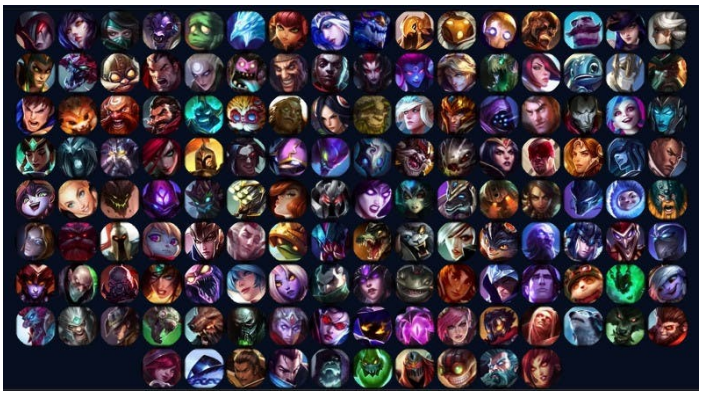
Présentation du Projet pour la certification du bloc 6 de Concepteur développeur en science des données

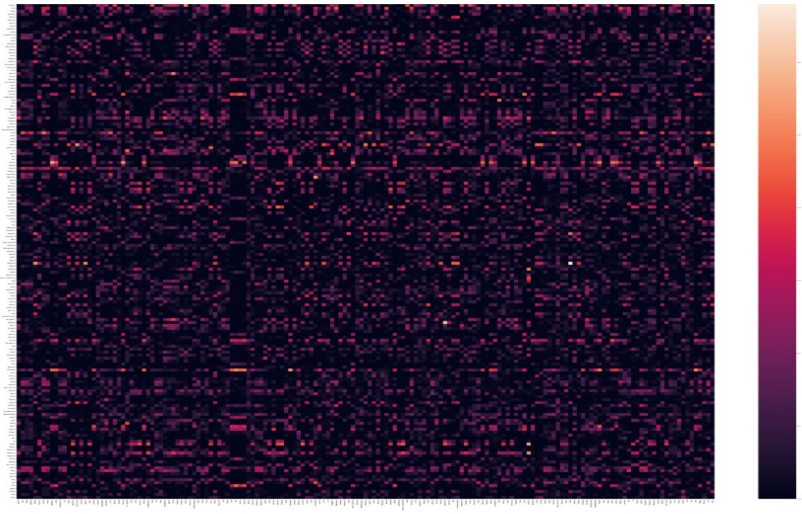
Cassiopeia



1- Présentation du projet :
- a- League of legends.
- b- Cassiopeia.
- c- Objectifs.

2- Base de données Mysql.
- *db_cassiopeia.py*

3- Outils d'extraction des données.
- *Dataframe_Cassiopeia.ipynb* :
  - *a- Found_summoners().*
  - *b- exe_getmatches().*
  - *c- champions_stats().*

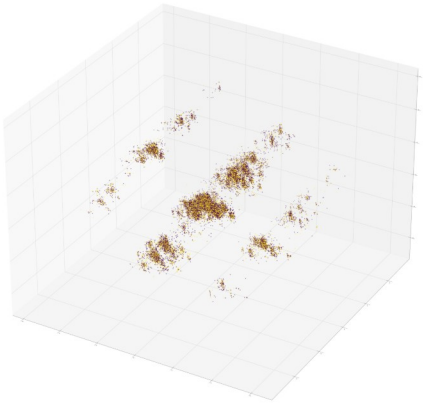


4- Outils de connexion pour Mysql.
- *connect_tools.py*

5- Statistiques matches.
- *New_statistics.ipynb :*
  - *a- champions_stats().*
  - *b- counter_stats().*
  - *c- Ncounter().*

6- RandomForestClassifier.
- *Ml_RFC.ipynb.*

7- Visualisation, pré-processing et cross validation.
- *Ml_all_cv.ipynb :*
  - *a- Visualisation 2d avec PCA.*
  - *b- Visualisation 3d avec PCA.*
  - *c- Méthode du coude pour les clusters.*
  - *d- Utilisation du clusturing pour analyse des données.*
  - *e- Recherche d'anomalies avec IsolationForest.*
  - *f- Réduction des composants du dataframe avec PCA.*
  - *g- Cross validition avec GridsearchCV.*



J'aimerais remercié Guillaume Mandersheid, formateur à JEDHA pour ces conseils et le temps qu'il m'a consacré.

League of legends est un jeu d'arène de bataille en ligne réunissant deux équipes de 5 joueurs qui se battent pour détruire le centre de la base adverse.

Il y a plus de 150 champions que peuvent sélectionner les joueurs pour former leur équipe.

Il y a 3 voies principales pour accéder à la base adverse. La rivière coupe la carte en son milieu et les espaces entre les voies et la rivière est appelé la jungle.

Pour chaque voie, un poste est confié au joueur. La jungle de chaque bord de la rivière est sous la protection d'un joueur spécifique. Et un joueur supplémentaire viens en appui des autres poste, avec une priorité sur la voie du bas.

En dehors des compétences de chaque joueur, la composition de chaque équipe est le point le plus important car il détermine l'efficacité de celle-ci durant partie.

