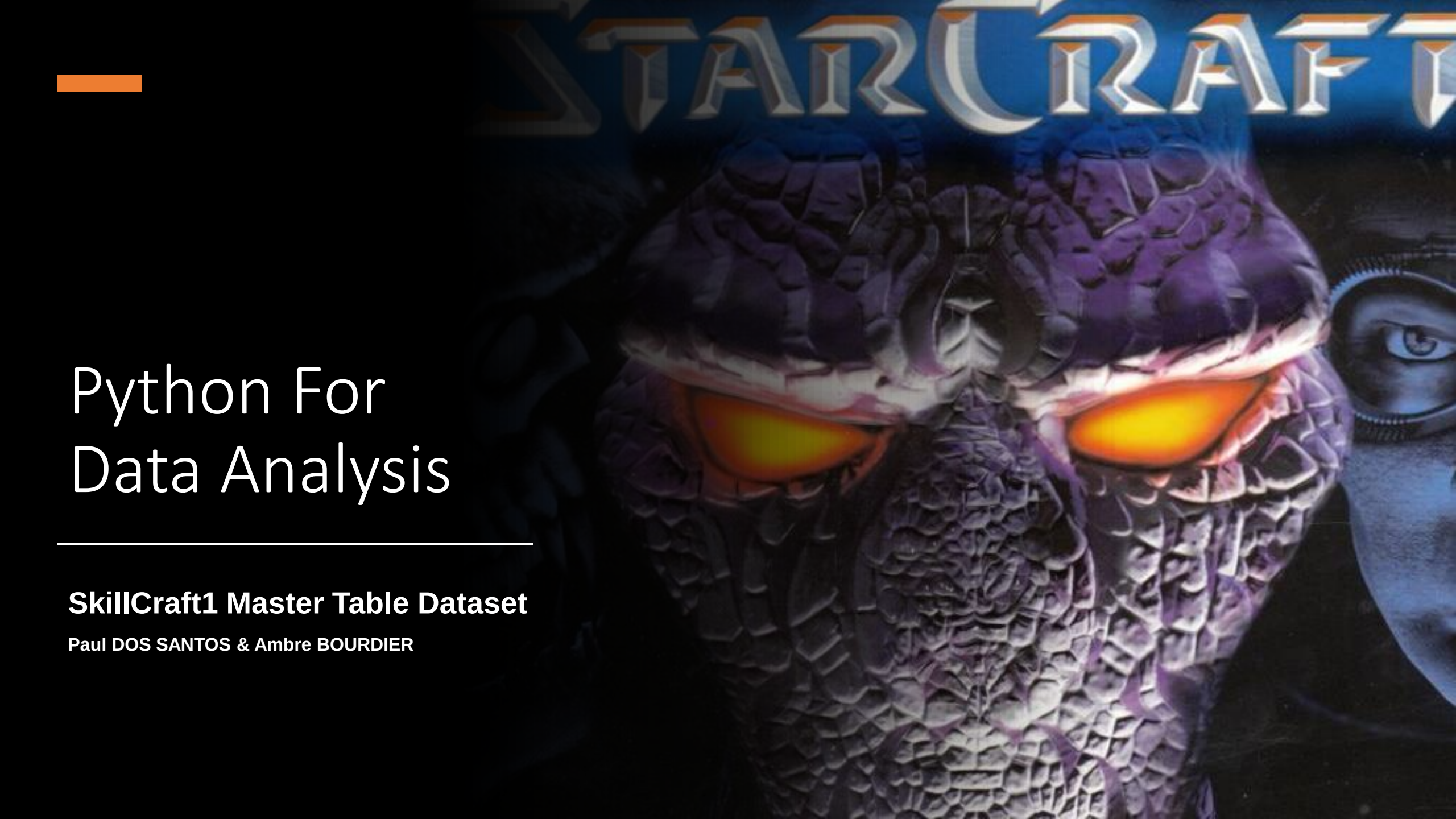
Python For Data Analysis

SkillCraft1 Master Table Dataset

Paul DOS SANTOS & Ambre BOURDIER



Objectifs

Starcraft est un jeu de stratégie multijoueur qui se concentre sur la collecte de ressources pour construire des unités et vaincre un adversaire.

