

Rapport technique d'évaluation

PARIS SPORTIFS: BEAT THE BOOKMAKERS (Tennis)



Promotion: DA_avril 22

Auteurs: Lucie Duhin, Channy Kheum, Isra Phetramphand

Mentors: Emilie Greff, Zakary Saheb, Gaspard Grimm

Table des Matières

Contexte

Objectif

Étape 1: Exploration des données

1- Base de données

2- Statistiques exploratoires

1.1- Variables quantitatives

- a) La moyenne et la médiane
- b) Les quantiles
- c) Les valeurs minimums et maximums
- d) La variance et l'écart-type
- e) Corrélation entre les variables

1.2- Variables qualitatives

- a) Les modalités les plus fréquentes
- b) Les modalités des variables qualitatives
- c) La table de contingence
- d) Corrélation entre les variables qualitatives

2- Visualisation des données

- a) Nombre de matchs joués par an
- b) Top 20 des meilleurs gagnants
- c) Nombre de set gagnés par match en tant que gagnant /perdant
- d) Elo winner, elo loser, proba elo
- e) Performance des prédictions des matchs

Étape 2: Modélisation statistique

1- Préparation des données

- a) Encodage des variables catégorielles
- b) Retraitement de la variable 'Winner' et 'Loser'
- c) Autres retraitement
- d) Base de modélisation

2- Modélisation de classification

a) modélisation: utilisation des classifieurs sans hyperparamètre

- modélisation 1: df_atp_final
- modélisation 2: standardisation avec MinMax scaler
- Interprétation des modèles sans moyenne mobile et sans hyperparamètre
- conclusion de la modélisation sans hyperparamètre

b) Modélisation avec optimisation des paramètres: travail sur les hyperparamètres

- Conclusion sur la modélisation avec paramètres optimisés:

c) Modélisation avec la moyenne mobile

Étape 3: Web Scraping

1 – Le Web Scraping

- a) Qu'est ce que le web scraping ?
- b) Pourquoi l'utiliser dans notre projet ?

c) Où utiliser le web scraping dans notre cas ?

d) Choix du web scraper

2 – Application au web scraping

a) Setup de Sélénium

b) Localiser l'élément à scraper et que scraper

c) Scrapping des variables tennistiques

d) Scrapping des statistiques personnelles des joueurs sur l'année 2016

3- modélisation avec statistiques scrapées

a) préparation du dataframe et réalisation de la jointure entre les 2 df

- Df_joueur: étape de text mining
- Df_stats2016: étape de préparation du dataframe
- Jointure entre le df_stats2016_final et le df_joueur

b) modélisation avec les statistiques scrapées

- modélisation sans moyenne mobile
- interprétation des résultats
- Modélisation avec moyenne mobile

4- Interprétabilité

a) L'arbre de décision

b) Les variables les plus importantes

c) Les variables les plus importantes - web scraping

Conclusion

Lexique

Définition des variables de la base de donnée 'atp_data.csv':

Définition des variables appartenant au code

Définition des variables du site: <https://www.ultimatetennisstatistics.com/>

Bibliographie / référence internet

Annexes

Contexte

D'après l'Autorité Nationale des Jeux, l'année 2021 a montré une grande croissance des jeux d'argents et ce particulièrement grâce aux paris sportifs en ligne.

En effet, rien qu'en 2021, le produit brut des jeux PBJ, c'est à dire la part des mises empochées par les sociétés de jeux, a dépassé pour la première fois le milliard d'euros en atteignant 1,4 milliard d'euros contre 940 millions d'euros en 2020, cela représente une croissance de 44% en 1 an uniquement.

Ce n'est pas tout, les mises ont aussi connu une croissance extraordinaire de 47% en passant de 5,3 milliard d'euros en 2020 à 7,9 milliard d'euros en 2021.

Cela peut s'expliquer en principe par la hausse des parieurs en ligne, les CJA ou comptes joueurs actifs, avec aujourd'hui 4,5 millions de français en 2021 qui font des paris sportifs en ligne contre 3,9 millions en 2020, soit une augmentation de 17%.

Le nombre de paris sportifs est encore en expansion et pourrait largement augmenter à l'aide des divers évènements sportifs à venir (Coupe du monde de football 2022, Jeux Olympique d'été en France en 2024...).






Le pari sportif est un jeu d'argent consistant à anticiper le résultat d'un événement lors d'une rencontre sportive. Il existe différents types de paris sportifs dont le plus simple consiste à miser sur la victoire d'une équipe ou d'un sportif.

Chaque pari a une cote bien définie qui permet de connaître à l'avance le montant gagné si l'événement se produit.

Les paris sont proposés par des bookmakers (ou sites de paris en ligne) qui sont des organismes autorisés à proposer aux joueurs de parier. L'ensemble de ces sites ou bookmakers sont régulés par l'autorité nationale des jeux (celle-ci distribuant les agréments officiels aux opérateurs sur le territoire français par exemple).

Il existe une multitude de sites de paris en ligne chacun ayant ses spécificités (diversité des paris, personnalisation des paris, bonus de bienvenue, paris en direct....) . Ainsi le parieur en ligne peut choisir l'opérateur qui convient à ses attentes.

Notre sujet abordé dans ce rapport traite des paris sportifs concernant le tennis masculin. Pour ce sport, selon une étude de Parieztennis , le top 5 des meilleurs bookmakers pour le paris sportif tennistique sur le territoire français est le suivant:

 100€ de paris gratuits et 500€ cashback ★★★★★ VOIR WINAMAX	 1er pari perdant remboursé jusqu'à 100€ ★★★★★ VOIR NETBET	 Jusqu'à 150€ de paris offerts ★★★★★ VOIR UNIBET	 1er pari perdant remboursé jusqu'à 100€ ★★★★★ VOIR BETCLIC	 100€ de paris gratuits et 150€ de cashback ★★★★★ VOIR BARRIERBET
✓ En moyenne, les cotes les plus élevées en France ✓ Programme fidélité VIP très généreux ✓ Le plus gros bonus, facile à obtenir	✓ Le plus généreux pour les paris gratuits ✓ Booste les gains des paris combinés jusqu'à 100% ✓ Dispose du moyen de paiement ApplePay	✓ Le meilleur streaming du marché ✓ Le plus complet pour le live-betting ✓ Personnalisation possible des paris avec Mybet	✓ Appli mobile très bien conçue ✓ cotes performantes sur les paris secondaires du type tie-break ✓ Les principaux tournois en streaming	✓ Panel de paris très développé ✓ Nombreux avantages et cadeaux offerts dans les casinos Barrière ✓ Bonus élevé et facile à obtenir
✗ Site web à améliorer. Opérateur clairement orienté mobile ✗ Service client par mail, pas de tchat, ni téléphone	✗ Cotes pas toujours au top sur les paris dits secondaires ✗ offre peu garnie en live-betting	✗ Service client peu agréable ✗ Limites de mises fréquentes, opérateur assez tatillon	✗ N'offre quasiment jamais de paris gratuits ✗ Absence de grilles et de cotes boostées	✗ Mauvaise fonctionnalité et ergonomie du site et de l'appli mobile ✗ Pas de diffusion de matchs

Top 5 des bookmakers en France selon le site parieztennis.fr

Objectif

L'objectif de ce projet est de construire un modèle de classification qui va prédire au mieux les joueurs gagnants et les joueurs perdants au cours de différents tournois et des matchs de tennis. L'estimation de la probabilité d'un joueur gagnant prédit par notre modèle doit battre les algorithmes des bookmakers 'Pinnacle Sport' et 'Bet 365'.

La première étape consiste à effectuer des recherches sur internet sur la source des données du fichier Excel 'atp_data.csv' (source: [ATP matches dataset | Kaggle](#)) mis à notre disposition et représentant l'ensemble des matchs de tournois entre 2000 et 2018. Sur un autre site, on peut y trouver les définitions des variables du fichier sur le site [Tennis results and betting odds data for tennis betting system development \(tennis-data.co.uk\)](#).

La seconde étape est l'exploration des données du fichier pour définir la liste des variables, les types de variables qualitatives ou quantitatives, une vision des modalités et à l'aide des études statistiques adaptées, déterminer le niveau de corrélation ainsi que des visualisations de données pour approfondir les liens entre les variables.

Après une bonne compréhension des données vient la partie de nettoyage, d'encodage, de normalisation des variables et de création d'identifiants uniques dans le but d'obtenir une base de données exploitable pour l'entraînement des différents algorithmes de classification.

Une fois que la partie de modélisation achevée vient la phase d'interprétabilité des modèles afin de mieux comprendre le lien entre la variable cible et les variables explicatives.

En complément, nous avons développé un programme de web scraping pour récupérer des statistiques complémentaires sur les joueurs afin d'améliorer les performances de notre modélisation.

Enfin la dernière étape du projet consiste à estimer la probabilité du joueur gagnant dans le but de confronter nos résultats avec ceux des bookmakers.

Étape 1: Exploration des données

1- Base de données

Pour mener à bien notre projet paris sportif, nous avons utilisé et exploité la base de données mise à disposition sur le site Kaggle: <https://www.kaggle.com/edouardthomas/atp-matches-dataset>

Les données disponibles sont séparées en 2 fichiers csv:

- atp_data.csv
- confidence_data.csv

En voici le détail:

Tableau des sources de données

nom du fichier	nombre d'attributs (= nombre de colonnes)	nombre d'enregistrement (nombre de lignes)
atp_data.csv	23	44708
confidence_data.csv	5	11054

Nous avons choisi de nous concentrer sur le fichier Excel 'atp_data.csv' car le fichier est composé des noms des joueurs gagnants et perdants. C'est l'information essentielle pour construire notre modélisation statistique. A contrario, le fichier Excel 'confidence_data.csv' comporte les variables suivantes: le numéro du match, la probabilité des joueurs gagnants de 'Pinnacle Sport', win0, confidence et une date. On constate bien que les variables sont moins importantes que le fichier ATP.

L'audit des données a été réalisé dans un environnement Google Colab à l'aide de la création des drive permettant l'importation des fichiers de Kaggle ainsi que le dézippage.

Il est possible de récupérer l'ensemble des données de 2 façons différentes:

- option 1: en téléchargeant le fichier .csv en local et en le chargeant dans le notebook,
- option 2: en utilisant une clé et un fichier .json permettant de récupérer les données directement sur le site kaggle

Après le chargement du fichier Excel dans le notebook Jupyter du collab qui se nomme 'Paris_sportif_Beat_the_Bookmakers.ipynb', nous avons réalisé un premier affichage du dataframe brut que nous avons nommé 'df_atp'.

Ensuite, nous avons affiché les informations de la base de données ATP afin d'observer le volume de cette base, la liste des noms des variables et identifier le type de variables qualitatives et quantitatives constituant le dataframe. Cela nous a également permis de constater que pour certaines variables, il existe des données manquantes.

Ci-dessous le résultat des informations du dataframe 'df_atp':

Liste des variables du df_atp

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44708 entries, 0 to 44707
Data columns (total 23 columns):
#   Column          Non-Null Count  Dtype
---  -
0   ATP              44708 non-null  int64
1   Location         44708 non-null  object
2   Tournament       44708 non-null  object
3   Date             44708 non-null  object
4   Series           44708 non-null  object
5   Court            44708 non-null  object
6   Surface          44708 non-null  object
7   Round            44708 non-null  object
8   Best of          44708 non-null  int64
9   Winner           44708 non-null  object
10  Loser            44708 non-null  object
11  WRank            44708 non-null  int64
12  LRank            44708 non-null  int64
13  Wsets            44521 non-null  float64
14  Lsets            44521 non-null  float64
15  Comment          44708 non-null  object
16  PSW              32743 non-null  float64
17  PSL              32743 non-null  float64
18  B365W            39037 non-null  float64
19  B365L            39057 non-null  float64
20  elo_winner       44708 non-null  float64
21  elo_loser        44708 non-null  float64
22  proba_elo        44708 non-null  float64
dtypes: float64(9), int64(4), object(10)
memory usage: 7.8+ MB
```

Les informations que nous pouvons lire à travers les résultats sont le volume du dataframe qui est de 44.708 lignes. On constate qu'il y a 23 variables dont 13 variables quantitatives de type float64(= décimales) et int64 (= entier) et 10 variables qualitatives.

On observe que les probabilités des bookmakers des joueurs gagnants et perdants ont des données manquantes car le volume de données pour les probabilités de 'Pinnacle Sport' est de 32.743 lignes et pour 'Best 365' 39.057 lignes. De plus, les sets des joueurs gagnants et perdants sont aussi manquants car on constate seulement 44.521 lignes.

Concernant les valeurs manquantes, nous avons décidé de les gérer de la manière suivante:

- Les variables 'Wsets' et 'Lsets' constituent les sets(ou manches) des joueurs gagnants et perdants qu'ils ont gagnés au cours d'un match. A l'issue d'un match, les parieurs ne sont pas censés connaître le score final d'un match, c'est pourquoi, nous avons choisi de supprimer ces variables de notre base d'étude. Si elles avaient été dans notre entraînement, cela aurait potentiellement donné un résultat biaisé de notre modélisation.
- Les probabilités des bookmakers 'PSW', 'PSL', 'B365W', 'B365L', ne seront pas supprimés de la base d'étude car nous pensons qu'elles n'auront pas d'impact sur la modélisation. De plus, ces données pourront servir à comparer la qualité des résultats de notre modélisation.

A l'aide du site, <http://tennis-data.co.uk/notes.txt>, nous avons pu définir l'ensemble des variables du fichier Excel 'atp_data.csv' que vous trouverez dans le lexique.

Cette première phase ainsi que quelques recherches internet nous a permis de comprendre de manière générale la composition du fichier Excel 'atp_data.csv': cette base de données correspond à l'ensemble des tournois ATP des joueurs masculins sur plusieurs périodes que nous allons développer à l'aide de visualisations.

Nous avons également très vite compris que notre variable cible sont les joueurs gagnants car le but du projet est d'avoir la meilleure estimation des joueurs gagnants que les bookmakers 'Pinnacle Sport' et 'Bet365' qui sont présents dans la base de données. Cependant, nous pouvons constater à l'aide de **la liste des variables du df atp** que les modalités des joueurs gagnants sont de type catégorielle. Il va falloir traiter cette variable catégorielle en numérique pour l'entraînement de notre modèle.

Ensuite, l'ensemble des variables explicatives semblent être toutes les variables qui conditionnent le niveau de jeu d'un joueur dans un match. Nous avons pensé aux variables sur les types de terrains, de surface, le classement ATP des joueurs, la série ATP dans laquelle les joueurs se situent et la probabilité elo.

C'est à l'aide de l'étude des variables quantitatives et des variables qualitatives que nous allons déterminer la qualité et la répartition des modalités des données ainsi que la corrélation entre les différentes variables. L'étude des variables qui composent la base de travail est très importante pour la préparation de la base d'entraînement de notre modélisation statistique.

2- Statistiques exploratoires

1.1- Variables quantitatives

L'étude des variables quantitatives est nécessaire afin d'observer l'étendue des informations et la pertinence des données. Pour cela, observons les indicateurs de positions c'est-à-dire la moyenne, la médiane, les quantiles, les valeurs minimums et maximums du dataframe 'df_atp'.

a) La moyenne et la médiane

La moyenne résume une liste de données numériques en une seule valeur, tandis que, la médiane est la valeur du milieu d'un jeu de données c'est-à-dire que 50% des unités ont une valeur inférieure ou égale à la médiane et 50% des unités ont une valeur supérieure ou égale.

Analyse de la moyenne et de la médiane

	num_moyenne	num_median	diff_moy_med	
elo_winner	1684.023280	1652.866073	31.157207	1
LRank	93.674108	65.000000	28.674108	
elo_loser	1608.755552	1580.743828	28.011724	1
WRank	59.159681	41.000000	18.159681	
PSL	4.240179	2.660000	1.580179	
B365L	3.551007	2.500000	1.051007	
ATP	32.803659	32.000000	0.803659	
Lsets	0.406325	0.000000	0.406325	
PSW	1.927563	1.549000	0.378563	
Best of	3.372596	3.000000	0.372596	
B365W	1.822246	1.500000	0.322246	
Wsets	2.140630	2.000000	0.140630	
proba_elo	0.585594	0.589218	0.003624	

Nous constatons que le calcul de la moyenne et la médiane sur les variables quantitatives de notre dataframe sont quasi proches. De plus, le calcul de la différence entre la moyenne et la médiane permet de constater que les écarts entre les indicateurs de positions ne sont pas très éloignés.

Nous pouvons en déduire que les variables quantitatives sont homogènes dans notre base de données. Cela indique aussi qu'il n'existe pas de valeurs extrêmes et donc qu'il n'y a pas de retraitement spécifique à effectuer sur les données.

Nous observons que l'écart est inférieur à un ou bien proche de un pour toutes les autres variables sauf pour les variables "elo_winner", "LRank", "elo_loser" et "WRank". Les variables "LRank" et "WRank" sont les niveaux des rangs des joueurs gagnants et perdants.

L'écart du rang des joueurs perdants est supérieur à celui des joueurs gagnants ($28,67 > 18,16$). Cela semble être cohérent car dans le jeu de tennis, au fur et à mesure des tournois, il y'a d'avantages de joueurs perdants que des joueurs gagnants.

Ensuite, les variables “elo_winner” et “elo_loser” correspondent à un classement ELO qui est un système d'évaluation comparatif du niveau de jeu des joueurs gagnants et perdants. Selon cet article de Pinnacle: **'Tournoi de l'US Open 2016: classement ATP et classement elo pour le tennis'**, le classement ELO pour le tennis est une méthode précise pour prédire la probabilité de victoire contre le classement ATP qui détermine l'entrée et la progression des joueurs dans tous les tournois. De plus, le classement ELO semble être plus efficace que le classement ATP. A travers nos résultats, le niveau de jeu des joueurs gagnants est supérieur à celui des joueurs perdants (31,16 > 28, 01). On comprend bien que plus le niveau du joueur est élevé et plus il a des chances de gagner un match d'un tournoi ATP.

b) Les quantiles

Nous pouvons approfondir l'analyse des variables quantitatives avec le calcul des quantiles. Ce sont les valeurs qui divisent un jeu de données en intervalles contenant le même nombre de données. De manière générale, on divise le jeu de données en 4 de telle sorte que le premier quantile renvoie la valeur à partir de laquelle on retrouve 25% des valeurs et ainsi de suite pour les autres quantiles.

Analyse des quantiles

	num_moyenne	num_median	diff_moy_med	q1	q2	q3
elo_winner	1684.023280	1652.866073	31.157207	1548.476977	1652.866073	1780.755524
LRank	93.674108	65.000000	28.674108	35.000000	65.000000	105.000000
elo_loser	1608.755552	1580.743828	28.011724	1501.546103	1580.743828	1684.607134
WRank	59.159681	41.000000	18.159681	17.000000	41.000000	77.000000
PSL	4.240179	2.660000	1.580179	1.794000	2.660000	4.270000
B365L	3.551007	2.500000	1.051007	1.720000	2.500000	4.000000
ATP	32.803659	32.000000	0.803659	19.000000	32.000000	49.000000
Lsets	0.406325	0.000000	0.406325	0.000000	0.000000	1.000000
PSW	1.927563	1.549000	0.378563	1.270000	1.549000	2.140000
Best of	3.372596	3.000000	0.372596	3.000000	3.000000	3.000000
B365W	1.822246	1.500000	0.322246	1.220000	1.500000	2.000000
Wsets	2.140630	2.000000	0.140630	2.000000	2.000000	2.000000
proba_elo	0.585594	0.589218	0.003624	0.447921	0.589218	0.737288

Nous avons divisé le jeu de données en 3 quartiles dont un premier quartile de 0,25, le second quartile de 0,50 et le dernier quartile de 0,75. On constate que les écarts ne sont pas très importants pour l'ensemble des variables. Mais pour les variables sur les rangs des joueurs gagnants et perdants entre le second et le troisième quartile, il existe un écart mais celui-ci n'est pas excessif. Nous pouvons en conclure que l'analyse des quantiles confirme que les données sont homogènes dans la base de données ATP et que nous n'avons pas de données extrêmes.

c) Les valeurs minimums et maximums

Les autres indicateurs de positions qui peuvent être intéressant à observer sont les valeurs minimums et maximums. Ils donnent généralement un aperçu de l'étendue de la répartition des valeurs pour l'ensemble des variables quantitatives.

Analyse des valeurs minimums et maximums

	num_moyenne	num_median	diff_moy_med	q1	q2	q3	num_min	num_max	diff_min_max
WRank	59.159681	41.000000	18.159681	17.000000	41.000000	77.000000	0.000000	2000.000000	-2000.000000
LRank	93.674108	65.000000	28.674108	35.000000	65.000000	105.000000	0.000000	2000.000000	-2000.000000
elo_winner	1684.023280	1652.866073	31.157207	1548.476977	1652.866073	1780.755524	1318.945207	2392.408923	-1073.463716
elo_loser	1608.755552	1580.743828	28.011724	1501.546103	1580.743828	1684.607134	1327.551888	2392.595567	-1065.043679
PSL	4.240179	2.660000	1.580179	1.794000	2.660000	4.270000	1.010000	121.000000	-119.990000
B365L	3.551007	2.500000	1.051007	1.720000	2.500000	4.000000	1.002000	101.000000	-99.998000
ATP	32.803659	32.000000	0.803659	19.000000	32.000000	49.000000	1.000000	69.000000	-68.000000
PSW	1.927563	1.549000	0.378563	1.270000	1.549000	2.140000	1.000000	46.000000	-45.000000
B365W	1.822246	1.500000	0.322246	1.220000	1.500000	2.000000	1.000000	29.000000	-28.000000
Wsets	2.140630	2.000000	0.140630	2.000000	2.000000	2.000000	0.000000	3.000000	-3.000000
Best of	3.372596	3.000000	0.372596	3.000000	3.000000	3.000000	3.000000	5.000000	-2.000000
Lsets	0.406325	0.000000	0.406325	0.000000	0.000000	1.000000	0.000000	2.000000	-2.000000
proba_elo	0.585594	0.589218	0.003624	0.447921	0.589218	0.737288	0.008899	0.995601	-0.986703

L'analyse des valeurs minimums et maximums démontre que l'étendue de la répartition des valeurs est très importantes pour les variables suivantes: 'WRank', 'LRank', 'elo_winner', 'elo_loser'. Ce sont exactement les mêmes variables que l'on a identifiées dans le calcul de la moyenne, de la médiane et des quartiles.

On peut en déduire qu'il existe des joueurs gagnants et perdants qui peuvent avoir un rang à 0 et que le rang maximum est de 2000. Les visualisations des données vont sûrement nous aider à comprendre ce qu'il se passe dans la base de données concernant ces variables.

Ensuite, le niveau de jeu des joueurs gagnants et perdants ont des valeurs minimums et maximum quasi proches. Il est aussi pertinent de faire des visualisations sur ces variables pour comprendre ce que cela signifie.

Les études des variables quantitatives ont démontré l'importance des données sur les rangs et le niveau de jeu des joueurs gagnants et perdants dans la base de données. En plus de ces variables, on constate que l'étendue des valeurs minimums et maximums pour les bookmakers existe à cause des valeurs manquantes.

d) La variance et l'écart-type

En plus des indicateurs de positions, il existe des mesures de dispersions pour analyser les variables quantitatives. On parle de la variance qui mesure la dispersion des valeurs autour de la moyenne, c'est-à-dire à quel point les valeurs quantitatives présentes dans la base de données sont écartées de la moyenne. Généralement, on observe l'écart-type, qui se définit mathématiquement comme la racine carrée de la variance. L'interprétation de l'écart-type est quasi proche de celle de la variance. Si l'écart-type est élevé, cela signifie que les valeurs sont dispersées autour de la moyenne et inversement.

Analyse de l'écart-type

	mean	std	I1	I2
ATP	32.803659	18.170565	14.633094	50.974224
Best of	3.372596	0.778702	2.593893	4.151298
WRank	59.159681	74.881003	-15.721322	134.040683
LRank	93.674108	124.987643	-31.313536	218.661751
Wsets	2.140630	0.462388	1.678242	2.603019
Lsets	0.406325	0.557349	-0.151023	0.963674
PSW	1.927563	1.359136	0.568427	3.286700
PSL	4.240179	5.744329	-1.504150	9.984508
B365W	1.822246	1.107547	0.714698	2.929793
B365L	3.551007	3.498689	0.052319	7.049696
elo_winner	1684.023280	179.246422	1504.776859	1863.269702
elo_loser	1608.755552	137.949299	1470.806253	1746.704851
proba_elo	0.585594	0.198732	0.386862	0.784326

L'analyse du calcul de l'écart-type pour l'ensemble des variables quantitatives est majoritairement inférieure à la moyenne. A l'exception des variables 'WRank' et 'LRank' comme nous avons pu constater au cours de l'analyse de l'étendue des valeurs minimums et maximums. Cependant, l'écart-type n'est pas excessif à côté de la moyenne concernant les rangs des joueurs gagnants et perdants.

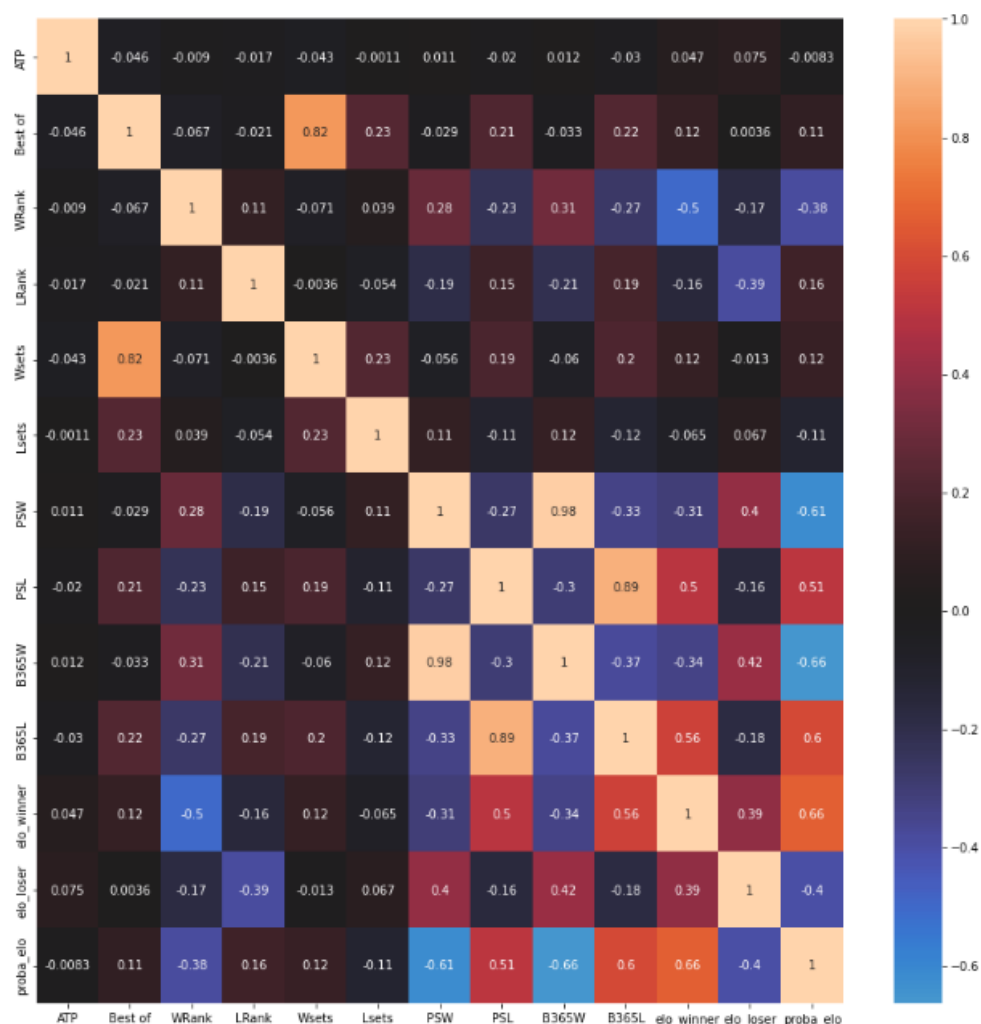
On peut donc conclure que les valeurs numériques ne sont pas très dispersées autour de la moyenne et qu'il n'existe pas de valeurs extrêmes constatées. Donc nous pouvons dire que la base de données est homogène. Concernant les variables 'WRank' et 'LRank', nous allons regrouper les modalités des données ensemble que l'on nommera 'Rank' afin d'avoir une seule colonne de rang pour l'ensemble des joueurs.

e) Corrélation entre les variables

L'étude des relations des variables quantitatives s'effectuent par l'étude des corrélations. On cherche à comprendre s'il existe une dépendance entre les variables et de déterminer le niveau de mesure.

Dans le cadre des valeurs numériques, nous allons utiliser dans un premier temps la heatmap du package seaborn. C'est une visualisation qui croisent l'ensemble des variables quantitatives à l'aide de l'échelle de couleur qui se trouve sur la droite dans lequel nous pouvons distinguer les niveaux de corrélations.

Analyse des corrélations des variables quantitatives



La matrice de confusion du package seaborn démontre le lien entre les différentes variables numériques. On observe pour les variables suivantes qu'il existe une très forte corrélation:

- 'Best of': le nombre maximum de sets joué dans un match & 'Wsets': le nombre de sets joué par un joueur gagnant ont une corrélation de 0,82. La relation entre le 'Best of' et 'Wsets' est liée au jeu du tennis, ce qui explique la mesure élevée.
- 'B365W': la probabilité du joueur gagnant de 'Bet365' & 'PSW': la probabilité du joueur gagnant de 'Pinnacle Sport' ont une corrélation de 0,98. La relation des probabilités des bookmakers est très forte concernant les joueurs gagnants. Cela signifie que les bookmakers estiment les mêmes joueurs gagnants.
- 'B365L' & 'PSL': les probabilité des joueurs perdants des bookmakers ont une corrélation de 0,89. La relation des probabilités des joueurs perdants est légèrement différente de celle des joueurs gagnants mais elle reste significative et les bookmakers estiment à peu près les mêmes joueurs perdants un match.
- 'elo_winner': le niveau de jeu des joueurs gagnants & 'proba_elo': la probabilité de gagner un match ont une corrélation de 0,66. La corrélation entre les deux mesures est assez bonne même si elle est tout de même éloignée de 1. Elle est aussi cohérente car la probabilité elo dépend du niveau de jeu du joueur gagnant.

Une autre manière d'étudier les relations est d'effectuer des tests statistiques, c'est-à-dire des tests d'indépendance des variables à l'aide du test de corrélation de Pearsons. A travers ce type de test, on va

chercher à comprendre, s'il existe un lien entre deux variables quantitatives. On fixe une hypothèse nulle pour le test de Pearson qui est la suivante, H_0 : "Les deux variables testées sont indépendantes".

Pour déterminer si l'on accepte ou on rejette l'hypothèse nulle, il faut regarder la p-value. Si celle-ci est inférieure à un seuil de 5% alors on rejette l'hypothèse nulle c'est-à-dire que l'on rejette que les deux variables testées sont indépendantes, inversement si le seuil est supérieur à 5%. Donc les variables testées sont bien dépendantes de l'une et de l'autre avec une mesure qui est le coefficient de corrélation de Pearson. Elle indique le sens de relation qui peut être positive ou négative et si la corrélation est nulle alors on dit que les variables sont décorréliées.

Ci-dessous, un récapitulatif des tests de Pearsons testés sur les variables mises en évidence par la matrice de confusion. Globalement, le test de Pearson confirme l'existence de corrélations entre les variables.

Test de Pearsons

Test de Pearsons	Coefficient	p_value	Interprétation
B365W - PSW	0,804774	0	Le coefficient est proche de 1 et la p-value < 5% => Corrélation entre B365W & PSW
B365L - PSL	0,820491	0	Le coefficient est proche de 1 et la p-value < 5% => Corrélation entre B365L & PSL
elo_winner - proba_elo	0,662299	0	Le coefficient assez proche de 1 et la p_value < 5% => Corrélation entre elo_winner & proba_elo
elo_loser - proba_elo	-0,401453	0	Le coefficient est de (-0,40) plus ou moins proche de -1 et la p_value < 5% => Corrélation négative entre 'elo_loser' et 'proba_elo'

Donc l'ensemble des variables quantitatives qui ont des corrélations significatives sont le 'Best of' & 'Wsets', 'B365W' & 'PSW', 'B365L' & 'PSL' et 'elo_winner' & 'proba_elo'.

Les analyses démontrent que les sets des joueurs ont une importance pour déterminer les joueurs gagnants. Mais dans le cadre de notre modélisation, nous allons exclure les sets des joueurs afin de ne pas biaiser nos résultats.

Ensuite, les probabilités des joueurs gagnants estimés par les bookmakers sont proches et ils vont nous servir de comparaison avec nos estimations.

Enfin, le niveau de jeu est une variable explicative très importante dans le cadre de notre modélisation pour estimer les joueurs gagnants.

1.2- Variables qualitatives

a) Les modalités les plus fréquentes

Concernant l'étude des variables qualitatives, commençons par observer les modalités des variables qui sont les plus fréquentes afin d'obtenir un premier aperçu.

Les modalités des variables qualitatives les plus fréquentes

	count	unique	top	freq
Location	44708	115	Paris	2784
Tournament	44708	207	Australian Open	2159
Date	44708	4104	2000-01-17	127
Series	44708	8	International	10792
Court	44708	2	Outdoor	36532
Surface	44708	4	Hard	23799
Round	44708	8	1st Round	20728
Winner	44708	899	Federer R.	970
Loser	44708	1400	Lopez F.	369
Comment	44708	4	Completed	43015

Nous observons que pour l'ensemble des variables qualitatives, il n'existe pas de valeurs manquantes. Ensuite, on constate que la variable 'Date' est considérée comme qualitative. Il va falloir la convertir en format date afin de la rendre exploitable dans le but d'ordonner la base de données. Il y a beaucoup de modalités de données pour les variables 'Location', 'Tournament', 'Date', 'Winner', 'Loser' et 'Comment'. Au niveau de la colonne des 'top', on peut voir les modalités de données les plus importantes pour chacune des variables.

b) Les modalités des variables qualitatives

Nous avons également réalisé une boucle nous permettant d'obtenir la fréquence des modalités pour l'ensemble des variables qualitatives, ci-dessous un exemple de réponse de la boucle:

Les modalités des variables qualitatives

