OCT 22
BOOTCAMP
DS

Papy sportifs

COMMENT BATTRE LES BOOKMAKERS SUR DES MATCHS DE TENNIS ?

CUKROWICZ ILAN – DELAHAYE VALENTIN – HAMCHAOUI NAWAL – SEGHIR OUSSAMA

SOUS LA SUPERVISION DE SAHEB ZAKARY

Table des matières

1. Introduction

Le pari sportif est un jeu d'argent sur la prédiction d'un événement sportif. Il existe de nombreux types de paris, dont les plus simples consistent à parier sur la victoire d'une équipe ou d'un sportif. Chaque pari a une côte et c'est elle qui permet de connaître à l'avance, en fonction de la somme mise, le montant gagné si l'événement se produit.

En notation décimale, si on mise 10 € sur l'équipe A dont la côte est de 1,2 par exemple :

- Si A gagne, la somme récupérée (gain) est de 12 €, soit une plus-value de 2 € (+ 20 %)
- Si A ne gagne pas, la somme misée est perdue, soit 10 €.

Les paris sont proposés par des bookmakers qui sont des organismes autorisés à proposer aux joueurs de parier. En français et avec l'essor des jeux d'argent sur internet, le terme de bookmaker fait généralement référence au site de paris en ligne.

Le 10 juin 2010, l'Autorité de régulation des jeux en ligne (ARJEL, remplacée par l'Autorité nationale des jeux (ANJ) en 2019) a été créée à la suite de la loi relative à l'ouverture à la concurrence des jeux d'argent et de hasard en ligne. Son rôle est de réguler le marché et de délivrer des agréments officiels aux opérateurs sur le territoire français.

Les paris sportifs étaient interdits en France jusqu'à cette date, la veille de l'ouverture de la Coupe du monde de football en Afrique du Sud.

Plus rien n'arrête l'expansion du marché, très dynamique, des paris sportifs depuis, et chaque parieur n'a qu'un objectif en tête : battre les bookmakers.

2. Présentation du projet

Notre projet s'inscrit dans le cadre de la formation de data scientist proposée par l'organisme Datascientest.

Le projet est porté par un groupe de quatre personnes : Ilan CUKROWICZ, Valentin DELAHAYE, Nawal HAMCHAOUI et Oussama SEGHIR, sous l'encadrement de Zakary Saheb, data scientist chez datascientest et sous le nom de « **Papy sportif** ».

L'objectif de ce projet est de « battre les bookmakers ». Pour ce faire, nous avons décidé d'élaborer la stratégie suivante :

- Explorer la base de données initiale
- L'enrichir
- L'épurer de toutes variables qui n'apportent pas suffisamment d'informations
- Tester différents modèles de prédiction
- Estimer l'espérance de gain
- Répondre à la question : Avons-nous battu les bookmakers ?

3. Niveau d'expertise et motivations de chacun des membres concernant le choix du projet

Ilan CUKROWICZ : Bonne connaissance de l'industrie des paris sportifs (ancien account manager chez Meta de plusieurs acteurs comme Winamax, ZeBET, Ruedesjoueurs.com…) et joueur occasionnel.

Valentin DELAHAYE : J'ai pratiqué le tennis en club pendant 8 ans durant ma jeunesse. J'ai une bonne connaissance de ce sport et de ses caractéristiques. Je continue à y jouer de temps en temps. Je regarde Roland Garros tous les ans depuis mon plus jeune âge. En ce qui concerne les bookmakers, je connais leur principe de fonctionnement pour avoir déjà expérimenté le pari en ligne.

Nawal HAMCHAOUI : Choisir ce projet était pour moi un défi, une aventure : je ne connais ni le monde du tennis, ni le monde des paris sportifs. En revanche, je suis de nature curieuse et aime apprendre de nouvelles choses. J'aime avoir plusieurs cordes à mon arc et repousser les limites de mes connaissances. L'avantage, ici, était de pouvoir, de surcroît, mettre directement en pratique les connaissances nouvellement acquises.

Oussama SEGHIR : Plusieurs projets nous ont été proposés. Mon choix n'était pas trop difficile car je suis un amateur de sports en général et en tennis particulièrement et j'ai beaucoup d'informations à propos de ce sport qui m'aideront certainement avec le projet, En outre, le pari sportif est une zone qui est en constante évolution et donc ce sera un atout majeur dans nos CV, et pourquoi ne pas rejoindre une entreprise de paris sportifs.

4. Présentation du dataset

Nous avons à disposition deux bases de données nommé **atp_data** et **confidence_data** qui sont disponibles librement sur : https://www.kaggle.com/code/edouardthomas/beat-the-bookmakers-with-machine-learning-tennis/data.

Nous avons décidé de ne pas prendre en considération confidence_data et de nous concentrer sur atp_data.

Ce dernier compile des données disponibles sur le site : Tennis results and betting odds data for tennis betting system development et couvre la période allant de janvier 2000 à mars 2018, à l'exception des années 2001 et 2014 pour lesquelles aucune donnée n'est présente.

Nous avons cherché à comprendre l'information contenue derrière chaque nom de variables en premier lieu, puis leur distribution et enfin, les relations « primaires » qu'elles peuvent avoir entre elles, tout ceci à l'aide de tableaux et de graphes.