Les joueurs de ce jeu jouent généralement dans une échelle en ligne qui place les joueurs dans **différentes ligues** en fonction de leurs performances contre d'autres joueurs.

L'objectif de ce projet est de ***prédire le placement dans la ligue d'un joueur Starcraft*** en utilisant uniquement les informations contenues dans notre base de donnée. La nature en temps réel du jeu nous permet de différencier les joueurs par la vitesse et l'efficacité de leurs actions en jeu. Ceci est dans le but de réduire le temps nécessaire pour arriver au bon placement.

Découverte des données

Notre base de données regroupe des informations de joueurs du jeu Starcraft1 et contient 3397 lignes, soit le nombre de joueurs sur cette période, en 2013.

Dans cette base, nous pouvons compter 20 attributs relatif à chaque joueurs :

- **GameID**: numéro d'identification unique pour chaque jeu (entier)
- **LeagueIndex**: ligues Bronze, Argent, Or, Platine, Diamant, Master, Grand Maître et Professionnel codées 1-8 (Ordinal)
- **Âge**: âge de chaque joueur (entier)
- **HoursPerWeek**: heures déclarées passées à jouer par semaine (entier)
- **TotalHours**: nombre total d'heures de lecture déclaré (entier)
- **APM**: action par minute (continue)
- **SelectByHotkeys**: nombre de sélections d'unité ou de bâtiment effectuées à l'aide de raccourcis clavier par horodatage (continu)
- **AssignToHotkeys**: nombre d'unités ou de bâtiments affectés aux raccourcis clavier par horodatage (continu)
- **UniqueHotkeys**: Nombre de raccourcis clavier uniques utilisés par horodatage (continu)
- **MinimapAttacks**: nombre d'actions d'attaque sur la minicarte par horodatage (continu)
- **MinimapRightClicks**: nombre de clics droit sur la minicarte par horodatage (continu)
- **NumberOfPACs**: nombre de PAC par horodatage (continu)
- **GapB BetweenPACs**: Durée moyenne en millisecondes entre les PAC (en continu)
- **ActionLatency**: latence moyenne entre le début d'un PAC et leur première action en millisecondes (en continu)
- **ActionsInPAC**: nombre moyen d'actions au sein de chaque PAC (en continu)
- **TotalMapExplored**: le nombre de grilles de coordonnées de jeu 24x24 vues par le joueur par horodatage (en continu)
- **WorkersMade**: nombre de SCV, drones et sondes formés par horodatage (continu)
- **UniqueUnitsMade**: Unités uniques créées par horodatage (continu)
- **ComplexUnitsMade**: nombre de fantômes et de hauts templiers entraînés par horodatage (en continu)
- **ComplexAbilitiesUsed**: capacités nécessitant des instructions de ciblage spécifiques utilisées par horodatage (en continu)

D'après l'article de recherche, les meilleurs joueurs sont souvent distingués par leur rapidité et augmentent leur efficacité en attribuant certaines fonctions à des touches de leur clavier. On s'attends donc à voir un APM croissance plus le niveau de league augmente ainsi qu'une plus forte utilisation des HotKeys.

Data Exploration

Premièrement, nous allons tenter d'étudier la répartition des joueurs et la comparer aux données réelles trouvées sur le net.

Comme les données ont été collectées depuis une communauté de jeux en ligne, il n'est pas surprenant que la proportion de données disponibles pour les matchs de la ligue Bronze et Argent soit très faible par rapport à la distribution des échelons officiels, alors que les matchs de la ligue Master dépassent sa proportion d'échelle.

En effet, les joueurs des ligues supérieures ont tendance à être plus impliqués dans la communauté, en se tenant au courant des dernières stratégies et discussions.