```
=====
colonne= 4 Series
=====
International      10792
ATP250             9550
Grand Slam         8255
Masters            4616
Masters 1000       4528
ATP500             3433
International Gold  3294
Masters Cup        240
Name: Series, dtype: int64
nombre de modalités: 8
nombre de valeur nulle: 0
=====
colonne= 5 Court
=====
Outdoor      36532
Indoor       8176
Name: Court, dtype: int64
nombre de modalités: 2
nombre de valeur nulle: 0
=====
colonne= 6 Surface
=====
Hard      23799
Clay      14470
Grass     4916
Carpet    1523
```

A la différence du tableau "**Les modalités des variables qualitatives les plus fréquentes**", cela nous permet de nous rendre compte du détail de l'intitulé des modalités et des fréquences pour l'ensemble des variables qualitatives.

c) La table de contingence

Nous utilisons également la table de contingence afin de croiser deux variables qualitatives ou plus pour se rendre compte de comment les modalités de données se répartissent entre les différentes variables et en déduire des relations entre les variables.

Nous avons effectué le croisement des variables ‘Series’, ‘Location’ et ‘Tournament’ car les modalités des données pour les deux dernières variables sont extrêmement volumineuses. Nous nous posons en plus la question de comment nous allons gérer les informations de ces variables.

Table de contingence ‘Series’, ‘Location’ et ‘Tournament’

Series	Location	Tournament	
ATP250	's-Hertogenbosch	Ordina Open	31
		Ricoh Open	27
		Topshelf Open	85
		Unicef Open	89
	Antalya	Antalya Open	27
...
Masters 1000	Toronto	Rogers Masters	157
Masters Cup	Houston	Masters Cup	30
	Lisbon	Masters Cup	15
	London	Masters Cup	120
	Shanghai	Masters Cup	75

Nous constatons que la variable ‘Series’ regroupe différents tournois de la variables ‘Tournament’ et chaque tournoi a lieu dans des endroits différents.

Un autre exemple de table de contingence, le croisement entre la variable 'Comment' & 'Round' qui nous a permis de se rendre compte qu’il y a un joueur qui a été disqualifié au cours d’un match.

Table de contingence ‘Comment’ & ‘Round’

Comment	Round	
Completed	1st Round	19920
	2nd Round	11860
	3rd Round	3007
	4th Round	720
	Quarterfinals	4070
	Round Robin	308
	Semifinals	2077
	The Final	1053
Disqualified	The Final	1
Retired	1st Round	804
	2nd Round	369
	3rd Round	85
	4th Round	26
	Quarterfinals	115
	Round Robin	4
	Semifinals	55
	The Final	18
Walkover	1st Round	4
	2nd Round	91
	3rd Round	36
	4th Round	6
	Quarterfinals	55
	Semifinals	20
	The Final	4

Figure 13: Table de contingence ‘Court’ & ‘Surface’

Court	Surface	
Indoor	Carpet	1523
	Clay	162
	Hard	6491
Outdoor	Clay	14308
	Grass	4916
	Hard	17308

Dans le croisement des variables 'Court' et 'Surface', nous remarquons que peu importe le type de tournoi intérieur ('Indoor') ou extérieur ('Outdoor'), la surface la plus représentée est le dur ('hard'). Logiquement, il n'y a pas de tournoi sur herbe ('grass') en intérieur ni de tournoi sur moquette ('carpet') en extérieur. Les tournois sur terre ('clay') sont plus représentés en extérieur qu'en intérieur.

Les tables de contingences nous aident à comprendre certaines relations que nous pouvons déterminer en croisant des variables qualitatives mais le choix des variables pour l'analyse est complètement arbitraire.

d) Corrélation entre les variables qualitatives

Tout comme les variables quantitatives, les variables qualitatives ont un test d'indépendance que l'on appelle le test de chi-2 et dont l'hypothèse nulle est la suivante H_0 "les deux variables sont indépendantes". La statistique du test renvoie la p-value, le degré de liberté, la liste des fréquences. On rejette l'hypothèse nulle lorsque la p-value est inférieure à 5%. Pour mesurer le niveau de corrélation entre les variables, on utilise le V de Cramer qui renvoie un résultat entre 0 et 1.

Test de chi 2 et de V de Cramer

	Test chi 2 & Test V de Cramer	Coefficient du V de Cramer	p_value du chi 2	Interprétation
Winner	Court	0,23	<5%	Corrélation même si le V_Cramer n'est pas élevé
	Surface	0,29	<5%	Corrélation même si le V_Cramer n'est pas élevé
	Round	0,1	<5%	Corrélation même si le V_Cramer n'est pas élevé
	Comment	0	>5%	Aucune corrélation
Loser	Court	0,16	<5%	Corrélation même si le V_Cramer n'est pas élevé
	Surface	0,22	<5%	Corrélation même si le V_Cramer n'est pas élevé
	Round	0,09	<5%	Corrélation même si le V_Cramer n'est pas élevé
	Comment	0	>5%	Aucune corrélation

Nous avons effectué les tests statistiques pour les joueurs gagnants et perdants sur les variables qualitatives 'Court': le type de terrain, 'Surface': le type de surface, 'Round': le type de match et 'Comment': le commentaire du match.

Nous constatons que les résultats de corrélation du V de Cramer pour les deux types de joueurs sur les variables testées ne sont pas très proches de 1. Nous en déduisons que la corrélation entre ces variables et les joueurs est faible. Nous observons particulièrement pour la variable 'Comment' que la corrélation du V de Cramer est nulle pour les joueurs gagnants et perdants. Donc les commentaires sont complètement décorrélés

des joueurs. Nous remarquons que même si les coefficients de V de Cramer ne sont pas très significatifs, les corrélations des joueurs gagnants sont supérieures à celles des joueurs perdants. Cependant, le test de chi 2 démontre que les seuils de la p-value pour l'ensemble des variables testés sur les joueurs sont inférieurs à 5 %, sauf pour les commentaires des joueurs. Donc on rejette l'hypothèse nulle que les deux variables testées sont indépendantes. En conclusion le type de terrain, le type de surface et le type de match sont dépendants des joueurs gagnants et perdants. Les commentaires n'ont pas d'impact sur les différents types de joueurs.

L'étude des variables qualitatives, nous a permis d'observer les variables les plus fréquentes dans la base de données. Nous avons également observé de manière arbitraire des tables de contingences pour constater comment les modalités des données se répartissent entre des variables qualitatives. Enfin nous avons effectué des tests statistiques de dépendances adéquats pour confirmer la corrélation entre les variables qualitatives des joueurs.

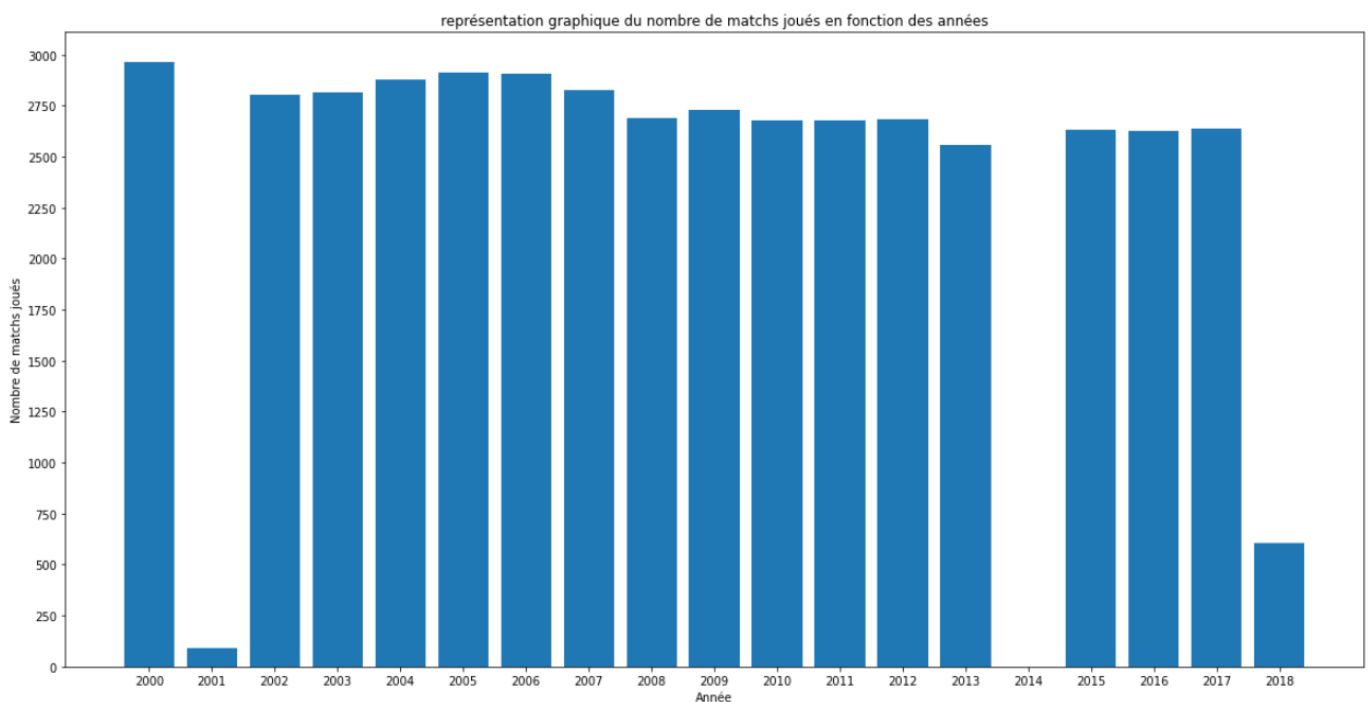
L'analyse des variables qualitatives tout comme les variables quantitatives sont des étapes importantes pour obtenir un aperçu de compréhension de notre base de données. Mais l'analyse des données catégorielles n'est pas évidente, c'est pourquoi nous allons approfondir nos analyses à l'aide des variables numériques par le biais de représentations graphiques. La visualisation des données va nous permettre de comprendre le lien entre les deux types de variables mais également d'observer le comportement des modalités des données.

2- Visualisation des données

Dans cette phase, nous avons réalisé une analyse des variables contenues dans le dataset permettant ainsi de faire ressortir des tendances, telles que la corrélation entre certaines valeurs numériques, ou la visualisation de fréquence en fonction d'une variable donnée (ex nombre de matchs gagnés en fonction du nom du joueur).

Une étape préalable à la datavisualisation consiste à faire apparaître une colonne 'year' pour faciliter l'exploitation des données. Ainsi, nous pouvons réaliser une estimation du nombre de matchs joués par an.

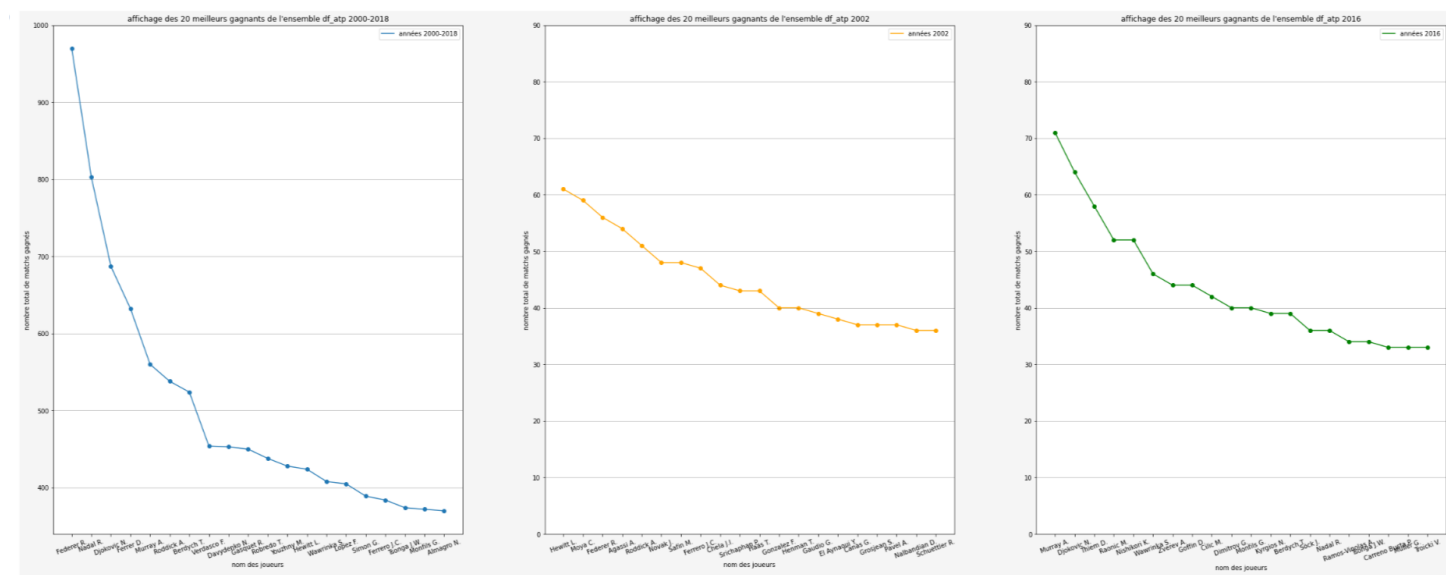
a) Nombre de matchs joués par an



Représentation graphique du nombre de matchs total joués par an

Cette représentation permet d'afficher une homogénéité du nombre total de matchs joués par an: en moyenne 2736 matchs en excluant les données aberrantes comme 2001 et 2018. Cette représentation permet également de mettre en évidence qu'il n'existe aucune donnée concernant l'année 2014 dans ce dataset.

b) Top 20 des meilleurs gagnants

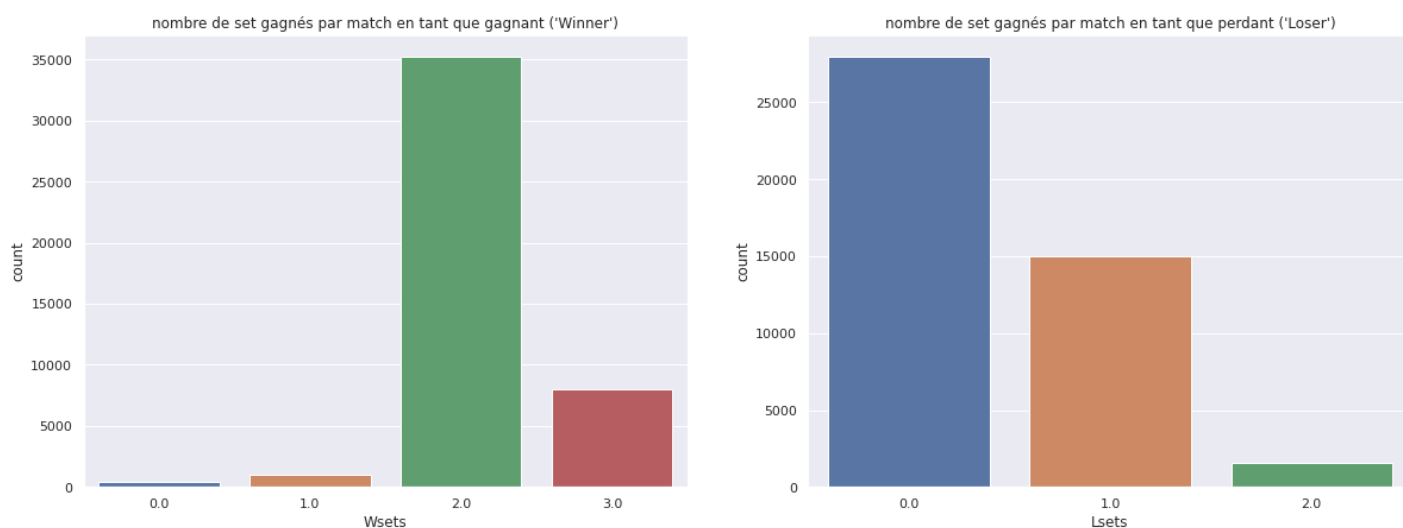


Top 20 des meilleurs joueurs gagnants (visible dans le notebook): à gauche top20 sur 2000-2018, milieu: top20 sur 2002, à droite: top20 sur 2016)

Dans ce graphe, nous constatons que les 8 meilleurs gagnants entre 2000 et 2018 ont gagné plus de 450 matchs. Ensuite sur ces 3 graphiques des top20 nous pouvons observer quelques différences:

- Sur l'ensemble du dataset (graphe de gauche), les 7 premiers joueurs se détachent avec plus de 500 matchs gagnés entre 2000 et 2018. Les 13 suivants se situant entre 380 et 450 matchs gagnés.
- La comparaison entre top 20 de 2002 (milieu) et top20 de 2016 (droite) nous montre une croissance du nombre de matches gagnés en 2016 pour les 2 premiers joueurs (Murray et Djokovic) et des écarts plus grands entre les 6 premiers joueurs de 2016 vs 2002 (env 26 matches d'écart en 2016 vs env 13 pour 2002).

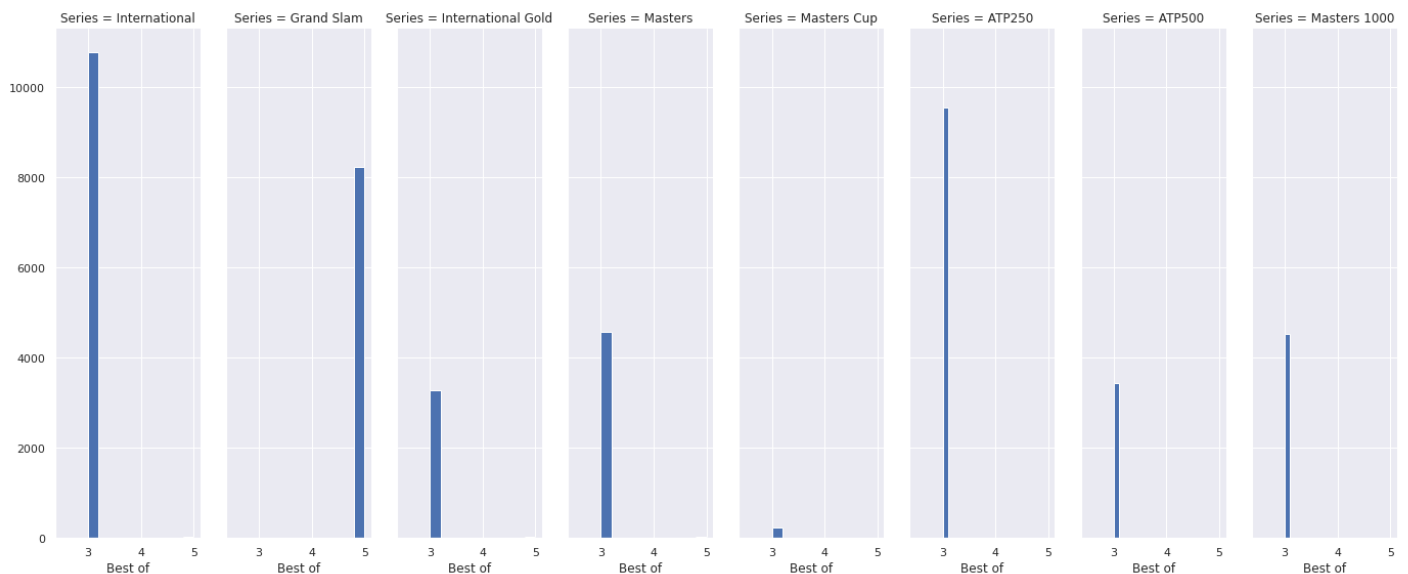
c) Nombre de set gagnés par match en tant que gagnant /perdant



histogramme représentant le nombre de sets gagnés gagnant/perdant

La représentation de cet histogramme nous permet de constater que la majorité des matchs joués et reportés dans ce dataset sont gagnés en 2 sets (3 sets gagnés pour les matchs des tournois de Grand Chelem). Quant aux perdants, ceux-ci ne gagnent aucun set ou 1 set dans presque 45% des matchs.

Pour mieux comprendre cette répartition, nous pouvons regarder le nombre de sets maximum gagnés par match en fonction du type de tournoi.



Représentation du nombre de set max en fonction de la catégorie de tournoi

Cette représentation nous permet de comprendre que tous les tournois se jouent en maximum 3 sets (dont 2 gagnants), excepté les tournois Grand Chelem (Grand Slam) en maximum 5 sets (3 sets gagnants). Nous remarquons également grâce à l'échelle des ordonnées partagées que le nombre de matches joués en 3 sets gagnants pour les tournois internationaux et pour les tournois ATP250 sont nettement supérieurs au nombre de matches en 3 sets gagnants pour les autres séries disputées. Ceci est dû au fait que les tournois type Masters (Cup, 1000) ainsi que ATP500 sont moins nombreux et plus prestigieux (plus de points ATP remportés et plus grande dote).

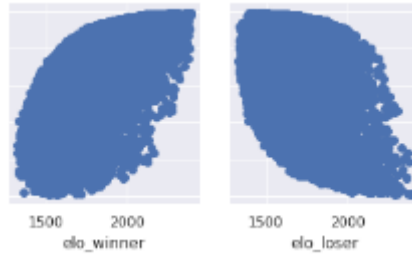
reference https://fr.wikipedia.org/wiki/Cat%C3%A9gorisation_des_tournois_de_tennis#Depuis_1990)

d) Elo_winner, elo_loser, proba_elo

Nous nous sommes intéressés à l'ensemble des variables en faisant une analyse de celle-ci 2 par 2 (utilisation de la fonction PairGrid de seaborn). Cette représentation est disponible en totalité dans le notebook colab et n'a pas été reportée intégralement pour faciliter la lecture.

Nous observons une symétrie entre la représentation du elo_winner en fonction de la proba_elo et la représentation du elo_loser en fonction de la proba_elo.

Nous constatons également une représentation assez linéaire de la variable B365W en fonction de PSW (il s'agit des côtes estimées pour les gagnants par nos 2 Bookmakers), montrant ainsi une homogénéité des cotes définies par nos 2 bookmakers du dataset.

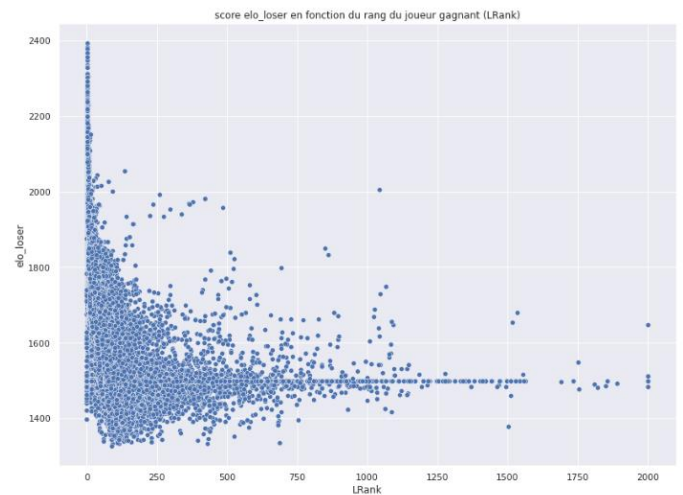
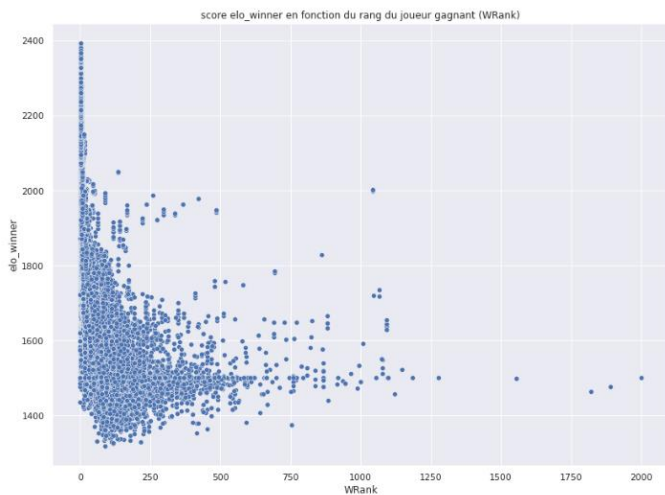


proba_elo en fonction de elo_winner(gauche) et elo_loser(droite)



PSW (côte pinacle du gagnant) en fonction de B365W(cote B365 du gagnant)

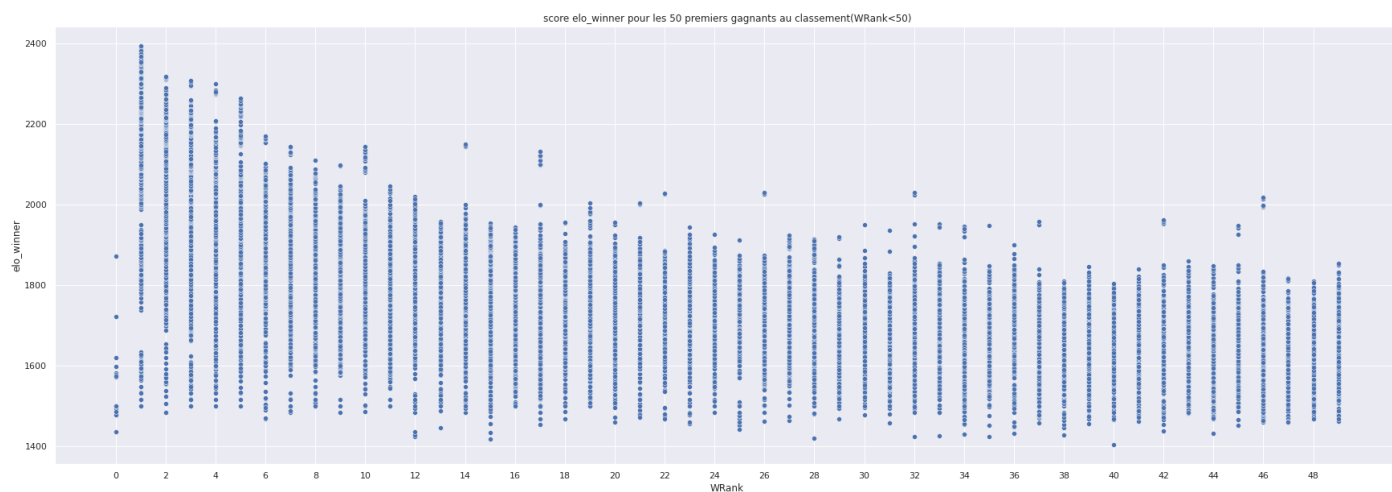
Faisons la représentation graphique de la relation entre le score elo du gagnant/perdant (elo_winner/elo_loser) en fonction du rang du joueur (WRank/LRank):



Représentation du score elo_winner (gauche) et elo_loser(droite) en fonction de leur classement

Ce graphique nous permet de constater une répartition très similaire du score elo indépendamment du résultat du match (les graphes winner et loser étant similaires entre les les rangs 0 et 500). Cependant, nous constatons également sur le graphique de droite, qu'une grande partie des perdants a un score elo de 1500 correspondant au classement provisoire qui évoluera durant la saison et la carrière du joueur. Nous observons également que les premiers joueurs au classement sont ceux ayant un score elo plus élevé.

En zoomant sur les 50 premiers joueurs au classement (WRank<50), nous observons que le score elo évolue de façon verticale pour une même position dans le classement. Ceci s'explique par l'évolution du score elo d'un même joueur durant la saison mais aussi au fait qu'un rang est pris par une multitude de joueurs dans notre dataset:



évolution du score elo pour les 50 meilleurs premiers joueurs du dataset

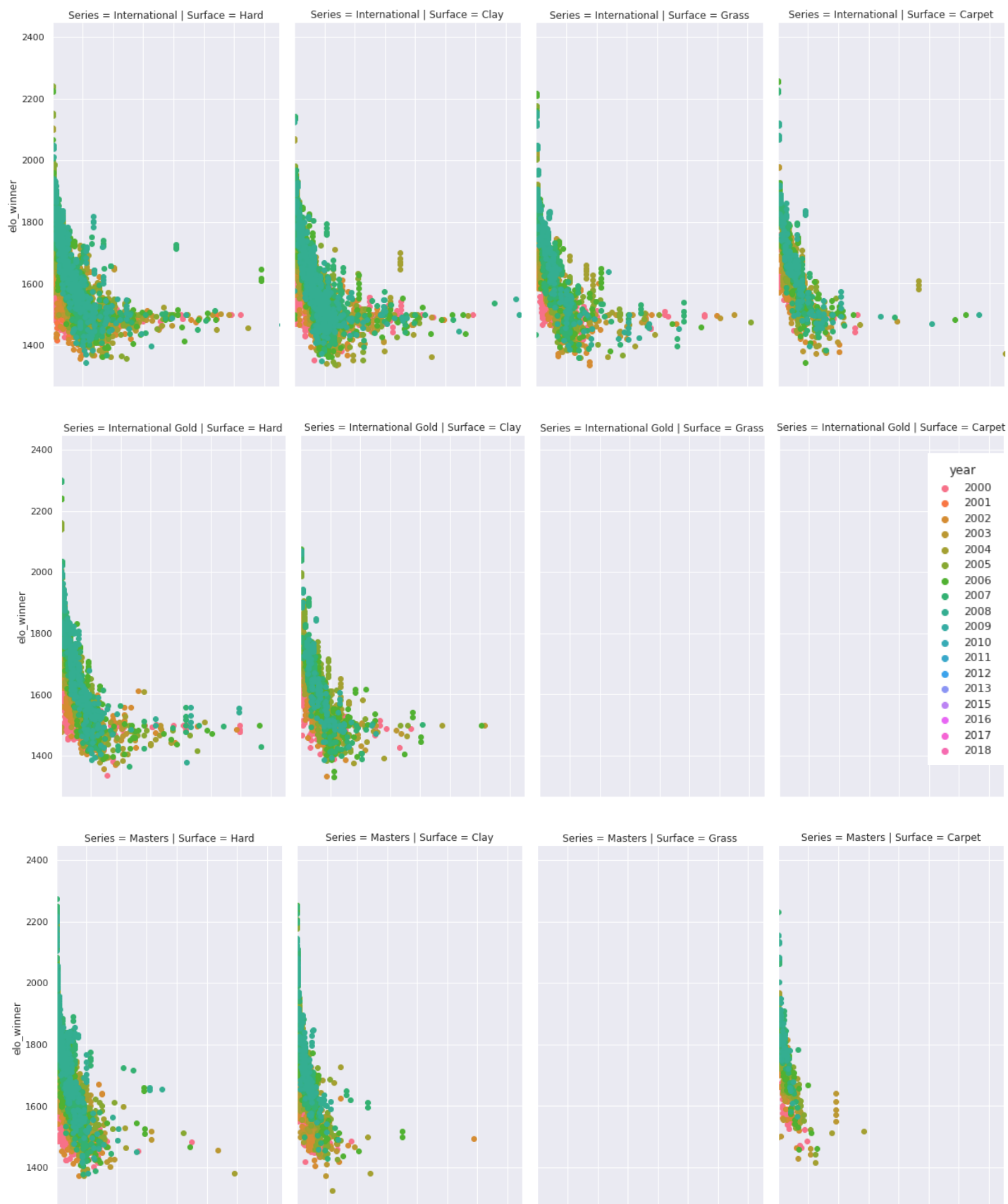
Ce graphe nous permet également de constater que certains joueurs occupent un rang égal à 0. En appliquant la fonction DataFrameInfo sur ces joueurs, nous observons qu'ils apparaissent en majorité pour la variable Round ayant une modalité égale à "1st round" correspondant aux premiers tours de tournois. Il s'agit donc très certainement de 'nouveaux' joueurs arrivant dans le circuit ayant gagné.

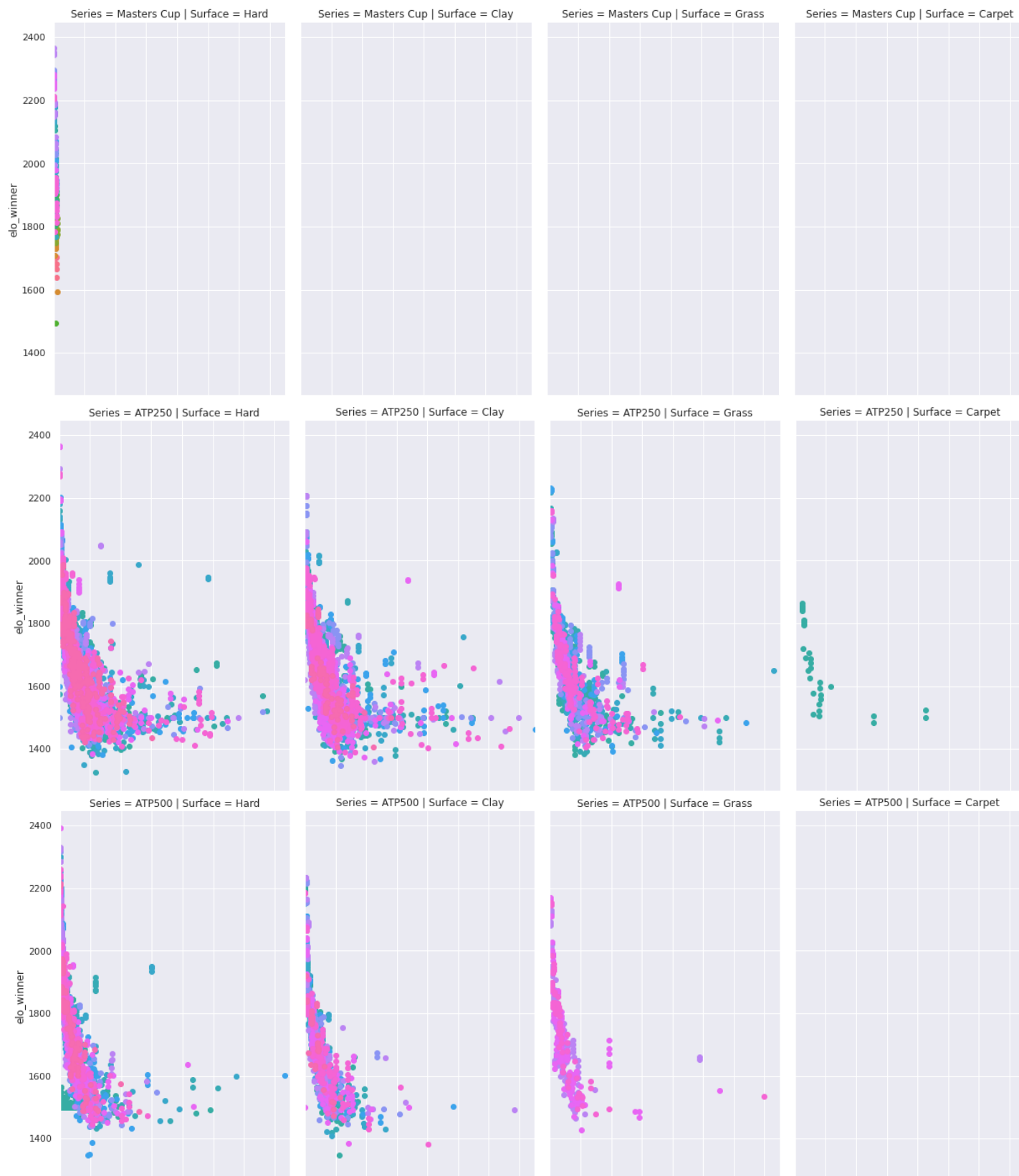
Considérant les N°1 au classement, la table affichée ci-dessous (table de contingence du elo_winner min et max en fonction des N°1 au classement et de l'année) nous comprenons que le score elo évolue également entre 2000 et 2018: André Agassi, numéro 1 en 2000 obtient un score elo entre 1500 et 1633; en 2005, Roger Federer obtient un classement elo entre 2142 et 2251; en 2016 Novak Djokovic obtient le meilleur score elo du dataset avec un mini à 2298 et un max à 2392.

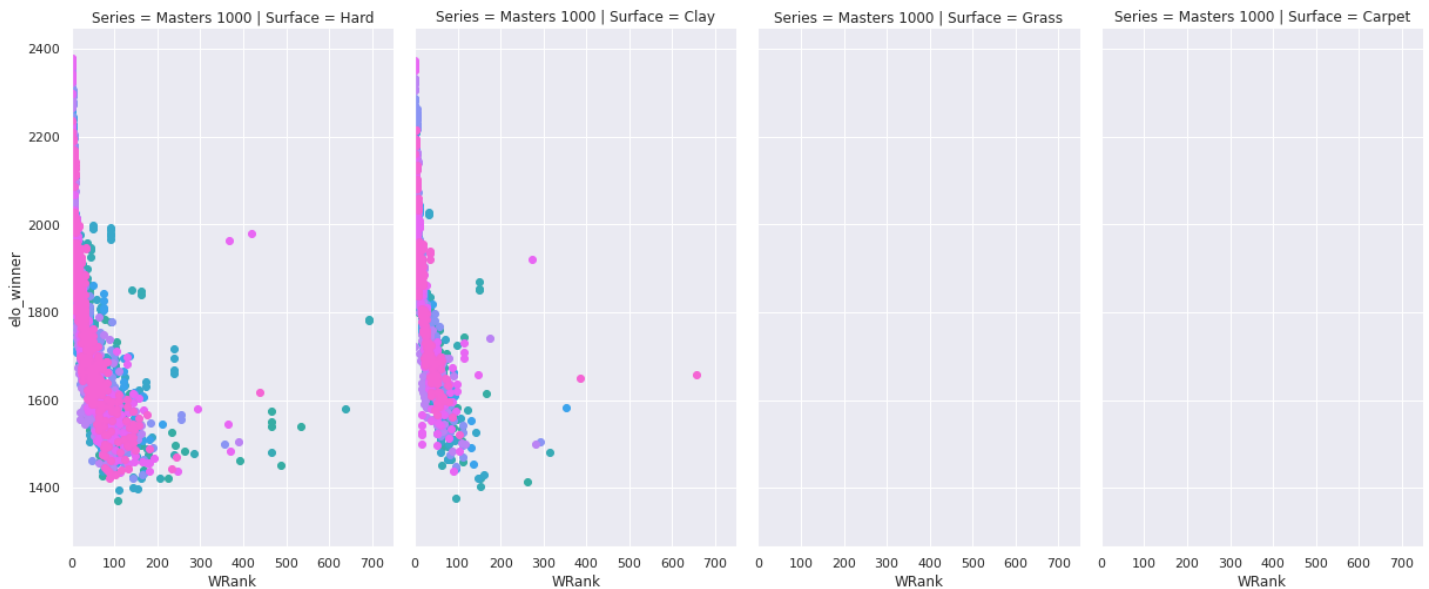
		elo_winner	
		min	max
year	Winner		
2000	Agassi A.	1500.000000	1633.681747
	Safin M.	1811.959224	1843.680886
2002	Hewitt L.	1737.639319	1868.804534
2003	Agassi A.	1858.827616	1898.839795
	Ferrero J.	1500.000000	1500.000000
	Ferrero J.C.	1866.488449	1908.373246
	Hewitt L.	1853.908751	1917.040954
	Roddick A.	1937.543655	1950.040628
2004	Federer R.	1987.564078	2137.809126
	Roddick A.	1910.610572	1933.654008
2005	Federer R.	2141.924564	2250.505199
2006	Federer R.	2210.619864	2278.562274
2007	Federer R.	2176.692260	2302.068719
2008	Federer R.	2105.859733	2209.250456
	Nadal R.	2155.108952	2198.200824
2009	Federer R.	2151.540461	2200.544823
	Federer R.	1500.000000	1548.000000
	Nadal R.	2108.262629	2237.583382
2010	Federer R.	2058.759008	2164.567100
	Nadal R.	2120.345799	2175.442620
2011	Djokovic N.	2246.040999	2300.891210
	Nadal R.	2099.179604	2219.539433
2012	Djokovic N.	2193.033869	2278.330121
	Federer R.	2225.820429	2264.165154
2013	Djokovic N.	2204.325420	2334.229704
	Nadal R.	2284.527051	2331.323732
2015	Djokovic N.	2263.303351	2366.419668
2016	Djokovic N.	2298.216862	2392.408923