1-b Cassiopeia :

Cassiopeia, un des personnage du jeu, est un framework pour faire des appels et manipuler les données depuis l'API de Riot Games, créateur du jeu league of legends.
Ce framework est composé de nombreuses classes qui me permettent de trouver les données utiles pour ce projet.

1-c Objectifs :

Le but de ce projet est donc d'essayer de prédire la victoire d'une équipe suivant sa composition et de trouver les influences des champions dans ces prédictions.

Ce projet est encore en cours de développement. Je souhaite améliorer ce projet pour continuer à apprendre le métier de data scientist.

Les objectifs à venir sont la création d'une base de données avec un facteur temps pour faire des prédictions dynamiques et la mise en place d'exécutable fenêtrés pour la visualisation des statistiques et ces prédictions.

Le but est de s'approcher le plus possible d'un programme livrable.

2- Base de donnés mysql.

https://github.com/simonbiver/Simon_BIVER_Projet_Cassiopeia/blob/main/Cassiopea/db_cassiopeia.py

Ayant déjà les outils en place lors de la formation à JEDHA, j'ai décidé d'utiliser un serveur local Mysql. Mais pour la suite du projet, je me servirais d'un serveur Sqlite qui me semble plus adapté pour un exécutable.

3- Outils d'extraction des données.

https://github.com/simonbiver/Simon_BIVER_Projet_Cassiopeia/blob/main/Cassiopea/Dataframe_Cassiopeia.ipynb

a- Found_summoners() :

Cette fonction va chercher les noms des joueurs les mieux classés et leurs identifiants. Le but est d'utiliser ces données pour chercher les matches leur correspondant en les stockant dans la table `summoners`.

b- exe_getmatches() :

Cette fonction utilise la table `summoners` pour trouver les matches et stocker les champions joués suivant leur position dans la table `matches`.

c- champions_stats() :

Cette fonction va chercher la liste des noms de tout les champions du jeu et l'identifiant correspondant et les stockent dans la table `champions_list`.

4- Outils de connexion pour Mysql.

https://github.com/simonbiver/Simon_BIVER_Projet_Cassiopeia/blob/main/Cassiopea/connect_tools.py

Ce module python contient des méthodes de classe pour utiliser la base de données Cassiopeia sur le serveur local Mysql .

5- Statistiques matches.

https://github.com/simonbiver/Simon_BIVER_Projet_Cassiopeia/blob/main/Cassiopea/New_Statistics.ipynb

a- champions_stats() :

Cette fonction utilise la table `matches` pour créer de statistiques, nombres de parties joués par le champions et taux de victoire pour chaque poste et équipe.

b- counter_stats() :

Cette fonction à pour but de créer un indice de fiabilité de victoire pour chaque champion et son vis à vis en tenant compte de la position de son équipe. Visualisation de la matrice 166 par 166 ainsi créer.

c- Ncounter() :

Cette fonction est sur le même principe que celle devant mais optimiser et sans prendre en compte la position de l'équipe celle si n'étant pas forcement pertinente dans le cas présent.

6- RandomForestClassifier.

https://github.com/simonbiver/Simon_BIVER_Projet_Cassiopeia/blob/main/Cassiopea/Ml_RFC.ipynb

Premier essai d'utilisation d'algorithme avec le RandomForestClassifier.

Dans un premier temps, j'ai créé un nouveau dataframe plus explicite pour l'algorithme avec des valeurs numériques traduisant le positionnement du champion dans la composition. Cela se traduit par une matrice de 2 fois 166, pour le nombre de champions possibles pour chaque équipe, plus 1 pour la cible.

Les premiers résultats ne sont pas très satisfaisant et montre un overfitting très fort. La raison est la mauvaise utilisation des hyper paramètres, prenant trop en compte le nombre d'arbres dans le modèle et pas assez la profondeur.

7- Visualisation, pré-processing et cross validation.

https://github.com/simonbiver/Simon_BIVER_Projet_Cassiopeia/blob/main/Cassiopea/Ml_all_cv.ipynb

a- Visualisation 2d avec PCA :

On peut déjà remarquer en 2d certains clusters avec un taux de victoire pour le blue side très faible.

b- Visualisation 3d avec PCA :

En 3d, les clusters deviennent beaucoup plus flagrant et certains sont clairement avec une répartition du taux de victoire déséquilibré.

On peut noter que les données sans considération pour la position des champions sont légèrement moins structuré.

c- Méthode du coude pour les clusters.

d- Utilisation du clusturing pour analyse des données :

Ayant vu des clusters avec un taux de victoire hors norme, j´utilise l'algorithme K-mean pour partitionner mes données afin de trouver les champions représentatifs de cette variance.

e- Recherche d'anomalies avec IsolationForest :

Utilisation de l'algorithme IsolationForest et essai de visualisation dans les données. Recherche dans les données d'origine pour un potentiel nettoyage.

f- Réduction des composants du dataframe avec PCA.

On peut remarquer une meilleur compression des données dans la dataframe avec la position des champions. Cela m'incite à penser que ce paramètre peut permettre un meilleur apprentissage.

g- Cross validition avec GridsearchCV.