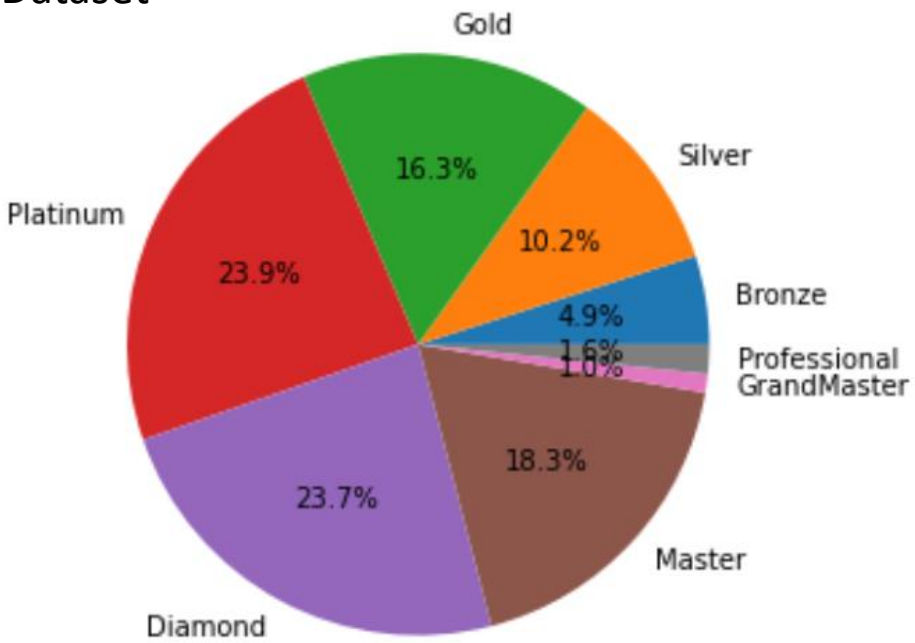
D'autre part, les joueurs de ligue inférieure sont probablement des joueurs nouveaux ou occasionnels, se connectant juste pour jouer au jeu pour le plaisir sans nécessairement s'engager avec la communauté en dehors du client du jeu.

Réalité

	World	
Grandmaster	730	(0.18%)
Master	17 958	(4.39%)
Diamond	99 296	(24.27%)
Platinum	86 632	(21.18%)
Gold	79 337	(19.39%)
Silver	90 933	(22.23%)
Bronze	34 178	(8.36%)
Population	409 064	(100.00%)

LeagueIndex	
1	4.918999
2	10.220913
3	16.288660
4	23.888071
5	23.740795
6	18.291605
7	1.030928
8	1.620029

Dataset



Data Cleaning

Dans le but d'étudier le développement de l'expertise chez un joueur de jeu vidéo, l'étude développe des techniques d'analyse des parties de jeu pour calculer plusieurs paramètres de performance moteur.

Il y a notamment le cycle de perception/action (PAC) qui représente un flux d'informations qui a lieu entre l'organisme et son environnement au cours d'une séquence de comportements guidées par des sensibilités vers un but.