	Murray A.	2237.666168	2258.556429
2017	Murray A.	2066.891137	2277.738398
	Nadal R.	2084.626060	2150.667670
2018	Nadal R.	2089.162703	2096.683796

Affichage des scores elo min et max par joueurs premiers au classement et par année.

Confrontons ce score elo avec d'autres variables du dataset:



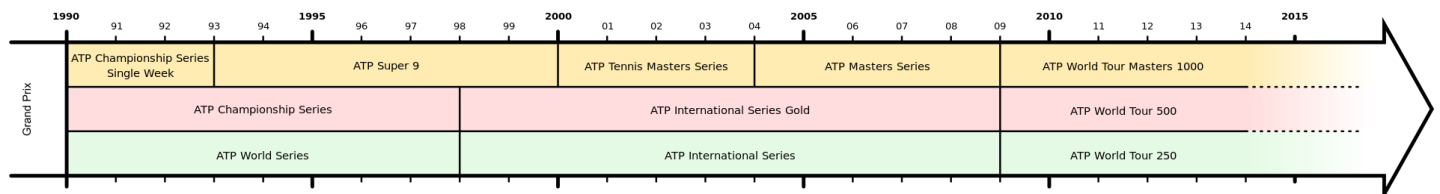




Représentation graphique du elo_winner en fonction du classement (WRank), du type de tournoi(Series) et de la surface

Grâce à la fonction facetGrid et à la représentation graphique ci dessus, nous constatons plusieurs points:

1. Certains types de tournoi sont ouverts aux meilleurs joueurs: par exemple Masters, Masters 1000, ATP 500, très peu de points figurent au-delà d'un WRanks à 200. De la même manière, le tournoi Masters Cup est ouvert uniquement à l'élite.
2. Certains types de surface sont spécifiques à un type de tournoi (par ex: la moquette ('Carpet') n'existe pas pour les tournois Grand Slam, international Gold ou encore Masters 1000 et ATP500. Les Masters Cup ne se jouent que sur surface "dure" ('Hard').
3. Les couleurs des années, nous permettent de constater qu'il existe une évolution de la représentation du type de surface en fonction des tournois. En effet, les Masters et Séries international sont davantage présents dans les années 2000 à 2009 (vert); les autres types de Series étant davantage représentés pour les années 2010 à 2018 (rose). Ceci nous permet de conclure que la dénomination des tournois (le type de Series) ainsi que les surfaces des terrains ont changé entre 2000 et 2018 (notamment fortement en 2009).
4. Nous constatons aussi que la moquette a disparu à partir de 2009 (aucun point rose sur les graphes), ce type de surface ayant été banni par l'ATP car trop exigeant (en entretien et pour les articulations des joueurs).



Evolution des types de tournois (référence Wikipédia)

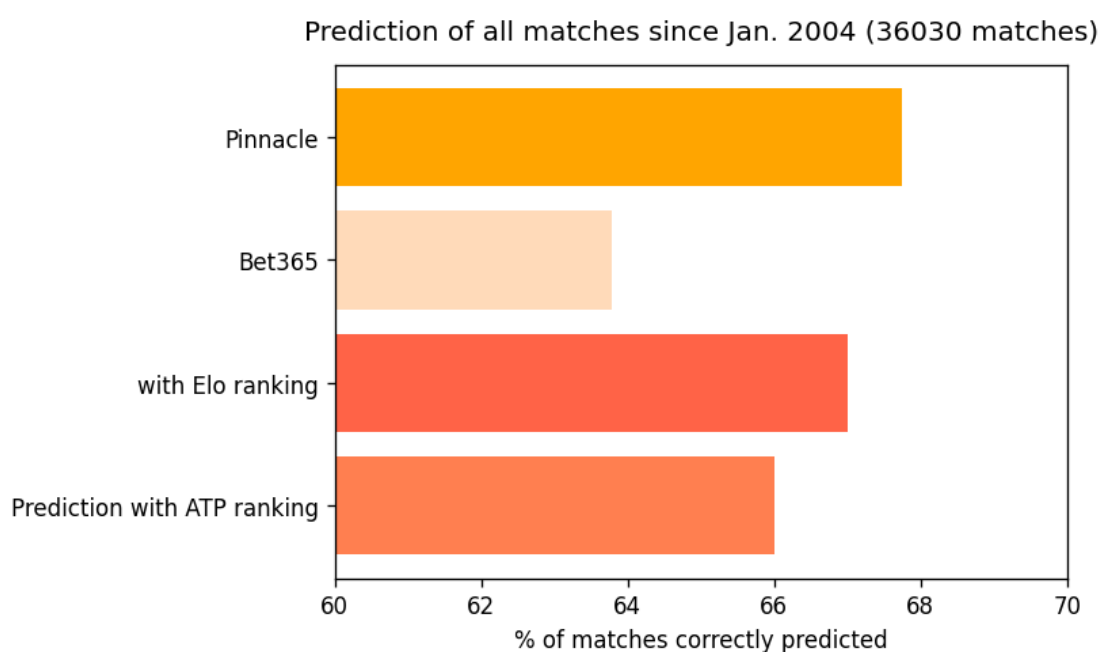
e) Performance des prédictions des matchs

Après avoir étudié notre dataset, notamment les différents types de tournoi et leur évolution, le fonctionnement des scores elo, nous pouvons nous intéresser aux données des bookmakers (côtes) qui traduisent les prédictions de ceux-ci quant à la victoire/défaite d'un joueur pour un match donné.

Nous pouvons prédire la victoire/défaite d'un joueur en fonction de son adversaire de différente façon en étudiant différentes variables:

- son rang au classement
- son score elo
- sa côte définie par les 2 bookmakers

l'application de la fonction développée par edouard Thomas (source : [Beat the bookmaker edouard thomas](#)), nous permet de visualiser le pourcentage de bonnes prédictions (c'est-à dire qu'un joueur gagne si son rang, son classement ou sa côte son meilleur que son adversaire) selon les variables citées précédemment:



Evolution des types de tournois (référence Wikipédia)

Selon le graphe, nous constatons que le bookmaker Pinnacle prédit mieux que son concurrent Bet365. Nous constatons également que le score elo montre de bonnes performances en comparaison au classement. Le but de cet histogramme est de comparer nos futures prédictions avec ceux des bookmakers et de se rendre compte, si nous devons améliorer nos futurs algorithmes.

Étape 2: Modélisation statistique

Après avoir bien pris connaissance de la base de données 'df_atp' à l'aide de tests statistiques et de nombreuses visualisations des données. Nous pouvons passer à l'étape de préparation de la base de données pour l'entraînement de notre modélisation statistique.

1- Préparation des données

a) Encodage des variables catégorielles

Nous avons choisi d'effectuer l'encodage des données catégorielles afin de faciliter l'entraînement de notre modélisation. Cette dernière ne peut s'effectuer que si les données sont numériques, c'est pourquoi nous avons choisi de traiter de manière arbitraire les modalités des variables catégorielles. Par défaut, toutes les variables catégorielles ordinales que l'on peut tout simplement ordonner sont remplacées par des nombres. Mais pour les variables 'Location' et 'Tournament', nous avons choisi d'effectuer la dichotomisation c'est-à-dire de les transformer en variable indicatrice afin qu'elles puissent être interprétées au cours de la modélisation.

Encodage des variables catégorielles

Variables	Modalités	Nouvelles Modalités
Date	Année + Mois	Une colonne Année et une colonne Mois
Surface	Carpet, Clay, Grass, Hard	0, 1, 2, 3
Comment	Completed, Disqualified, Retired, Walkover	1, 0, -1, -2
Court	Indoor, Outdoor	0, 1
Series	International Gold, International, Grand Slam, Masters Cup, Masters, Masters 1000, ATP500, ATP250	3500, 3000, 2500, 2000, 1500, 1000, 500, 250
Location	liste de lieux	dichotomisation
Tournament	liste de tournois	dichotomisation

b) Retraitement de la variable 'Winner' et 'Loser'

Après l'encodage, nous avons dû créer une nouvelle variable 'Player cible' à partir des variables catégorielles 'Winner' et 'Loser'. Par défaut, tous les joueurs 'Winner' valent 1 dans 'Player cible' puis 0 pour les joueurs perdants. Cette nouvelle variable constitue notre cible ou target dans notre modélisation statistique. Notre objectif est de prédire les joueurs gagnants et perdants, c'est pourquoi, nous avons déterminé une donnée numérique pour les joueurs gagnants et perdants. Dans le même temps, nous avons également construit un identifiant unique pour chaque joueur.

Donc au départ, les informations des joueurs gagnants et perdants étaient sur une même ligne et après retraitement des données nous retrouvons deux lignes d'information dont une pour le joueur gagnant et une autre pour le joueur perdant.

c) Autres retraitements

Nous avons constaté au cours de notre audit de données que certaines variables avaient des valeurs manquantes dont les probabilités des bookmakers et les sets des joueurs gagnants et perdants. Nous avons choisi de remplacer les valeurs manquantes des bookmakers par une valeur nulle. Enfin pour les sets des joueurs de les supprimer de notre base d'entraînement car nous ne sommes pas censés connaître le résultat final d'un match. Cela risque de biaiser les résultats de notre modélisation statistique.

Bien évidemment, toutes les variables d'encodage seront également supprimées de notre base d'entraînement. Mais avant de toutes les supprimer, nous ordonnons à l'aide de la variable 'Date' l'ensemble de notre DataFrame afin de partir sur une modélisation ordonnée pour l'entraînement.

d) Base de modélisation

Vous trouverez dans le notebook deux dataFrames qui vont être utilisés à la modélisation dont le 'df_atp_final' qui a subi tout le processus de préparation des données. Ensuite, le df_atp_moyenne_mobile dans lequel les données ont eu le même processus de préparation des données et en plus un lissage sur les données numériques en fonction du temps.

2- Modélisation de classification

Le but de notre modélisation est de créer un algorithme permettant de réaliser une prédiction du joueur gagnant un match supérieure aux prédictions des bookmakers. La variable cible étant 1 (gagnant) ou 0 (perdant), le type de modélisation à appliquer est l'apprentissage supervisé par classification.

a) modélisation: utilisation des classifieurs sans hyperparamètre

- modélisation 1: df_atp_final

Dans un premier temps, les classifieurs sélectionnés pour notre modélisation sont:

1. LogisticRegression
2. KNeighbors
3. RandomForest

Les premières modélisations ont été réalisées à partir de notre dataset final sans moyenne mobile et avec nos classifieurs sans hyperparamètre.

Nous avons choisi une base d'entraînement de notre modèle correspondant à 20% de la taille totale du dataset et avons sélectionné cette base d'entraînement selon un ordre chronologique (attribut du train_test_split shuffle=False).

L'entraînement et l'application de nos classifieurs nous donne des premières matrices de confusion et les scores de modèle suivant (1: le joueur gagnant, 0: le perdant):

'Matrice de confusion modele LogisticRegression:'

Classe prédite	0	1
Classe réelle		
0	0.224558	0.275442
1	0.123518	0.376482

'Matrice de confusion modele KNN:'

Classe prédite	0	1
Classe réelle		
0	0.356520	0.143480
1	0.320174	0.179826

'Matrice de confusion modele RandomForest:'

Classe prédite	0	1
Classe réelle		
0	0.422836	0.077164
1	0.141523	0.358477

'le score pour le modèle RegressionLogistic est:'
0.6010400357861776
'le score pour le modèle KNN est:'
0.536345336613733
'le score pour le modèle RandomForest est:'
0.781312905390293

Nous constatons suite à cette première modélisation que le classifieur RandomForest a un score moyen plus proche de 1 (0.78) que les 2 autres; par conséquent RandomForest semble plus performant pour ce type de modélisation par classification étant donné que nos 2 classes (gagnant/perdant) sont équilibrées.

- modélisation 2: standardisation avec MinMax scaler

Notre dataset final df_atp_final comportant des variables numériques de différentes échelles, nous décidons de réaliser une standardisation de nos données pour s'affranchir de cette différence entre les variables.

'Matrice de confusion modele LogisticRegression:'

Classe prédite	0	1
Classe réelle		
0	0.225565	0.274435
1	0.123742	0.376258

'Matrice de confusion modele KNN:'

Classe prédite	0	1
Classe réelle		
0	0.310445	0.189555
1	0.223608	0.276392

'Matrice de confusion modele RandomForest:'

Classe prédite	0	1
Classe réelle		
0	0.417860	0.082140
1	0.137721	0.362279

'le score pour le modèle RegressionLogistic est:'
0.6018228584209349
'le score pour le modèle KNN est:'
0.5868373965555804
'le score pour le modèle RandomForest est:'
0.780138671438157

Par comparaison avec notre première modélisation, nous obtenons les mêmes résultats, aussi bien au niveau des matrices de confusion que des scores moyens des modèles.

La standardisation n'influence pas la performance de nos modèles, nous décidons donc de ne pas poursuivre cette action pour les modélisations suivantes.

- Interprétation des modèles sans moyenne mobile et sans hyperparamètre

le rapport de classification pour LogisticRegression est:

	precision	recall	f1-score	support
0	0.65	0.45	0.53	8942
1	0.58	0.75	0.65	8942
accuracy			0.60	17884
macro avg	0.61	0.60	0.59	17884
weighted avg	0.61	0.60	0.59	17884

le rapport de classification pour KNN est:

	precision	recall	f1-score	support
0	0.53	0.71	0.61	8942
1	0.56	0.36	0.44	8942
accuracy			0.54	17884
macro avg	0.54	0.54	0.52	17884
weighted avg	0.54	0.54	0.52	17884

le rapport de classification pour RF est:

	precision	recall	f1-score	support
0	0.75	0.85	0.79	8942
1	0.82	0.72	0.77	8942
accuracy			0.78	17884
macro avg	0.79	0.78	0.78	17884
weighted avg	0.79	0.78	0.78	17884

le rapport de classification pour LogisticRegression avec standardisation MinMax est:

	precision	recall	f1-score	support
0	0.65	0.45	0.53	8942
1	0.58	0.75	0.65	8942
accuracy			0.60	17884
macro avg	0.61	0.60	0.59	17884
weighted avg	0.61	0.60	0.59	17884

le rapport de classification pour avec standardisation MinMax KNN est:

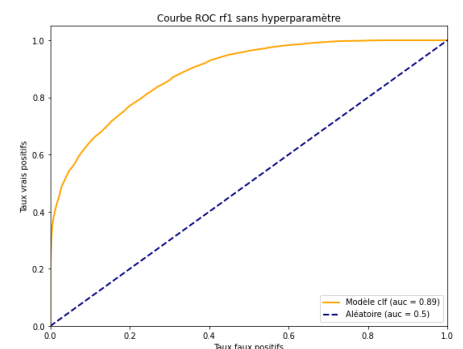
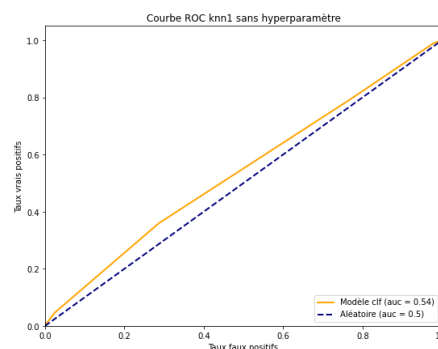
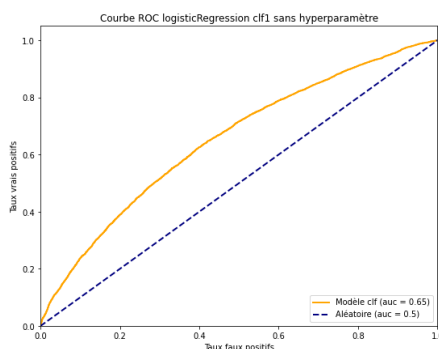
	precision	recall	f1-score	support
0	0.58	0.62	0.60	8942
1	0.59	0.55	0.57	8942
accuracy			0.59	17884
macro avg	0.59	0.59	0.59	17884
weighted avg	0.59	0.59	0.59	17884

le rapport de classification pour avec standardisation MinMax RF est:

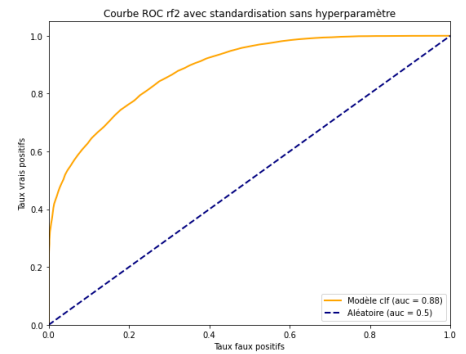
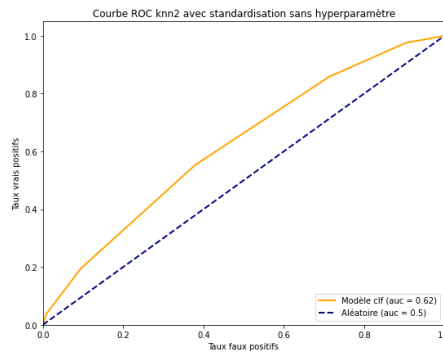
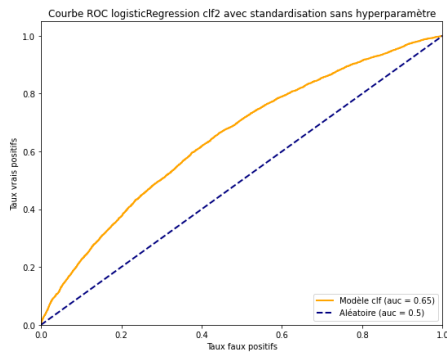
	precision	recall	f1-score	support
0	0.75	0.84	0.79	8942
1	0.82	0.72	0.77	8942
accuracy			0.78	17884
macro avg	0.78	0.78	0.78	17884
weighted avg	0.78	0.78	0.78	17884

Nous avons vu précédemment que le modèle RandomForest obtient un score moyen le plus performant. Dans notre cas concret, nous cherchons à prédire correctement les vainqueurs de match, par conséquent, nous cherchons un modèle ayant une mesure de précision performante. Celle-ci est de 0.82 pour RandomForest (avec et sans standardisation) s'agissant des gagnants (0.58 pour logisticRegression et 0.56/0.59 pour KNN).

Si nous regardons la courbe ROC et l'aire sous cette courbe (AUC), nous pouvons conclure que pour un match choisi au hasard, la capacité de notre modèle RandomForest à classifier correctement la victoire d'un joueur est bien supérieure aux autres modèles testés (0.88 et 0.89).



Courbes ROC et aire sous la courbe (AUC) modélisations sans hyperparamètre
 (à gauche: LogisticRegression, milieu: KNN, à droite: RandomForest)



Courbes ROC et aire sous la courbe (AUC) modélisations sans hyperparamètre avec standardisation
 (à gauche: LogisticRegression, milieu: KNN, à droite: RandomForest)

- conclusion de la modélisation sans hyperparamètre

Nous avons constaté que la standardisation MinMax n'améliore pas les performances du modèle LogisticRegression à l'inverse de celui des KNN. Le modèle le plus performant dans les 2 cas est celui de RandomForest. Nous choisissons donc de continuer avec ce modèle pour tenter d'améliorer ses performances.

b) Modélisation avec optimisation des paramètres: travail sur les hyperparamètres

Dans cette partie nous avons choisi de travailler sur 2 classifieurs pour travailler sur les hyperparamètres:

1. RandomForest (précédemment testé)
2. GradientBoosting

La recherche de paramètres optimaux est faite grâce à la fonction GridSearchCV.

Nous avons, dans un premier temps, défini le dictionnaire de paramètres à tester pour les 2 classifieurs, puis nous avons entraîné nos 2 modèles sur la base d'entraînement.

Choix des paramètres RandomForest	Choix des paramètres GradientBoosting
<ul style="list-style-type: none"> - n_estimators: 50,100,250,500 - min_sample_leaf: 1,3,5 - criterion: gini, entropy, log loss - max_features: sqrt, log2, none 	<ul style="list-style-type: none"> - n_estimators: 50,100,250,500 - loss: log_loss, deviance, exponential - subsample: 0.0,0.5,1.0 - criterion: friedman_mse, squared_error, mse - min_sample_leaf: 1,3,5 - max_features: sqrt, log2 None - learning_rate: 0.1, 1.0, 10, 100
Paramètres optimaux après entraînement	
<ul style="list-style-type: none"> - criterion: 'entropy', - max_features: None, - min_samples_leaf: 3, - n_estimators: 250 	<ul style="list-style-type: none"> - criterion:'friedman_mse', - learning_rate:0.1 - loss:'deviance' - max_features:None, - min_sample_leaf:3, - n_estimators:500 - subsample: 0.5

Cet établissement de paramètre nous a permis de générer les matrices de confusion et score moyen suivants:

