ml_k-means.asp