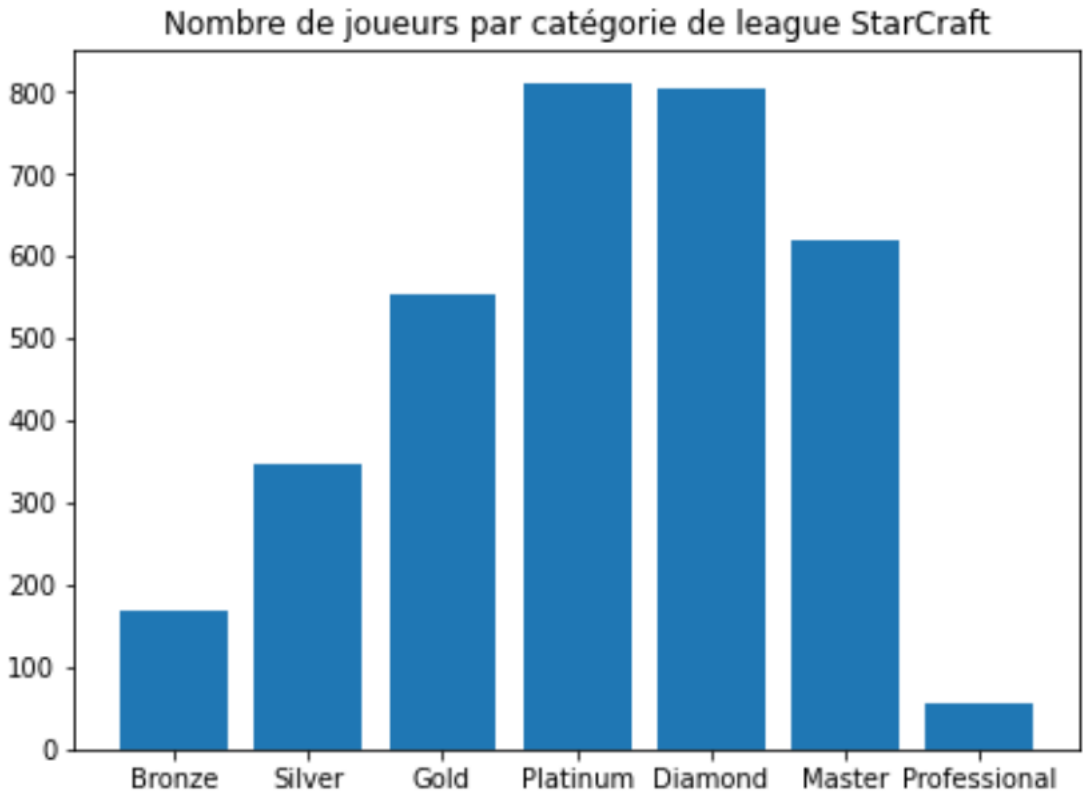
L'utilisation des HotKeys, ou raccourcis claviers est aussi pris en compte ainsi que plusieurs autres paramètres techniques du jeu. Le but de cette étude étant de modéliser l'expertise technique des joueurs de différentes Leagues, nous suivons la décision de l'article **de ne pas prendre en compte certaines colonnes telles que Age, HoursPerWeek, et TotalHours.**

Nous avons créé une **nouvelle colonne League Category** avec le nom de chaque league ainsi qu'une colone New Index qui réordonne les indexes pour qu'ils débutent à zéro comme cela est préférable pour la modélisation.

Dans l'article une question se pose concernant l'inégalité des valeurs entre les différentes leagues et leur influence sur l'importance des variables.

Les chercheurs testent donc les mêmes modèles sur un dataset équilibré à 125 colonnes pour chaque (excluant les professionnels).

Ils observent néanmoins les mêmes variations des rangs d'importance des attributs. Nous décidons selon ces constats de **ne pas équilibrer les nombres de données par League pour ne pas perdre trop d'information**.