```
'le score du model RandomForest:'
0.7861216730038023
'Matrice de confusion modele RandomForest:'
classe predite      0      1
classe reelle
0      0.413722  0.086278
1      0.127600  0.372400
'le score du model GradientBoosting:'
0.7781816148512637
'Matrice de confusion modele GradientBoosting:'
classe predite      0      1
classe reelle
0      0.421550  0.078450
1      0.143368  0.356632
```

Matrice de confusion modèle RandomForest et GradientBossting avec hyperparamètres

Les 2 matrices ainsi que les scores moyens des modèles sont similaires entre eux. En comparaison avec la modélisation sans hyperparamètre, nous obtenons également le même type de résultat. La fixation des hyperparamètres, améliore sensiblement mais reste dans le même ordre de grandeur de performance.

```
le rapport de classification pour RandomForest paramètres optimisés est:
precision    recall  f1-score   support

0      0.76    0.83    0.79     8942
1      0.81    0.74    0.78     8942

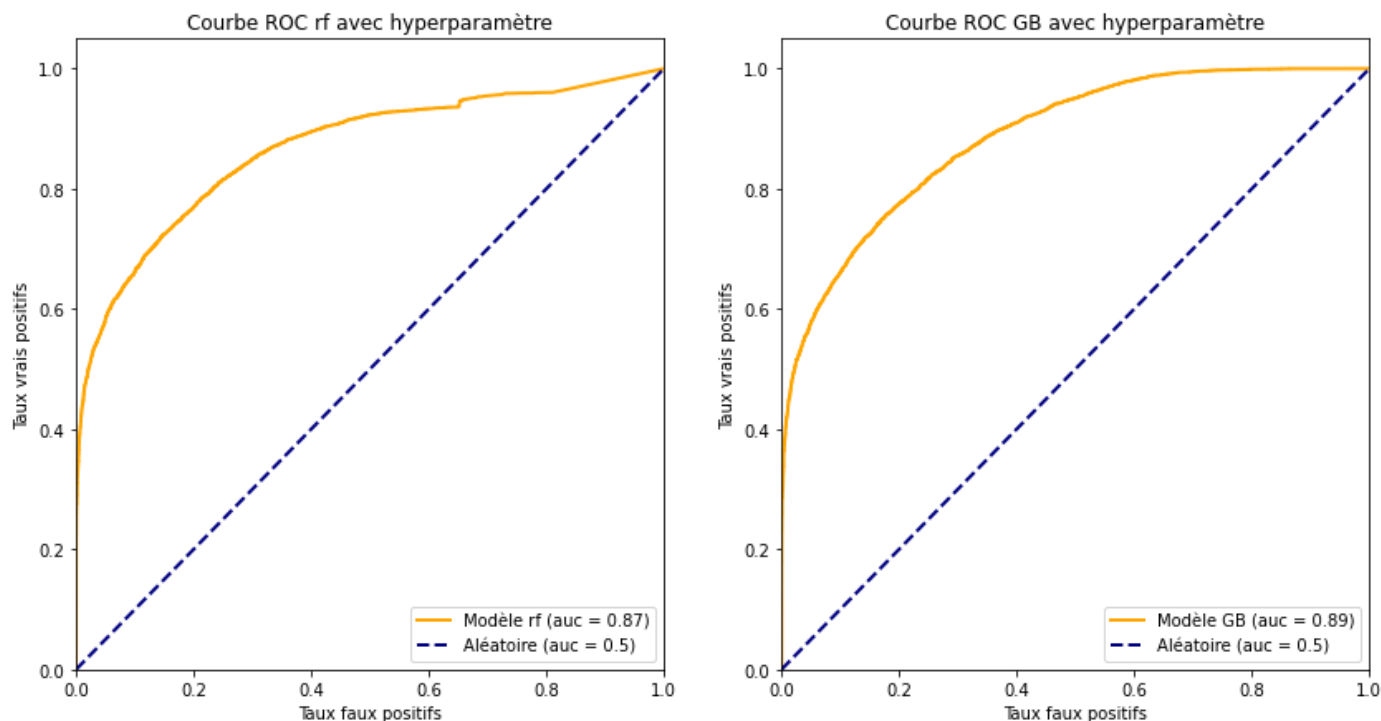
accuracy
macro avg    0.79    0.79    0.79     17884
weighted avg 0.79    0.79    0.79     17884

le rapport de classification pour GradientBoosting paramètres optimisés est:
precision    recall  f1-score   support

0      0.75    0.84    0.79     8942
1      0.82    0.71    0.76     8942

accuracy
macro avg    0.78    0.78    0.78     17884
weighted avg 0.78    0.78    0.78     17884
```

rapports de classification RandomForest et GradientBoosting avec hyperparamètres



Courbe ROC et AUC pour RandomForest et GradientBoosting

- Conclusion sur la modélisation avec paramètres optimisés:

Les 2 modèles avec paramètres optimisés (RandomForest et GradientBoosting) ont des performances similaires avec une très légère influence positive pour GradientBoosting.

Si nous nous intéressons à la précision de notre modèle, c'est-à-dire la capacité de notre modèle à catégoriser les vrais positifs parmi toutes les prédictions positives du modèle, nous constatons un score de 0.81 et 0.82 pour les gagnants. Ce que nous considérons comme un bon résultat.

Par ailleurs, nous constatons grâce à la courbe ROC et à l'aire sous celle-ci (AUC) que la capacité de nos modèles à catégoriser positivement un joueur (taux de vrais positifs) est bien supérieure au mode aléatoire. Par conséquent, notre modèle (RandomForest ou GradientBoosting) est performant dans un environnement de paris sportifs gagnant/perdant avec notre dataset `df_atp_final`.

Afin d'augmenter la performance de nos algorithmes, nous avons décidé de compléter notre dataset par les statistiques de joueurs via la modélisation avec moyenne roulante.

c) Modélisation avec la moyenne mobile

La moyenne mobile ou moyenne roulante ou moyenne glissante est une moyenne statistique qui est principalement utilisée sur une liste de série temporelle ordonnée afin de supprimer les fluctuations transitoires et donc observer les tendances.

Nous avons calculé la moyenne mobile sur l'ensemble de notre base de données afin de lisser les données et tenter d'améliorer le score de notre modélisation statistique. Pour cela, nous avons utilisé la fonction qui a été développée dans le module "Les étapes d'un projet" que nous avons adapté à notre dataframe.

Cette fonction est composée d'un paramètre '`n=2`' qui correspond à l'étendue de la moyenne pour ajuster le lissage des données. De plus, l'application de la fonction à notre dataframe prend énormément de temps, soit une durée d'environ de 45 minutes.

Score des classifications avec la moyenne mobile

Algorithme de classification	score de modélisation
Régression Logistique	0,552
Forêt Aléatoire	0,559
Gradient Boosting	0,569
Gradient Boosting avec hyperparamètre	0,572

Nous constatons sur les différents algorithmes de classifications que les scores de modélisations ont nettement diminué et que le calcul de la moyenne mobile sur notre base de données n' a pas amélioré nos résultats. Généralement, nos algorithmes de classifications nous procurent des résultats plutôt encourageant dans les alentours de 0,7 - 0,8. Donc, nous nous attendions à obtenir des résultats avec la moyenne roulante bien au-dessus de ce que nous avons l'habitude d'avoir.

Voyons dans l'étape de web scraping, si l'ajout de nouvelles statistiques sur les joueurs et l' application de la fonction de calcul de la moyenne mobile auront des résultats plus optimistes qu'avec le dataframe atp. Il est probable que le manque de statistiques dans notre base de données initiale n'aboutit pas à des résultats de modélisations satisfaisants.

Étape 3: Web Scraping

1 – Le Web Scraping

a) Qu'est ce que le web scraping ?

Le Web Scraping (Scraping signifiant « raclage » ou « grattage » en anglais) est une technologie permettant de récupérer, de manière automatisée ,des données provenant de divers pages ou sites internet.

Ce sont des scripts, des programmes informatiques, qui sont chargés d'extraire ces informations, c'est pourquoi qu'on dit qu'on récupère les données de manières automatisées.

Une fois les données récupérées, nous pouvons les transformer en différents formats (csv, xls...) pour faciliter leur utilisation et exploitation par la suite.

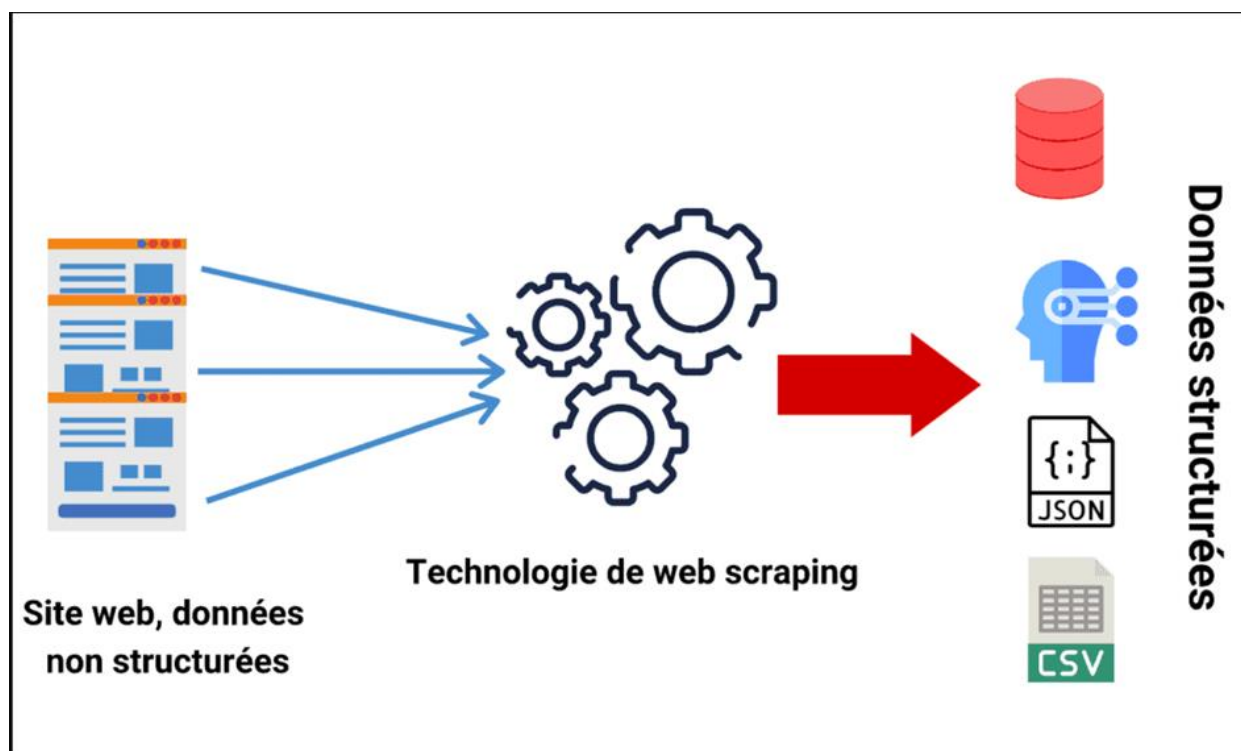


schéma récapitulatif du principe de web scraping (<https://www.data-transitionnumerique.com/web-scraping-python/>)

Le web scraping possède de nombreux intérêts pour une entreprise, il leur permet de se comparer aux autres entreprises et ainsi d'avoir plus d'avantages qu'eux par la suite, il peut aussi permettre d'extraire les avis et retour de leur propres clients et ainsi effectuer une analyse des sentiments pour juger la qualité de leur produit ou service.

Le web scraping, même si légal en France, ne peut pas être utilisé sur tous les sites web.

Il faut respecter les droits d'auteur des sites internet et il est recommandé de lire en amont les conditions d'utilisation afin d'être sûr d'avoir le droit de récupérer les données disponibles sur le site.

Pour bloquer le scraping, il n'est pas étonnant de trouver dans le fichier « robots.txt » du site les pages ou répertoires qui sont interdits à scraper.

b) Pourquoi l'utiliser dans notre projet ?

Pour donner suite aux diverses modélisations effectuées précédemment, nous avons décidé d'améliorer nos résultats en ajoutant diverses statistiques tennistiques pour chaque joueur.

En effet, notre data frame original ne comporte pas ou peu de statistiques que l'on pourrait considérer comme intéressantes pour déterminer qui va gagner un match entre un joueur A et un joueur B.

Rappelons nous des informations dont nous disposons pour le moment dans notre dataframe :

```
['ATP', 'Location', 'Tournament', 'Date', 'Series', 'Court', 'Surface', 'Round', 'Best of', 'Winner', 'Loser', 'WRank', 'LRank', 'Wsets', 'Lsets', 'Comment', 'PSW', 'PSL', 'B365W', 'B365L', 'elo_winner', 'elo_loser', 'proba_elo']
```

Pour déterminer les chances de victoires d'un joueur par rapport à un autre, nous pourrions uniquement regarder le classement des 2 joueurs et parier sur la victoire du joueur ayant le classement le plus élevé ou bien encore comparer leur élo et de même, parier sur le joueur ayant le meilleur élo parmi les 2.

En effet, il est logique de parier pour le plus haut classé lors d'une confrontation, pourquoi parier sur le joueur classé 300 au classement ATP au lieu du numéro 1 mondial.

Néanmoins, les chances de victoire du 300^e mondiale ne sont pas de 0% et de même les chances de victoire du premier joueur mondial ne sont pas de 100%.

C'est là qu'intervient notre web scraping, notre but étant d'améliorer encore plus la précision de nos modélisations en ajoutant diverses statistiques sur les performances globales des joueurs comme par exemple:

- Le pourcentage de coups gagnants
- Le pourcentage de fautes directes
- Le pourcentage de premiers services passés
- Le pourcentage de doubles fautes
- Etc...

c) Où utiliser le web scraping dans notre cas ?

Notre but étant de récupérer des informations et des statistiques sur les joueurs de tennis, il est logique d'aller soit sur le site officiel de l'ATP (Association of Tennis Professionals), divers sites tennistiques ou bien encore des sites de paris en ligne qui pourraient regrouper des informations intéressantes à notre égard.

Notre premier choix s'oriente bien sur le site officiel de l'ATP, <https://www.atptour.com/> , qui regroupe énormément de données pour un joueur :

- Son nombre de match gagné
- Son nombre de match perdu
- Son nombre de titres gagnés durant sa carrière
- Son classement ATP sur différentes périodes
- Ses statistiques durant sa carrière ou à la fin d'une certaine année
- Etc ...

Age 41 (1981.08.08)	Turned Pro 1998	Weight 187 lbs (85kg)	Height 6'1" (185cm)
Birthplace Basel, Switzerland	Plays Right-Handed, One-Handed Backhand	Coach Ivan Ljubicic, Severin Luthi	

OVERVIEW	BIO	ACTIVITY	WIN/LOSS	TITLES & FINALS	PLAYER STATS	RANKINGS HISTORY	RANKINGS BREAKDOWN
----------	-----	----------	----------	-----------------	--------------	------------------	--------------------

Infosys ATP STATS	Career	All Surfaces
--------------------------	--------	--------------

Singles Service Record		Singles Return Record	
Aces	11,478	1st Serve Return Points Won	32%
Double Faults	2,759	2nd Serve Return Points Won	51%
1st Serve	62%	Break Points Opportunities	11,948
1st Serve Points Won	77%	Break Points Converted	41%
2nd Serve Points Won	57%	Return Games Played	18,475
Break Points Faced	6,459	Return Games Won	27%
Break Points Saved	67%	Return Points Won	40%
Service Games Played	18,872	Total Points Won	54%
Service Games Won	89%		
Total Service Points Won	70%		

Néanmoins, comme vu précédemment dans le rapport, il faut vérifier si les données disponibles et qui nous intéressent sont récupérables légalement, pour cela on vérifie le « robot.txt » sur site.

```
User-Agent: *
Disallow: /sitecore/
Disallow: /*?
Disallow: */ajax/*
Disallow: */photos/photo-filter-results
Disallow: */players/*/player-activity
Disallow: */players/*/fedex-atp-win-loss
Disallow: */players/*/player-stats
Disallow: */players/*/rankings-history
Disallow: */players/*/rankings-breakdown
Disallow: */players/*/player-activity
Disallow: */players/*/fedex-atp-win-loss
Disallow: */players/*/player-stats
Disallow: */players/*/rankings-history
Disallow: */players/*/rankings-breakdown
Disallow: */news/news-filter-results
Disallow: */scores/archive/*
Disallow: */search-results
Disallow: */video/video-filter-results
Disallow: */stats/player-tendencies
Disallow: */scores/match-stats
Disallow: */scores/second-screen

Sitemap: http://www.atptour.com/sitemap
```


Après vérification, nous constatons qu'il est hors de question et impossible de récupérer les données sur les pages des joueurs, les données étant protégées, plus particulièrement les données qui nous intéressent : « [*/players/*/player-stats](https://www.ultimatetennisstatistics.com/) »

Nous allons donc rechercher un autre site internet centré sur le tennis où il serait possible de récupérer et stocker les informations qu'ils proposent.

C'est ainsi que notre prochain choix s'est posé sur <https://www.ultimatetennisstatistics.com/>, un site internet qui fournit une multitude d'informations tennistiques depuis 1991.

The screenshot shows the 'Ultimate Tennis Statistics' website. The top navigation bar includes links for GOAT List, Timelines, Rivalries, Rankings, Performance, Statistics, Seasons, Tournaments, Records, Live, and More. A search bar for players is also present. The main content area is divided into two sections: 'Latest Forecasts' and 'Latest Results'. The 'Latest Forecasts' section lists upcoming tournaments with details on surface, draw size, and favorite players. The 'Latest Results' section provides a table of recent matches, including dates, tournaments, surfaces, draw sizes, and match outcomes. On the right side, there are two sidebars: 'Top 10 ATP' showing current rankings and 'Top 10 GOATs' showing all-time greats.

Tournament	Surface	Draw	1-st Favorite	2-nd Favorite	Forecast
Basel	Hard (I)	KO 32	Carlos Alcaraz Garfia 28.4%	Casper Ruud 12.7%	Forecast
Vienna	Hard (I)	KO 32	Daniil Medvedev 21.8%	Jannik Sinner 12.0%	Forecast
Antwerp	Hard (I)	78	Felix Auger Aliassime 100.0%		Forecast
Naples	Hard	49	Lorenzo Musetti 100.0%		Forecast
Stockholm	Hard (I)	59	Holger Vitus Nodskov Rune 100.0%		Forecast

Date	Tournament	Surface	Draw	Part.	Final
10-10-2022	Florence	Hard	62	KO 28	18.6%
03-10-2022	Astana	Hard	64	KO 32	52.3% Novak Djokovic (4 WC) d. Stefanos Tsitsipas (3) 6-3 6-4
03-10-2022	Tokyo	Hard	62	KO 32	28.1% Taylor Harry Fritz (3) d. Frances Tiafoe (4) 7-6(3) 7-6(2)
26-09-2022	Seoul	Hard	63	KO 28	21.6% Yoshihito Nishioka d. Denis Shapovalov (4) 6-4 7-6(5)
26-09-2022	Sofia	Hard (I)	68	KO 28	18.7% Marc Andrea Huesler d. Holger Vitus Nodskov Rune (5) 6-4 7-6(8)
26-09-2022	Tel Aviv	Hard	73	KO 28	19.9% Novak Djokovic (1) d. Marin Cilic (2) 6-3 6-4
19-09-2022	Metz	Hard (I)	59	KO 28	23.0%
19-09-2022	San Diego	Hard	49	KO 28	8.9% Brandon Nakashima (5) d. Marcos Giron (3) 6-4 6-4
29-08-2022	US Open	Hard		KO 128	87.9% Carlos Alcaraz Garfia (3) d. Casper Ruud (5) 6-4 2-6 7-6(1) 6-3
21-08-2022	Winston-Salem	Hard	67	KO 48	20.1% Adrian Mannarino d. Laslo Djere 7-6(1) 6-4

Ici aussi, nous disposons des données techniques et statistiques permettant d'observer les performances sur la carrière globale ou sur une année spécifique d'un joueur :

The screenshot shows the player profile for Roger Federer. The page includes tabs for Profile, Season, Events, Matches, Timeline, Rivalries, Ranking, Performance, Statistics (selected), Tournaments, Streaks, GOAT Points, and Records. Below the tabs, there are filters for Career, All levels, All surfaces, All rounds, and Vs all. The main section displays various statistics categorized into Serve, Return, and Total. The 'Overview' tab is selected, showing a comprehensive breakdown of Federer's performance metrics.

Serve		Return		Total	
	%		%		%
Ace %	10.0%	Ace Against %	5.9%	Points Dominance	1.30
Double Fault %	2.4%	Double Fault Against %	3.5%	Games Dominance	2.37
1st Serve %	62.1%	1st Srv. Return Won %	32.4%	Break Points Ratio	1.26
1st Serve Won %	77.4%	2nd Srv. Return Won %	50.7%	Total Points Won %	54.1%
2nd Serve Won %	57.0%	Break Points Won %	41.1%	Games Won %	58.1%
Break Points Saved %	67.3%	Return Points Won %	39.6%	Sets Won %	75.8%
Service Points Won %	69.7%	Return Games Won %	26.6%	Matches Won %	81.9%
Service Games Won %	88.8%			Match Time	1:39

Comme effectué précédemment, nous devons vérifier que ces données-là soient récupérables et exploitables comparé au site officiel de l'ATP.

Pour cela nous allons vérifier le robot.txt de ce site :

```
User-Agent: *
Disallow: /InProgressEventProbableMatches
Disallow: /InProgressEventPlayerPath
Disallow: /InProgressEventFavorites
Disallow: /h2hHypotheticalMatchup
```

Cette fois ci, les données disponibles sont bien récupérables et exploitables pour la suite de nos travaux, nous allons donc nous concentrer sur le site <https://www.ultimatetennisstatistics.com/>

d) Choix du web scraper

Contrairement à ce que le nom suggère, le web scraper n'est pas la personne qui va s'occuper de scraper les données sur internet mais il s'agit en réalité du programme ou script informatique qui va récupérer le code HTML des sites webs et qui va l'analyser.

Il existe différents types d'outils utilisés pour le web scraping.

Tout d'abord nous avons les extensions de navigateur et modules complémentaires de navigateur, parmi eux WebScraper (extension du navigateur Webscraper.io de Chrome et Firefox) ou encore Data Miner(disponible sur Chrome et Edge).

Ensuite nous avons les logiciels à télécharger en ligne, ceux-ci sont particulièrement utiles pour les personnes ne sachant pas coder, on pourra compter parmi eux Octoparse, Heluim Scraper ou encore ParseHub.

Enfin, nous disposons des librairies/packages disponibles sur Python, c'est avec ces librairies que nous avons décidé de travailler du fait que nous travaillions tous sur Python, nous pouvons nommer tout un lot de librairies comme BeautifulSoup, Scrapy ou bien encore Selenium.

BeautifulSoup pourrait être le package à utiliser pour nos travaux. En effet, il s'agit d'un package très facile à utiliser tout en étant efficace. De plus, il s'agit du package appris pendant la formation DataScientest. Néanmoins, BeautifulSoup ne s'utilise pas sur notre site internet car il est codé en JavaScript, ce qui rend la page internet dynamique, c'est-à-dire que l'on peut interagir et changer des paramètres en direct sur la page.

Career

All levels

All surfaces

All rounds

Vs all

Overview

Aces & DFs

Serve

Serve Speed

Return

Points

Games

Sets & Matches

Performance

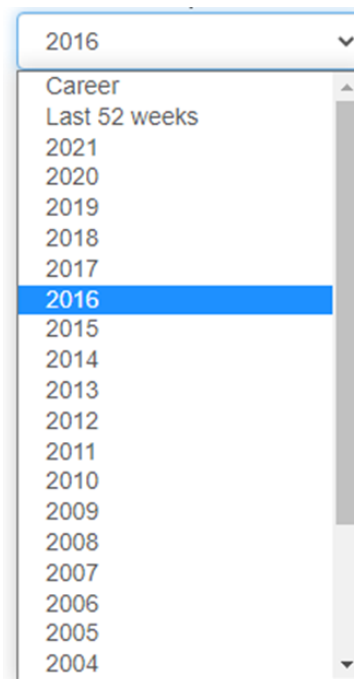
Opponent & Time

Serve	%	Return	%	Total	%
Ace %	10.0%	Ace Against %	5.9%	Points Dominance	1.30
Double Fault %	2.4%	Double Fault Against %	3.5%	Games Dominance	2.37

Ici par exemple il est possible d'interagir avec différentes catégories :

- Career
- All levels
- All surfaces
- All rounds
- Vs all

Nous pouvons changer chacune des catégories en sélectionnant des options plus spécifiques comme par exemple montrer les statistiques sur une année particulière ou bien encore choisir de montrer sur quelle surface.



En cliquant sur l'onglet Career par exemple, nous pouvons sélectionner diverses années en fonction de la carrière du joueur, néanmoins certaines années n'apparaissent pas comme 2022, cela signifie que le joueur est parti à la retraite en 2022 ou qu'il n'a pas joué de match sur l'année 2022.

2016

All levels

Grass

All rounds

Vs all

Overview

Aces & DFs

Serve

Serve Speed

Return

Points

Games

Sets & Matches

Performance

Opponent & Time

Serve	%	Return	%	Total	%
Ace %	11.9%	Ace Against %	8.3%	Points Dominance	1.30
Double Fault %	1.7%	Double Fault Against %	4.0%	Games Dominance	2.35

Dans la capture d'écran, nous avons changé les onglets « Career » et « All surfaces », nous avons choisi l'année 2016 et nous avons sélectionné les statistiques du joueur sur cette année mais uniquement sur Gazon (un des types de surfaces au tennis).

Ainsi, les statistiques ont changé en direct sans avoir chargé une autre page, ce qui rend ainsi la page dynamique et impossible à scraper avec BeautifulSoup.

Pour scraper ce genre de site dynamique, nous avons décidé d'utiliser le package Sélénium de Python.

Sélénium est un outil d'automatisation de test pour le web. Il crée des robots qui peuvent naviguer et interagir dans divers pages web comme pourrait le faire un humain.

Même si Selenium n'est pas en soit un package uniquement centré sur le web scraping, il est plus qu'utile dans notre cas car il permet de prendre en main des codes HTML codé avec du JavaScript, le JavaScript étant le langage de programmation de scripts employé dans les sites webs interactifs.

2 – Application au web scraping

a) Setup de Sélénium

Pour commencer il faut faire installer Sélénium sur là où l'on souhaite travailler, dans notre cas on souhaite l'installer pour l'utiliser sur Python.

Pour se faire, il suffit juste de faire une simple manipulation : `pip install selenium`