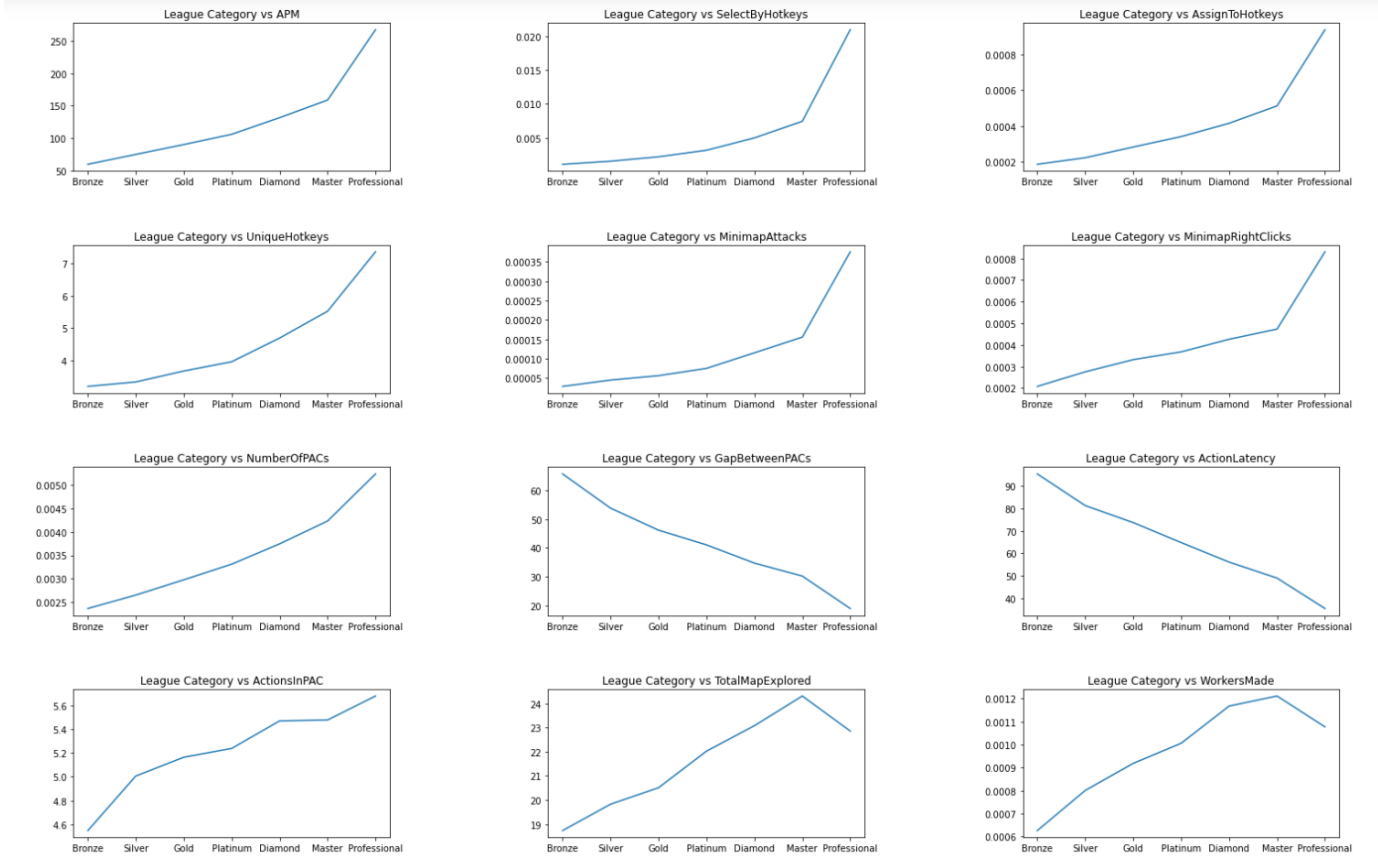
Data Visualisation

Suite à notre cleaning, nous avons pu afficher la variation des attribut en fonction des leagues à prédire.

Ce que nous avons pu remarquer directement est que dans l'ensemble les variables évoluent de façon linéaire avec le niveau de la league.

Nous avons donc pu grâce à cette visualisation, nous rendre compte de l'évolution et de la corrélation entre chaque attributs et le niveau de la league à prédire.

Cela nous a également permis d'identifier les modèles que nous allions appliquer, notamment la régression linéaire.

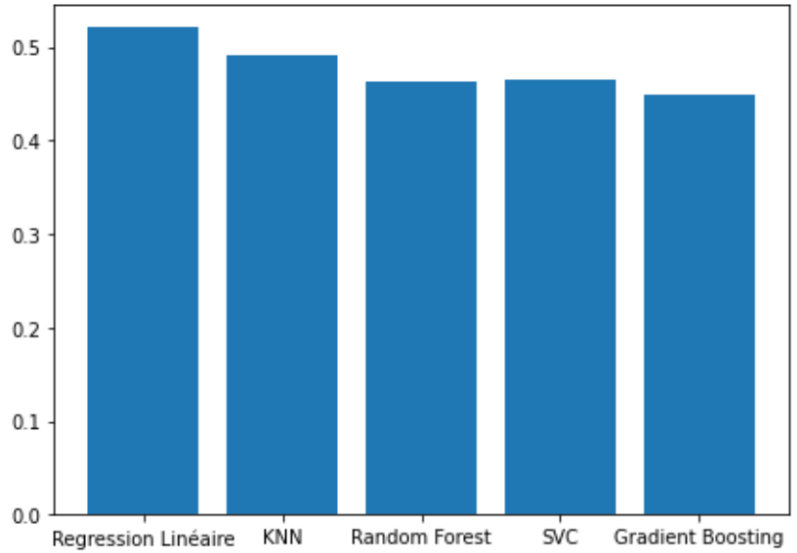


Modèles

Pour nos modèles, nous allons utiliser :