```
Requirement already satisfied: selenium in c:\users\pc\anaconda3\lib\site-packages (4.4.3)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\pc\anaconda3\lib\site-packages (from selenium) (0.9.2)
Requirement already satisfied: trio~=0.17 in c:\users\pc\anaconda3\lib\site-packages (from selenium) (0.21.0)
Requirement already satisfied: certifi>=2021.10.8 in c:\users\pc\anaconda3\lib\site-packages (from selenium) (2021.10.8)
Requirement already satisfied: urllib3[socks]~=1.26 in c:\users\pc\anaconda3\lib\site-packages (from selenium) (1.26.9)
Requirement already satisfied: idna in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (3.3)
Requirement already satisfied: sortedcontainers in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: attrs>=19.2.0 in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (21.4.0)
Requirement already satisfied: outcome in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (1.2.0)
Requirement already satisfied: cffi>=1.14 in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (1.15.0)
Requirement already satisfied: async-generator>=1.9 in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (1.10)
Requirement already satisfied: sniffio in c:\users\pc\anaconda3\lib\site-packages (from trio~=0.17->selenium) (1.2.0)
Requirement already satisfied: pycparser in c:\users\pc\anaconda3\lib\site-packages (from cffi>=1.14->trio~=0.17->selenium) (2.21)
Requirement already satisfied: wsproto>=0.14 in c:\users\pc\anaconda3\lib\site-packages (from trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: PySocks!=1.5.7,<2.0,>=1.5.6 in c:\users\pc\anaconda3\lib\site-packages (from urllib3[socks]~=1.26->selenium) (1.7.1)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\pc\anaconda3\lib\site-packages (from wsproto>=0.14->trio-websocket~=0.9->selenium) (0.13.0)
```

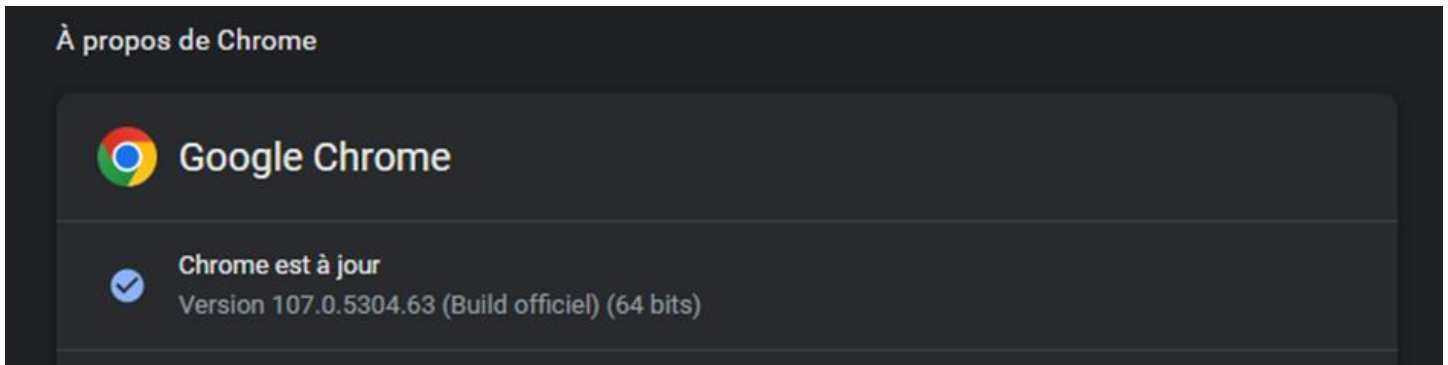
Une fois l'installation terminée, il faut ensuite télécharger un driver, c'est-à-dire un pilote pour le navigateur internet de notre choix.

Ce driver permettra de contrôler le navigateur et de l'utiliser comme si nous étions un utilisateur lambda, comme par exemple taper la recherche sur la barre google, cliquer sur un lien etc.

Il existe différents drivers différents en fonction du navigateur internet utilisé, dans notre cas nous souhaitons travailler sur une page google Chrome.

L'installation de son driver est faite suivant cette page : <https://sites.google.com/chromium.org/driver/>

Avant cela, nous devons regarder quelle est notre version actuelle de chrome dans les paramètres afin d'être sûr de la version du driver à télécharger.



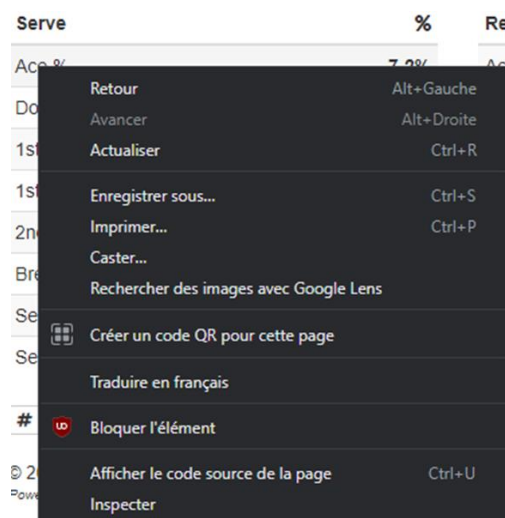
Après téléchargement du bon driver pour la bonne version, nous devons dézipper le fichier et placer le .exe dans un stockage de notre choix.

On pourra ainsi copier le chemin vers l'exécutable par la suite dans notre code. Néanmoins, il faut toujours faire attention aux mises à jour possible et réinstaller le bon driver pour notre version de Google Chrome mise à jour.

Attention, il sera important de noter que toutes les étapes jusqu'à la modélisation se feront à l'aide de Jupyter Notebook et non avec Google Colab. Ceci est dû au fait que Google Colab ne fonctionne pas en local et ne permet pas d'ouvrir un driver indépendant, ce qui rend notre web scraping alors inefficace et qui ne nous rendra pas la totalité des informations(pas de changement de page possible, pas de clic possible sur un joueur ...).

b) Localiser l'élément à scraper et que scraper

Pour tous éléments figurant sur une page web, il est possible de voir son code HTML en faisant un simple clic droit sur l'élément que l'on souhaite examiner puis de cliquer sur « Inspecter »



Une fois l'élément localisé, nous pouvons scraper à l'aide de différents attributs/méthodes :

- ID
- Classe
- XPath
- Nom de balise
- Etc

A l'aide d'un By nous localisons chacun de ses différents attributs/méthodes.

c) Scrapping des variables tennistiques

Pour commencer, nous souhaitons scraper les variables tennistiques qui sont intéressantes pour notre travail.

Nous allons donc nous diriger sur le site vers l'onglet « Statistics » en haut de la page et cliquer dans la catégorie « Statistics Leaders ».

Une fois sur cette page nous nous dirigeons et cliquons sur un joueur quelconque comme John Isner, Ivo Karlovic, Roger Federer etc.

Dès que nous sommes sur la nouvelle page du joueur que l'on a sélectionné, nous cliquons à nouveau sur « Statistics » puis « Statistics ».

Attention, « Statistics » du joueur n'est pas le même que le « Statistics » en haut de page qui permet de revenir à la liste de tous les joueurs du site.

Serve	%	Return	%	Total	%
Ace %	21.2%	Ace Against %	9.1%	Points Dominance	1.05
Double Fault %	2.6%	Double Fault Against %	3.8%	Games Dominance	1.24
1st Serve %	69.1%	1st Srv. Return Won %	22.4%	Break Points Ratio	1.04
1st Serve Won %	78.7%	2nd Srv. Return Won %	41.8%	Total Points Won %	50.6%
2nd Serve Won %	56.0%	Break Points Won %	30.3%	Games Won %	51.2%
Break Points Saved %	70.8%	Return Points Won %	29.6%	Sets Won %	58.6%
Service Points Won %	71.7%	Return Games Won %	10.2%	Matches Won %	61.6%
Service Games Won %	91.8%			Match Time	1:50

Ici, nous cherchons à scraper uniquement les variables tennistiques qui seront les mêmes pour tous les joueurs de tennis, c'est-à-dire les variables tels que « Ace% », « Double Fault% », « Return Points Won% », « Matches Won% » etc.

Comme expliqué précédemment, nous devons mettre la souris sur l'élément que l'on souhaite scraper, c'est-à-dire « Ace% » par exemple, et nous faisons clic droit puis Inspecter.

```
<div class="col-lg-3">
  <table class="table table-condensed table-hover table-striped">
    <thead>_</thead>
    <tbody>
      <tr>
        <td>Ace %</td> == $0
        <th class="text-right pct-data">21.2%</th>
        <th class="raw-data text-right" style="display: none;">13947
          / 65936</th>
      </tr>
      <tr>_</tr>
      <tr>_</tr>
      <tr>_</tr>
      <tr>_</tr>
      <tr>_</tr>
      <tr>_</tr>
```

Nous observons que « Ace% » est une balise HTML de type <td>. On peut supposer que chaque variable soit aussi de ce type-là.

Si c'est le cas, toutes les balises <td>seront scrapées et stockées dans une liste vide.

```

    <td>Matches Won %</td> == $0
    <th class="text-right pct-data">_</th>
    <th class="raw-data text-right" style="display: none;">_</th>
  </tr>
  <tr>_</tr>
</tbody>
```

En faisant clic droit puis Inspecter sur « Matches Won% », on remarque bien qu'il s'agit aussi d'une balise <td>.

La création d' une liste vide que l'on nommera tds_pct nous permettra de stocker toutes les variables récupérées de la page.

Avec cette liste désormais remplie, un dataframe est créé sur pandas que l'on nommera df_td2 avec en colonne toutes les variables qu'on a récupéré, en plus de d'une colonne ['index '] qui nous sera utile par la suite, une fois que nous aurons récupéré toutes les statistiques individuelles de chaque joueur.

	variables	index
0	Ace %	0
1	Double Fault %	1
2	1st Serve %	2
3	1st Serve Won %	3
4	2nd Serve Won %	4
5	Break Points Saved %	5
6	Service Points Won %	6
7	Service Games Won %	7
8	Ace Against %	8
9	Double Fault Against %	9
10	1st Srv. Return Won %	10
11	2nd Srv. Return Won %	11
12	Break Points Won %	12
13	Return Points Won %	13
14	Return Games Won %	14
15	Points Dominance	15
16	Games Dominance	16
17	Break Points Ratio	17
18	Total Points Won %	18
19	Games Won %	19
20	Sets Won %	20
21	Matches Won %	21
22	Match Time	22
--		--

d) Scrapping des statistiques personnelles des joueurs sur l'année 2016

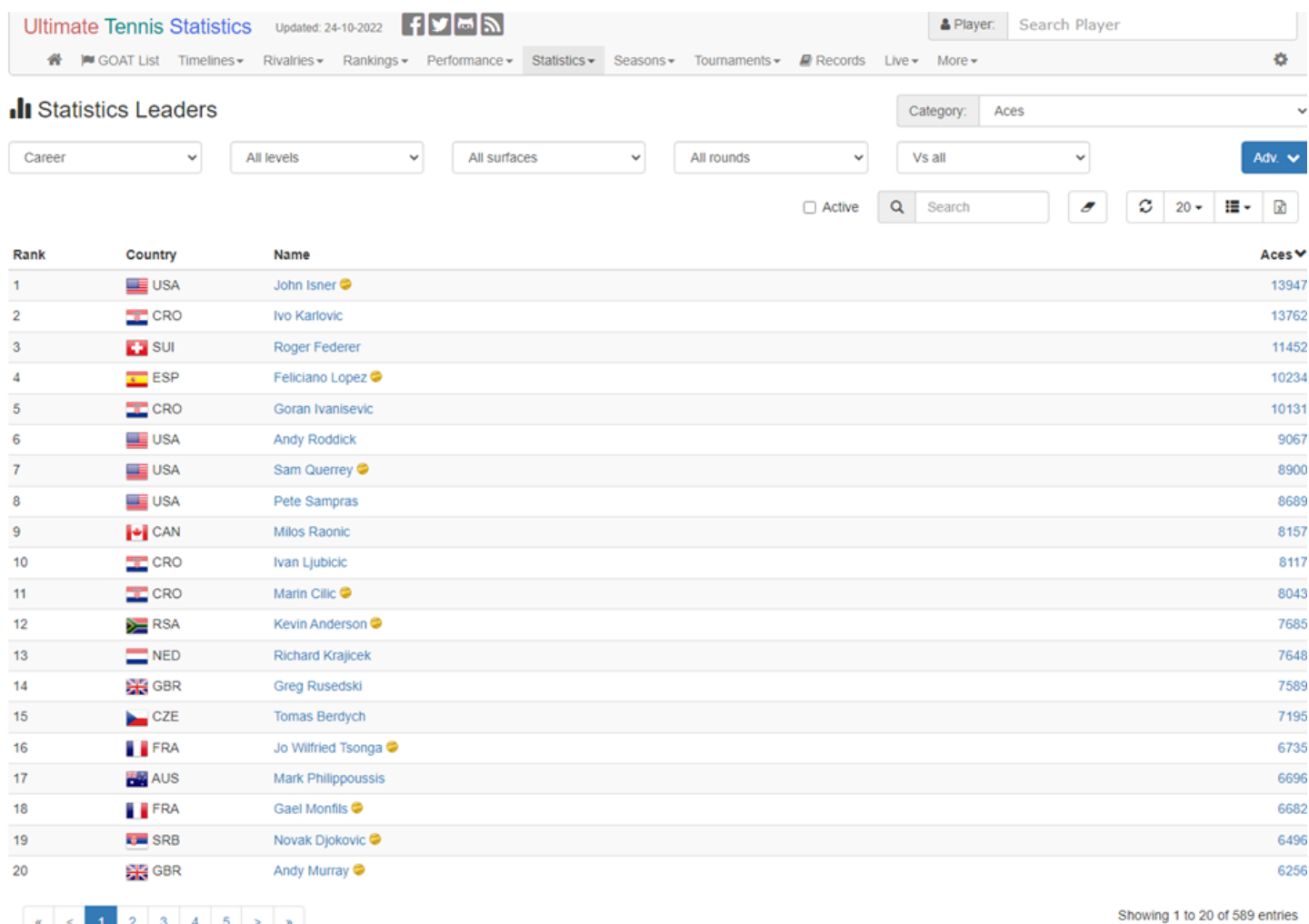
Nous avons récupéré les variables tennistiques qui nous sont importantes pour la réalisation d'une modélisation encore plus approfondie. Maintenant nous devons récupérer pour chaque joueur leur statistiques pour chacune de ces variables.

Nous avons décidé de nous concentrer sur une modélisation de nos prédictions sur l'année 2017 de notre dataframe original. Pour ce faire, nous devons prédire le joueur gagnant pour tous les matchs de l'année 2017 sans connaître ses performances sur l'année même.

Pour se faire, nous devons donc récupérer les statistiques des joueurs sur l'année 2016 afin de tenter de prédire pour l'année 2017. Ce principe marche pour toutes les années que l'on souhaite, si on souhaite prédire des résultats pour une année dite « n », alors il est préférable de prendre les statistiques de l'année « n-1 » afin de prédire les résultats de manière juste et non biaisée.

Encore une fois, nous allons ouvrir un driver nous emmenant sur le site internet de référence et nous allons cliquer sur « Statistics » puis « Statistics Leaders ».

Cette nouvelle page sera la première base de notre scraping des statistiques.



The screenshot shows the 'Ultimate Tennis Statistics' website. The navigation bar includes links for GOAT List, Timelines, Rivalries, Rankings, Performance, Statistics (selected), Seasons, Tournaments, Records, Live, and More. The 'Statistics Leaders' section is active, displaying a list of players ranked by Aces. The table includes columns for Rank, Country, Name, and Aces. The top 20 players are listed, with John Isner at the top with 13947 Aces. The page also features filters for Career, All levels, All surfaces, All rounds, and Vs all, along with a search bar and pagination controls.

Rank	Country	Name	Aces
1	USA	John Isner	13947
2	CRO	Ivo Karlovic	13762
3	SUI	Roger Federer	11452
4	ESP	Feliciano Lopez	10234
5	CRO	Goran Ivanisevic	10131
6	USA	Andy Roddick	9067
7	USA	Sam Querrey	8900
8	USA	Pete Sampras	8689
9	CAN	Milos Raonic	8157
10	CRO	Ivan Ljubicic	8117
11	CRO	Marin Cilic	8043
12	RSA	Kevin Anderson	7685
13	NED	Richard Krajcek	7648
14	GBR	Greg Rusedski	7589
15	CZE	Tomas Berdych	7195
16	FRA	Jo Wilfried Tsonga	6735
17	AUS	Mark Philippoussis	6696
18	FRA	Gael Monfilis	6682
19	SRB	Novak Djokovic	6496
20	GBR	Andy Murray	6256

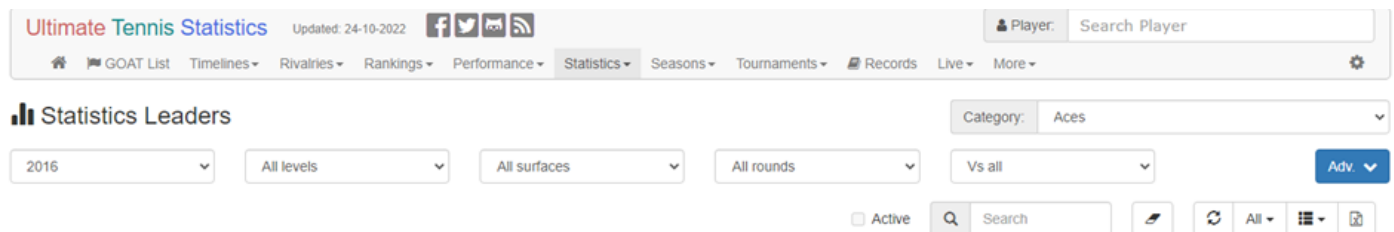
Avant de créer une boucle pour scraper les statistiques propres à chaque joueur, nous allons créer un dataframe que l'on va nommer df_noms.

Ce df_noms est utilisé par la suite dans la boucle afin de passer d'un joueur à un autre automatiquement une fois que la boucle aura fini de scraper les statistiques du premier joueur.

Pour cela, nous allons d'abord vérifier combien de joueurs nous pouvons récupérer sur l'année 2016.

Dans les onglets de la page, il est possible de changer l'année ainsi que le nombre de joueurs affichés sur la page.

Nous changeons donc « Career » par « 2016 » ainsi que « 20 » par « All » .



Après modification, nous voyons qu'il y a 176 joueurs qui apparaissent sur la page, c'est-à-dire qu'il sera possible de scraper les données de chacun de ces 176 joueurs pour 2016.

Nous pouvons commencer à récupérer les noms, on inspecte un des joueurs pour déterminer l'élément à scraper.

```
<a href="/playerProfile?playerId=4544" title="Show John Isner's profile">
John Isner</a>

<a href="/playerProfile?playerId=3333" title="Show Ivo Karlovic's profil
e">Ivo Karlovic</a>
```

Ici, nous devons récupérer les balises <a> correspondant à des liens hypertextes. Comme précédemment pour la partie variable, nous allons stocker tous ces éléments dans une liste vide que l'on aura nommé players_names.

Le dataframe df_noms est créé à partir de cette liste.

Noms - index		
0	John Isner	0
1	Ivo Karlovic	1
2	Gilles Muller	2
3	Milos Raonic	3
4	Marin Cilic	4
...
171	Marton Fucsovics	171
172	Carlos Berlocq	172
173	Pablo Andujar	173
174	Noah Rubin	174
175	Tobias Kamke	175

176 rows × 2 columns

Comme prévu, nous avons 176 lignes qui correspondent aux 176 joueurs dont nous avons des informations à récupérer pour l'année 2016.

Nous pouvons maintenant commencer à scraper les statistiques. Un dataframe vide nommé df_ind est rempli au fur et à mesure.

Pour cela nous allons créer une boucle, son but sera d'aller de la page où les 176 joueurs apparaissent jusqu'à la page des statistiques d'un joueur.

Une fois sur la page du joueur nous souhaitons scraper les pourcentages apparaissant à droite des variables récupérées auparavant.

Serve	%	Return	%	Total	%
Ace %	21.2%	Ace Against %	9.1%	Points Dominance	1.05
Double Fault %	2.6%	Double Fault Against %	3.8%	Games Dominance	1.24
1st Serve %	69.1%	1st Srv. Return Won %	22.4%	Break Points Ratio	1.04
1st Serve Won %	78.7%	2nd Srv. Return Won %	41.8%	Total Points Won %	50.6%
2nd Serve Won %	56.0%	Break Points Won %	30.3%	Games Won %	51.2%
Break Points Saved %	70.8%	Return Points Won %	29.6%	Sets Won %	58.6%
Service Points Won %	71.7%	Return Games Won %	10.2%	Matches Won %	61.6%
Service Games Won %	91.8%			Match Time	1:50

Avant cela, nous faisons en sorte de modifier l'année afin de récupérer les bons pourcentages.