- Régression Linéaire
- KNN
- Random Forest
- SVC
- Gradient Boosting

Afin d'obtenir les meilleurs algorithmes possibles, nous les avons testés en faisant varier les hyperparamètres.
Nous avons ensuite pu comparer les performances des modèles avec le R2 et les voici ci-contre :



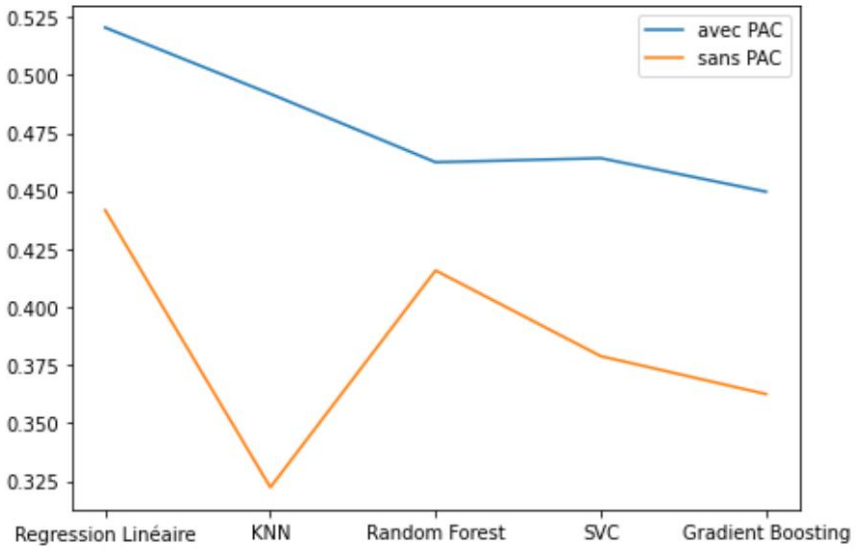
On relance le modèle, cette fois-ci sans les données dépendantes du cycle Perception/Action (PAC) soit les quatre données suivantes :

- NumberOfPACs: Number of PACs per timestamp (continuous)
- GapBetweenPACs: Mean duration in milliseconds between PACs (continuous)
- ActionLatency: Mean latency from the onset of a PACs to their first action in milliseconds (continuous)
- ActionsInPAC: Mean number of actions within each PAC (continuous)

On obtient les meilleurs résultats avec la régression linéaire.

Nous remarquons également que lorsque nous retirons PAC, le moins bon modèle est le KNN.

Comme soutenu dans l'article, nous remarquons sur le deuxième graphique que le PAC apporte beaucoup aux modèles en terme de performance.

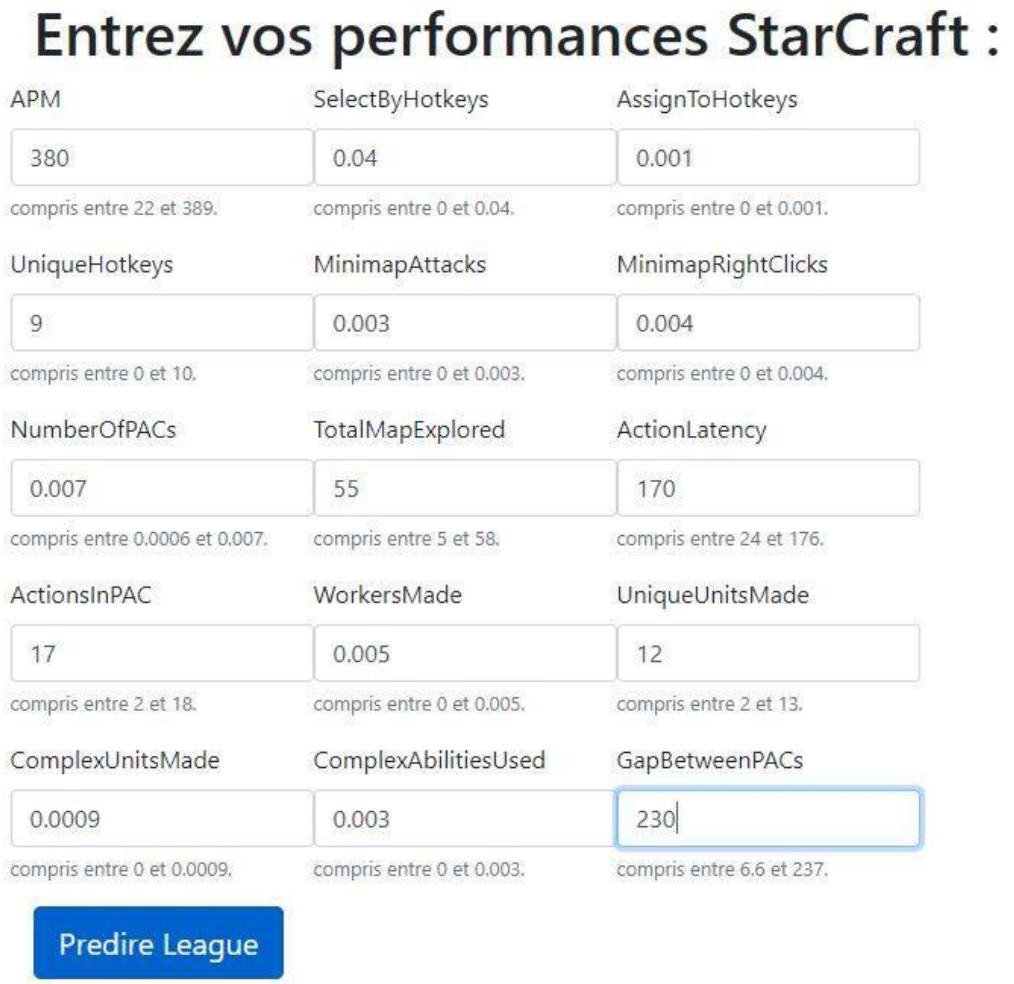


API FLASK

Pour finir, nous avons développer une API Flask afin de mieux représenter notre prédiction.

Pour effectuer cette prédiction, nous avons utilisé le modèle de régression linéaire qui avait les meilleures performances.

L'API consiste en une requête GET qui récupère les données entrées par l'utilisateur et renvoie la prédiction de la régression linéaire sur le page résultat.



Conclusion & Perspectives d’évolution

Notre ensemble de données ne contient actuellement que 3395 lignes de données, ce qui correspond à une très petite proportion de la base de joueurs existante.

Cela a également entraîné un déséquilibre dans la répartition des ligues dans les données qui nous a amené à fusionner plusieurs ligues ensemble.

Ces modifications peuvent rendre le modèle biaisé pour être précis dans la prédiction des placements dans les ligues. Les caractéristiques trouvées dans les données à elles seules ne sont probablement pas suffisantes pour prédire très précisément le placement dans la ligue d'un joueur. Ils sont très concentrés sur les interactions physiques du joueur avec le jeu, en plus de cela, il y a de nombreux chevauchement entre les ligues ce qui peut induire les modèles en erreur (notamment les KNNs).

Lors de la création des modèles, nous avons choisi d'utiliser de nombreuses combinaisons d'hyperparamètres, puis affiner le modèle par la suite. Bien que cela semble avoir été efficace dans une certaine mesure, nous pourrions potentiellement améliorer notre puissance prédictive en utilisant une analyse approfondie pour sélectionner manuellement plus de caractéristiques et de valeurs d'hyperparamètres.

Avec une prédiction fiable à 50%, nous ne pouvons pas conclure sur l’efficacité certaine de notre modèle. Néanmoins, elle est plus rapide et efficace que si nous répartissions les joueurs nous même, un par un.