```
<th class="text-right pct-data">21.2%</th> == $0
```

Ici nous allons récupérer tous les éléments des balises <th> correspondant à des cellules d'en-tête et les stocker dans une liste vide nommée ths_pct_player.

Nous allons ensuite utiliser cette liste pour créer un dataframe temporaire entre chaque joueur que l'on nommera df_ind_tmp. Le dataframe comprenant beaucoup de lignes vides (dû au fait que les éléments des balises <th> ne comportent pas tout le temps du texte), nous remplaçons ces vides par des NaN qui seront supprimé du dataframe. La dernière action effectuée dans ce dataframe sera de rajouter un index.

Une fois ce travail sur df_ind_tmp terminé, nous concaténons df_ind, dataframe vide créé au préalable, et df_ind_tmp. Le but est ici de faire une concaténation à chaque statistique récupérée d'un joueur pour remplir df_ind au fur et à mesure, c'est-à-dire qu'entre chaque joueur, une nouvelle colonne s'ajoute dans df_ind, la nouvelle colonne étant les pourcentages récupérés sur un tel joueur.

Après chaque concaténation, on demande à revenir sur la page montrant les 176 joueurs et la boucle se répète jusqu'à récupérer toutes les statistiques nécessaires.

A noter qu'il est important de laisser des temps de repos dans la boucle car les chances d'erreur sont grandes (liée aux nombreuses publicités apparaissant sur les pages scrapées). Des temps de repos de quelques secondes entre chaque changement de page ou après modification d'onglets ("Career" à "2016" ou "20" à "All" par exemple) sont donc définis dans notre programmation.

Finalement, nous récupérerons bien un df_ind avec tous les joueurs et leurs statistiques.

level_0	John Isner	Ivo Karlovic	Gilles Muller	Milos Raonic	Marin Cilic	Nick Kyrgios	Sam Querrey	Andy Murray	Feliciano Lopez	...	Marco Trungelliti	Adrian Ungur	Di Wu	Franco Skugor	Marton Fucsovics	Carlos Berlocq	Pablo Andujar
0	0	Serve	Serve	Serve	Serve	Serve	Serve	Serve	Serve	...	Serve	Serve	Serve	Serve	Serve	Serve	Serve
1	1	25.6%	24.2%	19.1%	16.0%	13.6%	17.4%	16.0%	8.6%	12.6%	...	3.7%	3.1%	3.0%	4.3%	4.3%	2.5%
2	2	2.9%	6.2%	4.6%	4.2%	3.1%	3.7%	3.8%	3.2%	3.7%	...	4.8%	3.8%	2.6%	3.2%	3.7%	3.1%
3	3	69.4%	64.2%	60.5%	64.1%	55.4%	66.5%	58.5%	59.5%	58.3%	...	61.3%	62.0%	63.2%	60.0%	55.7%	72.1%
4	4	82.0%	81.9%	81.4%	80.0%	79.3%	75.9%	79.9%	76.4%	76.8%	...	66.1%	64.0%	61.3%	69.1%	59.6%	60.2%
5	5	55.1%	55.2%	52.6%	55.3%	53.8%	55.2%	51.6%	53.4%	51.3%	...	51.2%	48.3%	48.9%	42.4%	51.3%	45.1%
6	6	71.7%	71.5%	65.4%	69.4%	62.6%	69.0%	64.4%	64.5%	64.8%	...	64.3%	57.1%	59.3%	60.7%	57.8%	57.9%
7	7	73.8%	72.3%	70.0%	71.1%	67.9%	69.0%	68.2%	67.0%	66.2%	...	60.3%	58.1%	56.7%	58.4%	55.9%	56.0%
8	8	93.4%	92.2%	87.9%	90.6%	86.0%	88.6%	85.3%	85.3%	84.6%	...	73.7%	68.9%	66.4%	69.0%	64.0%	63.0%
9	9	Return	Return	Return	Return	Return	Return	Return	Return	...	Return	Return	Return	Return	Return	Return	Return
10	10	9.5%	9.0%	9.0%	6.0%	8.1%	9.9%	9.0%	5.5%	10.7%	...	5.9%	8.0%	6.3%	6.8%	4.1%	3.4%
11	11	3.8%	3.7%	3.7%	3.4%	3.6%	4.2%	3.7%	3.8%	3.9%	...	3.1%	2.6%	4.6%	5.1%	2.9%	4.3%
12	12	22.6%	19.4%	25.2%	28.2%	28.7%	26.1%	25.9%	34.1%	27.8%	...	29.9%	27.8%	33.3%	27.8%	35.4%	32.2%
13	13	42.4%	39.2%	44.0%	49.1%	49.9%	50.0%	44.1%	56.4%	47.1%	...	48.2%	45.7%	55.2%	47.7%	52.3%	50.2%
14	14	32.7%	29.4%	33.2%	35.5%	39.2%	38.0%	36.9%	45.0%	35.3%	...	40.0%	51.4%	41.8%	40.6%	48.1%	54.8%
15	15	30.2%	26.6%	32.1%	36.1%	37.0%	35.7%	33.2%	42.6%	35.2%	...	37.1%	33.6%	41.6%	37.0%	42.9%	39.8%
16	16	10.9%	7.1%	14.1%	18.3%	22.6%	20.1%	16.1%	33.9%	17.1%	...	20.8%	17.6%	30.6%	18.3%	33.8%	32.1%
17	17	Total	Total	Total	Total	Total	Total	Total	Total	...	Total	Total	Total	Total	Total	Total	Total
18	18	1.15	0.96	1.07	1.25	1.15	1.15	1.04	1.29	1.04	...	0.94	0.80	0.96	0.89	0.97	0.91
19	19	1.65	0.90	1.17	1.94	1.61	1.77	1.09	2.30	1.11	...	0.79	0.57	0.91	0.59	0.94	0.87
20	20	1.16	1.03	0.96	1.16	1.05	1.23	1.04	1.27	1.00	...	1.12	1.20	1.03	1.03	1.14	1.30
21	21	51.6%	49.8%	50.9%	53.0%	52.0%	52.2%	50.6%	54.5%	50.5%	...	48.8%	46.3%	49.2%	48.1%	49.6%	48.6%
22	22	52.5%	49.6%	51.4%	54.9%	54.5%	54.8%	51.0%	59.9%	51.0%	...	47.1%	43.0%	48.6%	44.4%	48.7%	48.1%
23	23	59.5%	53.8%	55.6%	69.6%	63.4%	70.1%	55.3%	78.7%	55.1%	...	42.1%	27.3%	37.5%	28.6%	43.8%	39.1%
24	24	61.1%	57.1%	59.3%	75.4%	67.1%	72.2%	54.9%	89.7%	56.4%	...	33.3%	25.0%	44.4%	16.7%	40.0%	40.0%
25	25	2.00	1.49	2.02	1.49	1.54	1.36	1.37	2.09	1.55	...	2.22	1.46	1.41	1.44	2.17	1.43

26 rows x 178 columns

A noter que nous avons 178 colonnes, 176 colonnes représentant chacune les stats d'un joueur, 1 colonne ['level_0'] à supprimer et 1 colonne ['index'].

Il nous reste encore à supprimer les lignes 0,9 et 17 car elles ne correspondent pas à des pourcentages, en effet en récupérant les pourcentages des balises <th>, nous avons aussi récupéré « Serve », « Return » et « Total » qui ne sont pas désirables.

Finalement, l'index est remis à zéro et nous supprimons les colonnes ['index'] de df_ind et de df_td2 (dataframe où l'on a stocké les variables).

Nous fusionnons ensuite ces 2 dataframes dans un dataframe nommé df2016, avec les variables en index et on transpose.

Nous supprimons les NaN possibles pour se retrouver avec le dataframe final df_stats2016.

variables	Ace %	Double Fault %	1st Serve %	1st Serve Won %	2nd Serve Won %	Break Points Saved %	Service Points Won %	Service Games Won %	Ace Against %	Double Fault Against %	...	Return Points Won %	Return Games Won %	Points Dominance	Games Dominance	Break Points Ratio	Total Points Won %	Game Won %
John Isner	25.6%	2.9%	69.4%	82.0%	55.1%	71.7%	73.8%	93.4%	9.5%	3.8%	...	30.2%	10.9%	1.15	1.65	1.16	51.6%	52.5%
Ivo Karlovic	24.2%	6.2%	64.2%	81.9%	55.2%	71.5%	72.3%	92.2%	9.0%	3.7%	...	26.6%	7.1%	0.96	0.90	1.03	49.8%	49.6%
Gilles Muller	19.1%	4.6%	60.5%	81.4%	52.6%	65.4%	70.0%	87.9%	9.0%	3.7%	...	32.1%	14.1%	1.07	1.17	0.96	50.9%	51.4%
Milos Raonic	16.0%	4.2%	64.1%	80.0%	55.3%	69.4%	71.1%	90.6%	6.0%	3.4%	...	36.1%	18.3%	1.25	1.94	1.16	53.0%	54.9%
Marin Cilic	13.6%	3.1%	55.4%	79.3%	53.8%	62.6%	67.9%	86.0%	8.1%	3.6%	...	37.0%	22.6%	1.15	1.61	1.05	52.0%	54.5%
Nick Kyrgios	17.4%	3.7%	66.5%	75.9%	55.2%	69.0%	69.0%	88.6%	9.9%	4.2%	...	35.7%	20.1%	1.15	1.77	1.23	52.2%	54.8%
Sam Querrey	16.0%	3.8%	58.5%	79.9%	51.6%	64.4%	68.2%	85.3%	9.0%	3.7%	...	33.2%	16.1%	1.04	1.09	1.04	50.6%	51.0%
Andy Murray	8.6%	3.2%	59.5%	76.4%	53.4%	64.5%	67.0%	85.3%	5.5%	3.8%	...	42.6%	33.9%	1.29	2.30	1.27	54.5%	59.9%
Feliciano Lopez	12.6%	3.7%	58.3%	76.8%	51.3%	64.8%	66.2%	84.6%	10.7%	3.9%	...	35.2%	17.1%	1.04	1.11	1.00	50.5%	51.0%
Viktor Troicki	10.1%	2.8%	61.4%	72.5%	48.7%	61.0%	63.3%	78.1%	6.1%	4.1%	...	37.1%	21.1%	1.01	0.96	0.97	50.3%	49.7%
Bernard Tomic	11.1%	2.7%	65.7%	73.6%	49.0%	64.6%	65.2%	83.1%	13.8%	3.5%	...	34.2%	17.9%	0.98	1.06	1.09	50.1%	50.7%
Tomas Berdych	10.3%	3.7%	57.4%	78.1%	50.8%	63.3%	66.5%	84.4%	8.8%	3.7%	...	36.7%	21.4%	1.09	1.37	1.05	51.4%	53.1%
Jo Wilfried Tsonga	11.0%	3.9%	60.3%	78.7%	51.9%	67.5%	68.1%	87.9%	6.5%	4.0%	...	35.4%	19.2%	1.11	1.58	1.05	51.6%	53.9%

df_stats2016: dataframe des statistiques de joueurs pour l'année 2016

3- modélisation avec statistiques scrapées

Pour utiliser les statistiques scrapées, nous devons réaliser une jointure entre les noms des joueurs (df_joueur: dataframe créé pour la modélisation) et les noms de joueurs listés en index du dataframe scrapé df_stats2016.

a) préparation du dataframe et réalisation de la jointure entre les 2 df

- Df_joueur: étape de text mining

	Player	Player_id
0	Dosedel S.	1000
1	Kiefer N.	1001
2	Gaudio G.	1002
3	El Aynaoui Y.	1003
4	Cherkasov A.	1004
...
1397	Andreev A.	2397
1398	Korda S.	2398
1399	Auger-Aliassime F.	2399
1400	Gaston H.	2400
1401	Seyboth Wild T.	2401

1402 rows x 2 columns

df_joueur

le dataframe df_joueur est composé de 2 colonnes:

- 'Player' correspondant aux noms de famille des joueurs ainsi que l'initiale de leur(s) prénom(s) suivie d'un ".".
- 'Player_id' correspondant à l'identifiant numérique unique que nous avons créé pour chaque joueur.

Le travail à réaliser sur ce dataframe est d'utiliser la colonne 'Player' et de séparer le nom de famille des prénoms. Pour cela nous avons choisi de réaliser une étape de text mining.

Après concaténation de la colonne 'Player' (sous forme de texte), nous avons définie une expression régulière correspondant à tous les cas rencontrés tels que:

- Nom + P.
- Nom + P₁.P₂.
- Nom₁ Nom₂ + P₁.
- Nom₁ Nom₂ Nom₃ + P₁.
- Nom₁ Nom₂ Nom₃ Nom₄ + P₁.
-

La variable texte obtenue (nom_joueur) est ensuite transformée en Serie pour être ajoutée en tant que colonne 'Name' dans le dataframe df_joueur.

	Player	Player_id	Name
0	Dosedel S.	1000	Dosedel
1	Kiefer N.	1001	Kiefer
2	Gaudio G.	1002	Gaudio
3	El Aynaoui Y.	1003	El Aynaoui
4	Cherkasov A.	1004	Cherkasov

Df_joueur avec colonne 'Name'

- Df_stats2016: étape de préparation du dataframe

```
> <class 'pandas.core.frame.DataFrame'>
RangeIndex: 176 entries, 0 to 175
Data columns (total 24 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   Unnamed: 0                                176 non-null    object
1   Ace %                                     176 non-null    object
2   Double Fault %                           176 non-null    object
3   1st Serve %                              176 non-null    object
4   1st Serve Won %                           176 non-null    object
5   2nd Serve Won %                           176 non-null    object
6   Break Points Saved %                     176 non-null    object
7   Service Points Won %                     176 non-null    object
8   Service Games Won %                      176 non-null    object
9   Ace Against %                            176 non-null    object
10  Double Fault Against %                   176 non-null    object
11  1st Srv. Return Won %                    176 non-null    object
12  2nd Srv. Return Won %                    176 non-null    object
13  Break Points Won %                       176 non-null    object
14  Return Points Won %                      176 non-null    object
15  Return Games Won %                       176 non-null    object
16  Points Dominance                          176 non-null    float64
17  Games Dominance                          176 non-null    float64
18  Break Points Ratio                       176 non-null    float64
19  Total Points Won %                       176 non-null    object
20  Games Won %                              176 non-null    object
21  Sets Won %                               176 non-null    object
22  Matches Won %                            176 non-null    object
23  Match Time                               176 non-null    object
dtypes: float64(3), object(21)
memory usage: 33.1+ KB
```

df_stats2016.info()

Concernant le df_stats2016, les étapes de préparation de ce df pour la jointure des noms de joueurs et la modélisations sont les suivantes:

- création d'un "sous-dataframe" contenant les variables de type "object"
- suppression des "%" dans la majorités des colonnes "object": par la création d'une fonction retirant le dernier caractère des colonnes
- transformations des colonnes de type object en float
- concaténation des colonnes transformées avec le df_stats2016 initial
- utilisation de la colonne 'Player' pour séparer les noms et prénoms en créant deux fonctions "split_p" et split_n"
- ajustement du dataframe obtenu en réalisant des changements manuellement pour les lignes non-transformées avec les fonctions split_p (colonne "1stName") et split_n(colonne "Name").

Ces transformations nous permettent d'obtenir le dataframe final df_stats2016_final utilisable pour la modélisation après jointure avec le df_joueur:

	Player	Points Dominance	Games Dominance	Break Points Ratio	Match Time	Ace %	Double Fault %	1st Serve %	1st Serve Won %	2nd Serve Won %	...	2nd Srv. Return Won %	Break Points Won %	Return Points Won %	Return Games Won %	Total Points Won %	Games Won %	Sets Won %	Matches Won %	1stName	Name
0	John Isner	1.15	1.65	1.16	2:00	25.6	2.9	69.4	82.0	55.1	...	42.4	32.7	30.2	10.9	51.6	52.5	59.5	61.1	John	Isner
1	Ivo Karlovic	0.96	0.90	1.03	1:49	24.2	6.2	64.2	81.9	55.2	...	39.2	29.4	26.6	7.1	49.8	49.6	53.8	57.1	Ivo	Karlovic
2	Gilles Muller	1.07	1.17	0.96	2:02	19.1	4.6	60.5	81.4	52.6	...	44.0	33.2	32.1	14.1	50.9	51.4	55.6	59.3	Gilles	Muller
3	Milos Raonic	1.25	1.94	1.16	1:49	16.0	4.2	64.1	80.0	55.3	...	49.1	35.5	36.1	18.3	53.0	54.9	69.6	75.4	Milos	Raonic
4	Marin Cilic	1.15	1.61	1.05	1:54	13.6	3.1	55.4	79.3	53.8	...	49.9	39.2	37.0	22.6	52.0	54.5	63.4	67.1	Marin	Cilic
...
171	Marton Fucsovics	0.97	0.94	1.14	2:17	4.3	3.7	55.7	59.6	51.3	...	52.3	48.1	42.9	33.8	49.6	48.7	43.8	40.0	Marton	Fucsovics
172	Carlos Berlocq	0.91	0.87	1.30	1:43	2.7	3.1	72.1	60.2	45.1	...	50.2	54.8	39.8	32.1	48.6	48.1	39.1	40.0	Carlos	Berlocq
173	Pablo Andujar	0.88	0.57	0.91	1:46	2.5	3.2	66.3	65.6	43.9	...	54.5	43.9	36.5	18.9	47.6	42.6	25.0	12.5	Pablo	Andujar
174	Noah Rubin	0.84	0.47	0.83	1:49	2.6	4.2	61.3	67.0	53.3	...	48.7	31.0	32.2	11.0	47.3	44.5	37.5	33.3	Noah	Rubin
175	Tobias Kamke	0.96	0.93	0.88	1:36	1.4	4.3	61.7	65.1	49.3	...	52.0	46.9	39.5	28.6	49.5	49.1	47.8	50.0	Tobias	Kamke

176 rows x 26 columns

df_stats2016_final

- Jointure entre le df_stats2016_final et le df_joueur

La jointure entre les 2 dataframe a été réalisée avec la fonction “merge” en utilisant la clé de jointure “Name” présente dans les 2 df.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1408 entries, 0 to 1407
Data columns (total 28 columns):
#   Column                                Non-Null Count  Dtype
---  ---
0   Player_x                              1408 non-null   object
1   Player_id                             1408 non-null   int64
2   Name                                  1408 non-null   object
3   Player_y                              223 non-null    object
4   Points Dominance                      223 non-null    float64
5   Games Dominance                       223 non-null    float64
6   Break Points Ratio                    223 non-null    float64
7   Match Time                           223 non-null    object
8   Ace %                                223 non-null    float64
9   Double Fault %                       223 non-null    float64
10  1st Serve %                           223 non-null    float64
11  1st Serve Won %                       223 non-null    float64
12  2nd Serve Won %                       223 non-null    float64
13  Break Points Saved %                  223 non-null    float64
14  Service Points Won %                  223 non-null    float64
15  Service Games Won %                  223 non-null    float64
16  Ace Against %                        223 non-null    float64
17  Double Fault Against %                223 non-null    float64
18  1st Srv. Return Won %                 223 non-null    float64
19  2nd Srv. Return Won %                 223 non-null    float64
20  Break Points Won %                    223 non-null    float64
21  Return Points Won %                   223 non-null    float64
22  Return Games Won %                    223 non-null    float64
23  Total Points Won %                    223 non-null    float64
24  Games Won %                           223 non-null    float64
25  Sets Won %                            223 non-null    float64
26  Matches Won %                         223 non-null    float64
27  1stName                               223 non-null    object
dtypes: float64(22), int64(1), object(5)
memory usage: 319.0+ KB
```

dataframe joueur jointure (concaténation du df_joueur et df_stats2016_final)

b) modélisation avec les statistiques scrapées

- modélisation sans moyenne mobile

Dans cette partie nous allons réaliser une nouvelle modélisation en utilisant les données statistiques scrapées sur l'année 2016 pour prédire les résultats de l'année 2017.

Dans cet objectif, nous avons filtré les données de df_atp_final pour extraire les lignes de matchs joués en 2016 et 2017.

Par conséquent, dans notre cas, la séparation des données d'entraînement des données de test (train_test_split) s'est faite en séparant la variable cible (Player_cible de 2017) des variables explicatives (data de l'année 2016).

Nous avons choisi de garder les classifieurs testés dans les précédentes modélisations, c'est-à-dire RandomForest et GradientBoosting (dans les 2 cas avec fixation des hyperparamètres).

```
'le score du model RandomForest:'
0.8175771971496437
'Matrice de confusion modele RandomForest:'
classe predite      0      1
classe reel
0      0.412827  0.087411
1      0.095012  0.404751
'le score du model GradientBoosting:'
0.807125890736342
'Matrice de confusion modele GradientBoosting:'
classe predite      0      1
classe reel
0      0.404751  0.095487
1      0.097387  0.402375
```

matrices de confusion et score moyen des modélisations randomForest et GradientBoosting avec stats2016

Les résultats obtenus montrent une très sensible amélioration du score moyen de chaque modèle en comparaison avec la modélisation avec hyperparamètres sans statistique mais ne change pas radicalement la performance de nos modèles.

- interprétation des résultats

le rapport de classification pour RandomForest paramètres optimisés est:

	precision	recall	f1-score	support
0	0.80	0.85	0.82	1081
1	0.84	0.80	0.82	1094
accuracy			0.82	2175
macro avg	0.82	0.82	0.82	2175
weighted avg	0.82	0.82	0.82	2175

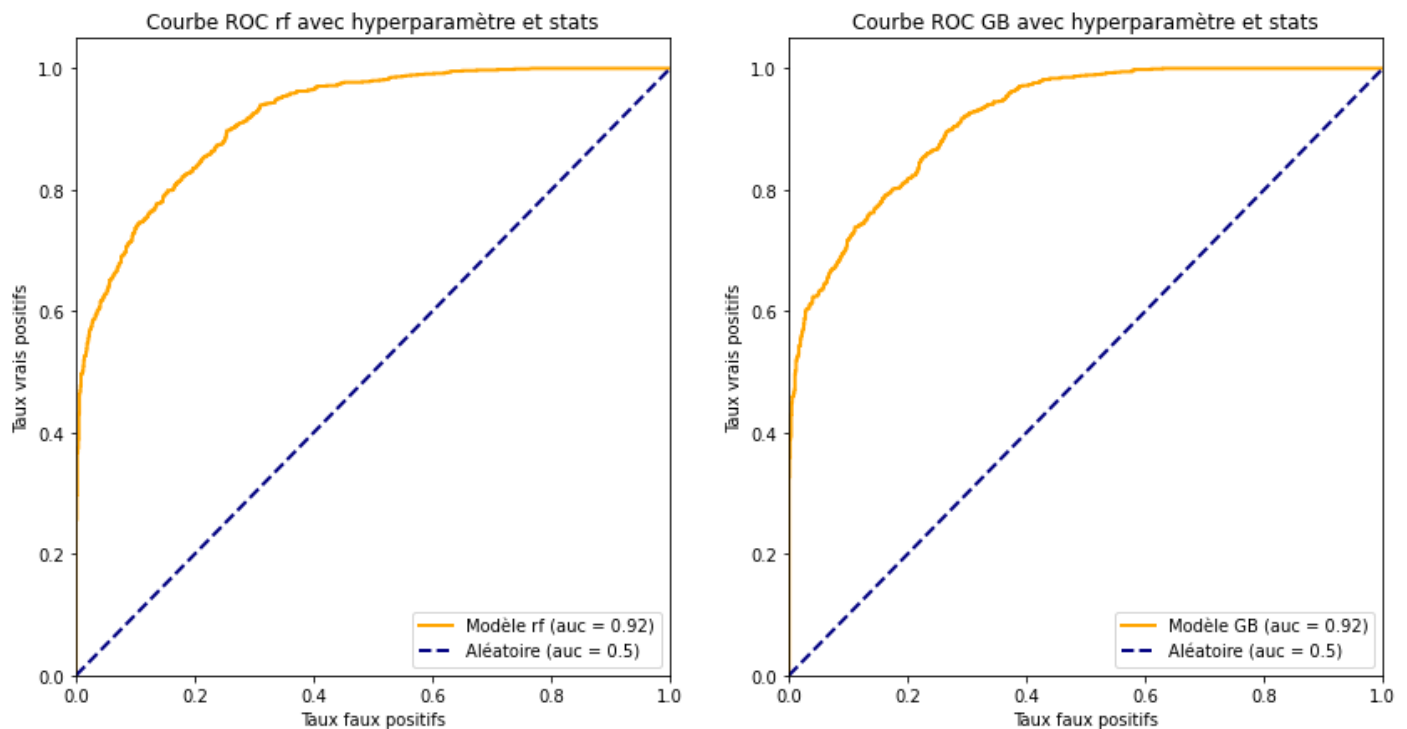
le rapport de classification pour GradientBoosting paramètres optimisés est:

	precision	recall	f1-score	support
0	0.80	0.84	0.82	1081
1	0.83	0.79	0.81	1094
accuracy			0.81	2175
macro avg	0.81	0.81	0.81	2175
weighted avg	0.81	0.81	0.81	2175

rapport de classification de RandomForest et GradientBoosting avec hyperparamètres et stats2016

Les rapports de classifications nous révèlent une très bonne performance de nos modèles concernant la précision sur les joueurs gagnants et le rappel des joueurs perdants. Les scores d'accuracy sont similaires aux précédentes modélisations.

Nos modèles ont des performances similaires aux précédents, cependant nous constatons que le taux de vrais positifs est légèrement supérieur et va donc améliorer notre prédictions de joueur gagnant un match donné.



courbes ROC et AUC pour les modélisations avec hyperparamètres et statistiques 2016

les courbes ROC et l'AUC nous montrent une meilleure catégorisation des vrais positifs (c'est à dire des joueurs gagnants), par conséquent, nos modèles RF et GB avec stats et hyperparamètres sont performants pour prédire un joueur gagnant d'un match donné (testé sur l'année 2016 afin de prédire les résultats de 2017).

Une dernière étape dans la modélisation, est de tester nos modèles avec la moyenne mobile

- **Modélisation avec moyenne mobile**

Nous avons décidé de tester une dernière modélisation pour tenter d'améliorer notre modèle, en utilisant la moyenne mobile sur les variables statistiques scrapés.

Pour comparer avec les résultats précédents, nous choisissons de garder les classifieurs RandomForest et GradientBoosting avec hyperparamètres. Nous obtenons des matrices de confusion et scores moyens bien en dessous des performances de nos autres modèles. L'utilisation de la moyenne mobile dans notre cas de paris sportif influence négativement la capacité de notre modèle à détecter la bonne catégorie.

```

↳ 'le score du model RandomForest:'
0.5809567617295308
'Matrice de confusion modele RandomForest:'
classe predite      0.0      1.0
classe reelle
0.0      0.260810  0.235971
1.0      0.183073  0.320147
'le score du model GradientBoosting:'
0.5699172033118676
'Matrice de confusion modele GradientBoosting:'
classe predite      0.0      1.0
classe reelle
0.0      0.336247  0.160534
1.0      0.269549  0.233671

```

matrices de confusion RF et GB avec hyperparamètres, stats 2016 et moyenne mobile

le rapport de classification pour RandomForest paramètres optimisés est:

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

0.0	0.59	0.53	0.55	1080
1.0	0.58	0.64	0.60	1094

accuracy			0.58	2174
macro avg	0.58	0.58	0.58	2174
weighted avg	0.58	0.58	0.58	2174

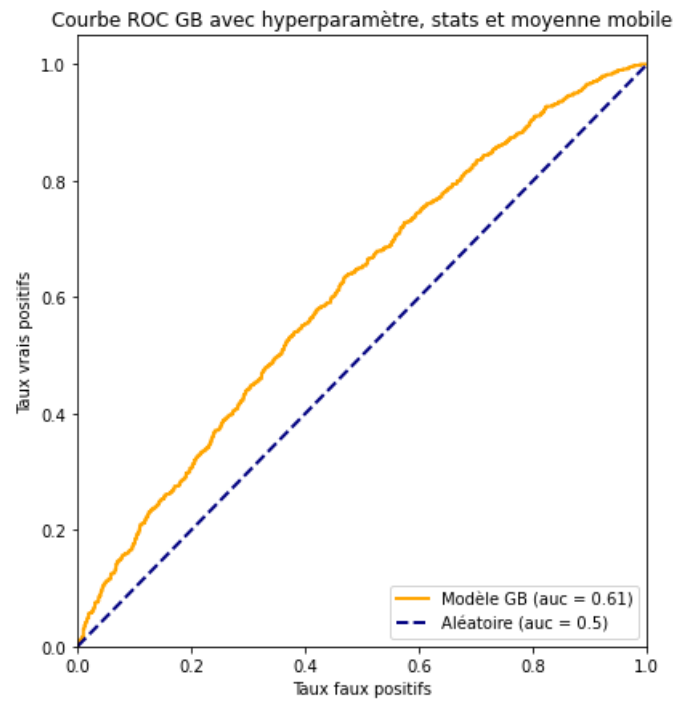
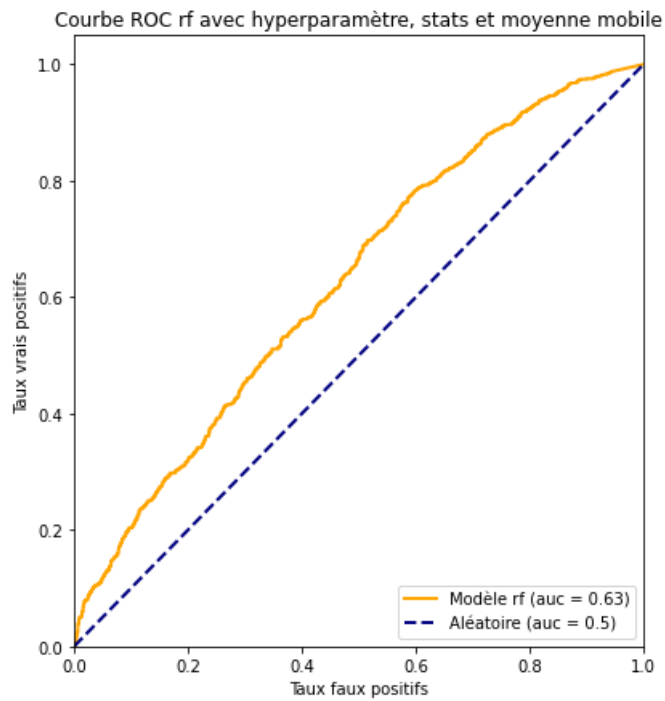
le rapport de classification pour GradientBoosting paramètres optimisés est:

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

0.0	0.56	0.68	0.61	1080
1.0	0.59	0.46	0.52	1094

accuracy			0.57	2174
macro avg	0.57	0.57	0.57	2174
weighted avg	0.57	0.57	0.57	2174

rapport de classification de RandomForest et GradientBoosting avec hyperparamètres stats2016 et moyenne mobile



courbes ROC et AUC pour les modélisations avec hyperparamètres ,statistiques 2016 et moyenne mobile

Concernant l'aire sous la courbe ROC celles-ci sont de 0.63 et 0.61 ce qui est bien en dessous de nos résultats précédents et montrent une moindre capacité de notre modèle à catégoriser correctement les joueurs gagnants.

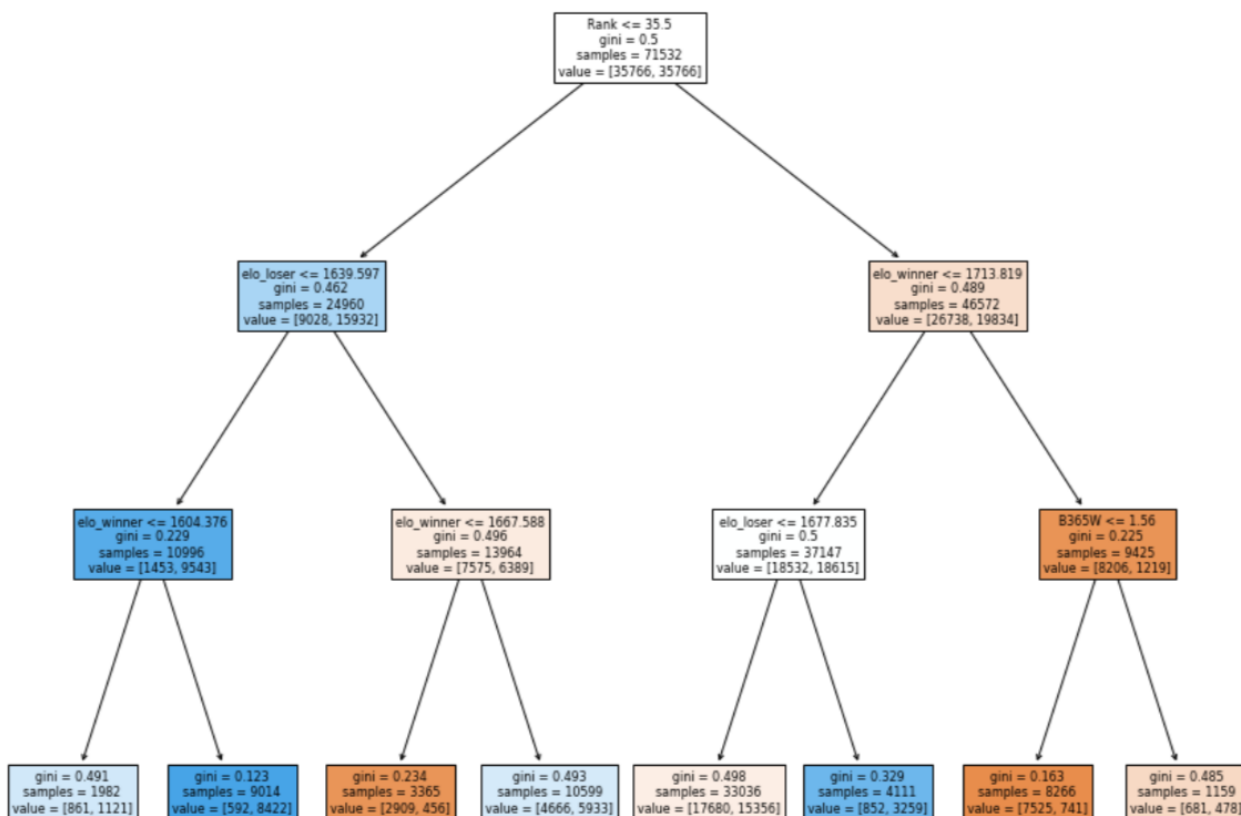
4- Interprétabilité

Après avoir préparé nos données pour l'entraînement de nos différentes modélisations et tester de multiples algorithmes de classifications, la phase importante lorsque l'on produit des modélisations statistiques est l'interprétation des résultats.

La phase d'interprétabilité a pour but de comprendre comment le modèle réagit avec l'ensemble des données que nous lui avons injectées dans sa machine.

a) L'arbre de décision

L'arbre de décision peut nous aider à comprendre les décisions du modèle entraîné à l'aide de la fonction `plot_tree`.



On observe que les variables mises en évidence dans l'arbre de décision sont 'Rank': le rang des joueurs, 'elo_loser', 'elo_winner': les niveaux de jeu des joueurs perdants et gagnants, et 'B365W': la probabilité des joueurs gagnants.

L'arbre contient la distribution en fréquence de la variable à prédire soit 71.532 sont utilisées pour la construction de l'arbre de décision. La variable 'Rank' est la première variable utilisée puis elle produit deux nœuds fils c'est-à-dire les variables 'elo_loser' et 'elo_winner'. Ces dernières créent également d'autres nœuds. Donc, le 'elo_loser' va produire deux 'elo_winner' tandis que le 'elo_winner' donne un 'elo_loser' et 'B365W'.

Donc l'arbre de décision nous démontre que seulement quatre variables contribuent à classer nos joueurs gagnants et perdants. Il serait intéressant de retirer la probabilité des joueurs gagnants du bookmaker 'Bet365' car cela biaise nos estimations des joueurs. Il est cohérent que nous obtenions de bons résultats sur les

joueurs car nous nous appuyons sur les probabilités du bookmakers alors que le but est de produire la meilleure estimation des joueurs gagnants.

b) Les variables les plus importantes

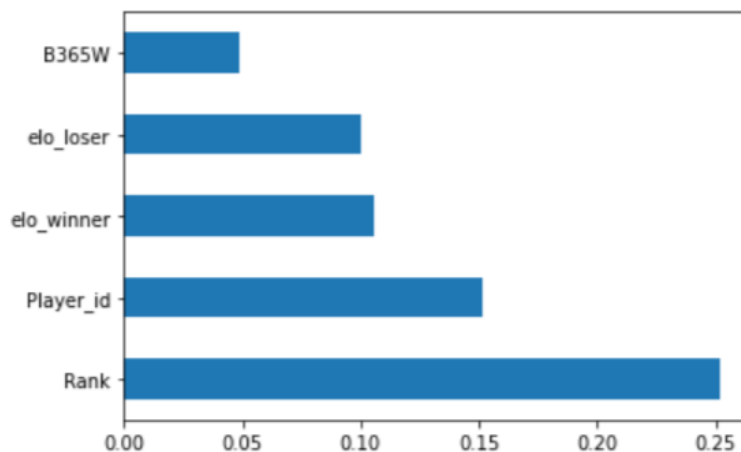
Observons les variables explicatives les plus importantes qui ont un impact sur les résultats de modélisation de la forêt aléatoire (rf1) et pour rappel les résultats du rapport de classification, ci-dessous:

le rapport de classification pour RF est:

	precision	recall	f1-score	support
0	0.75	0.85	0.79	8942
1	0.82	0.72	0.77	8942
accuracy			0.78	17884
macro avg	0.79	0.78	0.78	17884
weighted avg	0.79	0.78	0.78	17884

La modélisation statistique avec la méthode de forêt aléatoire présente un score de 0,78 ce qui est très significatif. De plus, la précision de prédiction des joueurs confirme que le modèle prédit assez bien les joueurs car le niveau de précision des joueurs perdants est de 0.75 et de 0.82 pour les joueurs gagnants.

Désormais, observons les variables explicatives ou les features qui expliquent la classification des joueurs et également un score de modélisation significatif.



Tout comme l'arbre de décision, nous retrouvons les quatre variables suivantes, 'elo_loser', 'elo_winner': les niveaux de jeu des joueurs perdants et gagnants, et 'B365W': la probabilité des joueurs gagnants. L'histogramme montre bien que le rang est fortement lié à la variable cible qui se nomme 'Player_cible' et elle correspond à la détermination des joueurs gagnants et perdants. Ensuite, on voit apparaître une nouvelle variable qui est le 'Player_id' soit l'identifiant du joueur. C'est une variable que nous avons construite à partir des joueurs et il n'est donc pas étonnant que la variable soit sur le graphique. Nous retrouvons le 'elo_winner', 'elo_loser et enfin 'B365W'.

Nous pouvons conclure que le rang et le niveau des joueurs du df_atp explique la classification des joueurs gagnants et perdants. Enfin, la probabilité des joueurs gagnants du bookmakers 'Bet365' peut être exclue car elle biaise nos résultats mais en réalité la variable ne semble pas avoir trop de poids dans la classification.

c) Les variables les plus importantes - web scraping

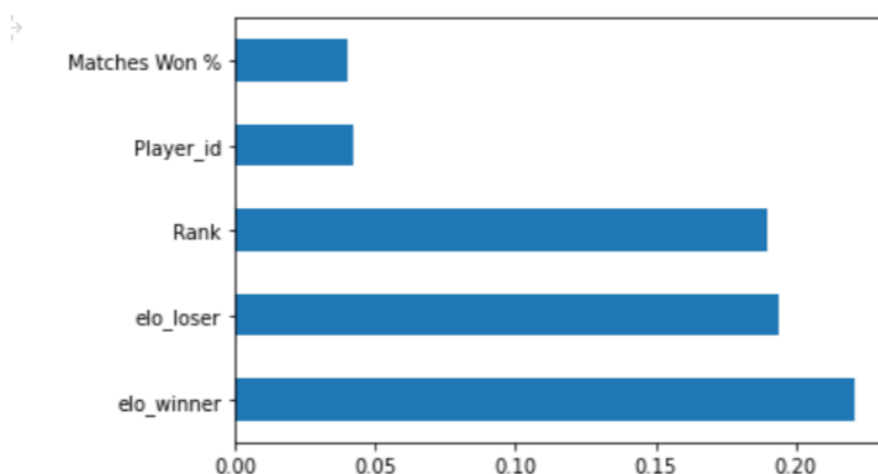
Afin d'améliorer nos résultats de modélisation, nous avons construit une boucle de web-scraping qui va récupérer les statistiques supplémentaires sur les joueurs. Observons à partir des données 'df_atp' et les statistiques, les variables explicatives les plus importantes qui ont un impact sur les résultats de modélisation de la forêt aléatoire (RF1617ST). Pour rappel les résultats du rapport de classification, ci-dessous:

```
le rapport de classification pour RandomForest paramètres optimisés est:
      precision    recall  f1-score   support

     0       0.81      0.85      0.82      1081
     1       0.84      0.80      0.82      1094

 accuracy          0.82      2175
 macro avg         0.82      0.82      0.82      2175
 weighted avg      0.82      0.82      0.82      2175
```

La modélisation statistique avec la méthode de forêt aléatoire présente un score de 0,82 ce qui est très significatif et on observe une nette amélioration du score avec les données statistiques des joueurs que nous avons récupéré sur internet. De plus, la précision de prédiction des joueurs confirme que le modèle prédit très bien les joueurs car le niveau de précision des joueurs perdants est de 0.81 VS 0.75(rf1) et de 0.84 VS 0.82(rf1) pour les joueurs gagnants.



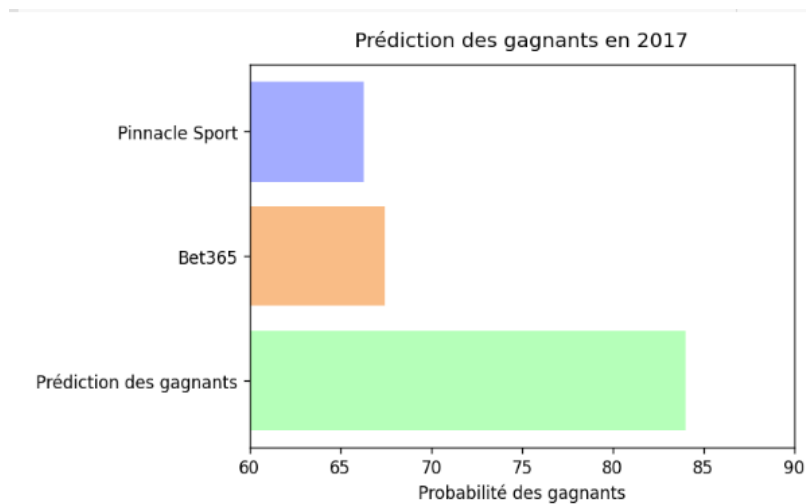
Les variables les plus importantes dans la modélisation sont le 'elo_winner', le 'elo_loser' et le 'Rank'. Ensuite les variables 'Player_id' n'est pas très significatives dans la modélisation et on voit bien que la corrélation avec la variable cible est moins importante. On constate l'apparition d'une nouvelle variable qui vient des statistiques que nous avons récupérées sur internet. Il s'agit des matchs gagnés par les joueurs.

Globalement, cette modélisation semble être la plus pertinente lorsque l'on observe l'importance des variables et on ne retrouve pas la probabilité des joueurs gagnants du bookmakers 'Bet365' qui aurait pu biaiser la classification de nos joueurs. L'ajout de statistiques contribue également à améliorer les résultats de la modélisation comme nous l'indiquent le rapport de classification et l'histogramme.

En conclusion, les probabilités des niveaux des jeu des joueurs, le rang ainsi que le pourcentage de matchs gagnés contribuent à rendre performant la modélisation de classification des joueurs.

Conclusion

L'objectif de ce projet 'Beat the bookmakers' est de construire une modélisation de classification des joueurs ayant une estimation de probabilité des joueurs gagnants plus importantes que celle des bookmakers 'Pinnacle Sport' et 'Bet365'.



L'histogramme démontre bien que notre prédiction est bien au-dessus des probabilités des bookmakers sur l'année 2017. Ce sont les prédictions construites à partir des données 'df_atp' et des statistiques de notre web-scraping. Il aurait été intéressant de récupérer les statistiques de l'ensemble des périodes (2000 - 2018) à l'image du 'df_atp' afin de réellement comparer nos prédictions des joueurs gagnants avec les bookmakers sur l'ensemble de la base de données.

Lexique

Définition des variables de la base de donnée 'atp_data.csv':

Variables	Définitions
ATP	Le numéro du tournoi ATP
Location	Lieu du Tournoi
Tournament	Nom du tournoi (y compris le sponsors)
Date	Date du match (remarque: avant 2003, la date indiquée pour tous les matchs joués dans un même tournoi est la date de début)
Series	Nom de la série de tennis ATP (Grand Chelem, Masters, International ou International Gold)
Court	Type de terrain (extérieur ou intérieur)
Surface	Type de surface (argile, dur, moquette ou herbe)
Round	Tour de match
Best of	Nombre maximum de sets jouables en match
Winner	Vainqueur du match
Loser	Perdant du match
Wrank	Classement ATP du vainqueur du match au début du tournoi
Lrank	Classement ATP du perdant du match au début du tournoi
Wsets	Nombre de sets gagnés par vainqueur du match
Lsets	Nombre de sets gagnés par le perdant du match
Comment	Commentaire sur le match (Terminé, gagné grâce à l'abandon du perdant ou via Walkover)
PSW	Pinnacles Sports probabilité du vainqueur du match
PSL	Pinnacles Sports probabilité du perdant du match
B365W	Bet365 probabilité du gagnant du match
B365L	Bet365 probabilité du perdant du match
Elo_winner	Niveau de jeu des joueurs gagnants
Elo_loser	Niveau de jeu des joueurs perdants
Proba_elo	Probabilité de gagner

Définition des variables appartenant au code

Variables	Définitions
df_atp	Nom du Dataframe donnée pour la lecture du fichier .scv initial
num_df_atp	Nom du Dataframe contenant les variables numériques (float64 et int64) de df_atp
num_stats	Nom du Dataframe affichant les indicateurs statistiques des variables numériques (num_df_atp) c'est-à-dire moyenne, quartiles, médianes, min&max
var_num	Dataframe affichant les variables de num_stats pour lesquelles la différence 'moyenne-médiane' est supérieure à 18
stats_desc	Dataframe affichant la description numériques des variables de num_df_atp
stats_std	Dataframe affichant la moyenne, l'écart type de stats_desc ainsi que la moyenne-écart-type (I1) et la moyenne+l'écart-type(I2)
cat_df_atp	Dataframe regroupant les variables catégorielles de df_atp
df_atp_2002	Dataframe regroupant l'ensemble des matchs jouant durant l'année 2002
df_atp_2016	Dataframe regroupant l'ensemble des matchs jouant durant l'année 2016
df_winner	Dataframe regroupant l'ensemble des joueurs de Tennis gagnant, leur classement et le match_id
df_loser	Dataframe regroupant l'ensemble des joueurs de Tennis perdant, leur classement et le match_id
df_player	Dataframe correspondant à la concaténation de df_winner et df_loser et permettant l'élimination de la mention gagnant/perdant
df_joueur	dataframe contenant le nom des joueurs et l'initiale de leur prénom dans la 1ere colonne, et le player_id (numéro unique de chaque joueur)
df_player_id	Dataframe df_player avec ajout d'une colonne d'identifiant unique à chaque joueur
df_atp_encodage	Dataframe après encodage des variables catégorielles
df_atp_final	dataframe après dichotomisation des variables tournament et location
df_book	Dataframe regroupant toutes les côtes des 2 bookmakers (Pinacle et Bet365) pour l'ensemble des années étudiées(2000-2018)
df_atp_moyenne_mobile	dataframe après application de la fonction moyenne mobile
df_ind	Dataframe de données scrapées des statistiques des joueurs de tennis pour l'année 2016
df_td2	Dataframe de données scrapées des variables tennistiques
RFx	Classifieur RandomForest utilisé en modélisation (x étant un entier dépendant de la modélisation)
GBx	Classifieur GradientBoosting utilisé en modélisation (x étant un entier dépendant de la modélisation)
target_mb, data_mb	ensembles train/test utilisés avec le dataset moyenne mobile pour la modélisation

df_stats2016	dataframe contenant les joueurs de l'année 2016 et leur statistiques de matchs (informations scrapées)
df_stats2016obj	dataframe df_stats2016 contenant uniquement les variables de type object
joueur_jointure	dataframe correspondant à la concaténation du df_joueur et du df_stats2016 avec le nom de famille comme clé de jointure
df_atp_1617	dataframe issue de df_atp_encodage filtré sur les années 2016 et 2017
data_1617, target_1617	ensemble train/test issue du df_atp_1617 (pour modélisation sur les années 2016 2017)
df_atp_1617ST	dataframe issue de df_atp_encodage filtré sur les années 2016 et 2017 avec ajout des statistiques scrapées
data_1617ST, target_1617ST	ensemble train/test issue du df_atp_1617 avec ajout des statistiques scrapées(pour modélisation sur les années 2016 2017)
df_atp_1617ST_ MB	dataframe issue de df_atp_encodage filtré sur les années 2016 et 2017 avec ajout des statistiques scrapées et application de la moyenne mobile
data_1617ST_M B, target_1617ST_ MB	ensemble train/test issue du df_atp_1617 avec ajout des statistiques scrapées(pour modélisation sur les années 2016 2017) et application de la moyenne mobile

Définition des variables du site: <https://www.ultimatetennisstatistics.com/>

Variables	Définitions
Ace	Pourcentage de service gagnant sans que l'autre joueur ne touche la balle
Double faute	Pourcentage de double faute effectué par le joueur
1st Serve	Pourcentage de premier service passé par le joueur
1st Serve Won	Pourcentage de points gagnés durant le premier service
2nd Serve Won	Pourcentage de points gagnés durant le deuxième service
Break Points Saved	Pourcentage de balle de break sauvé
Service Points Won	Pourcentage de point gagné durant le jeu de service du joueur
Service Games Won	Pourcentage de jeux de service gagné
Ace against	Pourcentage d'ace que le joueur s'est pris
Double Fault Against	Pourcentage de double faute effectué par le joueur adverse
1st Srv Return Won	Pourcentage de point gagné sur le premier service de l'adversaire
2nd Srv Return Won	Pourcentage de point gagné sur le deuxième service de l'adversaire
Break Points Won	Pourcentage de balle de break gagné
Return Points Won	Pourcentage de point gagné en retour de service
Return Games Wons	Pourcentage de jeux de retour de service gagné
Points Dominance	Pourcentage de points où le joueur a été dominant
Games Dominance	Pourcentage de jeux où le joueur a été dominant
Break Points Ratio	Ratio de balle de break
Total Points Won	Pourcentage de points gagnés
Games Won	Pourcentage de jeux gagnés
Sets Won	Pourcentage de sets gagnés
Matches Won	Pourcentage de matchs gagnés
Match Time	Durée moyen du match (format hh

Bibliographie / référence internet

- <https://anj.fr/marche-2021-tres-forte-dynamique-pour-les-paris-sportifs-en-ligne-croissance-de-la-loterie>
- <https://parieztennis.fr/comparatif-bookmakers/>
- <https://www.kaggle.com/code/edouardthomas/beat-the-bookmakers-with-machine-learning-tennis/notebook>
- [Tennis results and betting odds data for tennis betting system development \(tennis-data.co.uk\)](https://tennis-data.co.uk/)
- <https://towardsdatascience.com/making-big-bucks-with-a-data-driven-sports-betting-strategy-6c21a6869171>
- https://www.researchgate.net/publication/331218530_Exploiting_sports-betting_market_using_machine_learning
- <https://www.pinnacle.com/fr/betting-articles/Tennis/us-open-2016-atp-rankings-elo-rating/78DJVFQJ6FYJBGBU>
- https://fr.wikipedia.org/wiki/Cat%C3%A9gorisation_des_tournois_de_tennis#Depuis_1990
- <https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc>
- <https://www.data-transitionnumerique.com/selenium-python/>
- [Touchez la meilleure cote pour vos paris tennis \(cotesports.fr\)](https://cotesports.fr/)
- <https://selenium-python.readthedocs.io/>
- <https://pandas.pydata.org/>

Annexes

- Diagramme de gantt:

[Diagramme de Gantt - projet bookmakers](#)

- Lien vers les données source Kaggle:

<https://www.kaggle.com/datasets/edouardthomas/atp-matches-dataset>

- Description des fichiers de code:

- Google Colab (audit des données, data viz, modélisation, interprétabilité)
Paris_sportif_Beat_the_Bookmakers :

https://drive.google.com/file/d/1jBAKHJbDzxNxOIhliuVogRwdJlwFtUIV/view?usp=share_link

- Jupyter notebook (webscraping site internet ultimatetennis)

https://drive.google.com/file/d/1eWbfhsilEN2LHcuYxJBBG9plQOiPBcdR/view?usp=share_link

- Fichiers csv / excel après audit et nettoyage des données:

[df_atp_final](#)

- Fichier d'exploration des données:

[DA_bookmakers- Rapport exploration des données](#)

- Site internet de web scraping:

<https://www.ultimatetennisstatistics.com/>