A. Prise en main du dataset

La première étape est de faire une exploration du jeu de données et visualiser ses différentes variables.

Avant toute chose, il convient de regarder un aperçu du jeu de données ainsi qu'un résumé à l'aide la fonction info() :

	ATP	Location	Tournament	Date	Series	Court	Surface	Round	Best of	Winner	...	Wsets	Lsets	Comment	PSW	PSL	B365W	B365L	elo_winner	elo_loser	proba_elo
0	1	Adelaide	Australian Hardcourt Championships	2000-01-03	International	Outdoor	Hard	1st Round	3	Dosedel S.	...	2.0	0.0	Completed	NaN	NaN	NaN	NaN	1500.0	1500.0	0.5
1	3	Doha	Qatar Open	2000-01-03	International	Outdoor	Hard	1st Round	3	Kiefer N.	...	2.0	0.0	Completed	NaN	NaN	NaN	NaN	1500.0	1500.0	0.5
2	3	Doha	Qatar Open	2000-01-03	International	Outdoor	Hard	1st Round	3	Gaudio G.	...	2.0	1.0	Completed	NaN	NaN	NaN	NaN	1500.0	1500.0	0.5
3	3	Doha	Qatar Open	2000-01-03	International	Outdoor	Hard	1st Round	3	El Aynaoui Y.	...	2.0	1.0	Completed	NaN	NaN	NaN	NaN	1500.0	1500.0	0.5
4	3	Doha	Qatar Open	2000-01-03	International	Outdoor	Hard	1st Round	3	Cherkasov A.	...	2.0	0.0	Completed	NaN	NaN	NaN	NaN	1500.0	1500.0	0.5

Aperçu sur le jeu de donnée

```
<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
None
```

Le jeu de données contient 44708 lignes et 23 colonnes. Nous comprenons que chaque ligne correspond à un match de tennis opposant 2 joueurs et que les colonnes représentent certaines caractéristiques de ces matchs. Nous notons la présence :

- de données manquantes
- de variables numériques et catégorielles
- d'une série temporelle

En regardant, plus en détail, on comprend également quelles informations sont contenues dans les variables. Le tableau ci-dessous décrit ces différentes variables :

Tableau explicatif de chaque variable

La variable	La signification
ATP	Tournament number (men)
Location	Venue of tournament
Tournament	Name of tournament (including sponsor if relevant)
Data	Date of match (note : prior to 2003 the date shown for all matches)
Séries	Name of ATP tennis series (Grand Slam, Masters, International or International Gold)
Court	Type of court (outdoors or indoors)
Surface	Type of surface (clay, hard, carpet or grass)
Round	Round of match
Best of	Maximum number of sets playable in match
Winner	Match winner
Loser	Match loser
W Rank	ATP Entry ranking of the match winner as of the start of the tournament

LRank	ATP Entry ranking of the match loser as of the start of the tournament
Wsets	Number of sets won by match winner
Lsets	Number of sets won by match loser
Comment	Comment on the match (Completed, won through retirement of loser, or via Walkover)
B365W	Bet365 odds of match winner
B365L	Bet365 odds of match loser
PSW	Pinnacles Sports odds of match winner
PSL	Pinnacles Sports odds of match loser
elo winner	cote elo pour le gagnant
elo_loser	cote elo pour le perdant
proba elo	Probabilité de victoire avec la méthode elo

Nous avons donc 7 variables numériques, 15 catégorielles et une série temporelle.

Pour plus de détails, se référer au fichier Excel « **PaPy Sportifs** - Rapport exploration des données 1.xlsx ».

Dans un second temps, nous avons cherché à connaître le contenu des variables, à l'aide d'un describe, sur les variables numérique mais aussi sur les variables catégorielle.

	PSW	PSL	B365W	B365L	elo_winner	elo_loser	proba_elo
count	32743.000000	32743.000000	39037.000000	39057.000000	44708.000000	44708.000000	44708.000000
mean	1.927563	4.240179	1.822246	3.551007	1684.023280	1608.755552	0.585594
std	1.359136	5.744329	1.107547	3.498689	179.246422	137.949299	0.198732
min	1.000000	1.010000	1.000000	1.002000	1318.945207	1327.551888	0.008899
25%	1.270000	1.794000	1.220000	1.720000	1548.476977	1501.546103	0.447921
50%	1.549000	2.660000	1.500000	2.500000	1652.866073	1580.743828	0.589218
75%	2.140000	4.270000	2.000000	4.000000	1780.755524	1684.607134	0.737288
max	46.000000	121.000000	29.000000	101.000000	2392.408923	2392.595567	0.995601

Describe sur variables numériques

Nous pouvons remarquer que l'ordre de grandeur des variables et leurs « caractéristiques » diverge de 4 degrés, ce qui sous-tend la nécessité de centrer et réduire nos variables avant toute modélisation.

	ATP	Location	Tournament	Series	Court	Surface	Round	Best of	Winner	Loser	WRank	LRank	Wsets	Lsets	Comment
count	44708	44708	44708	44708	44708	44708	44708	44708	44708	44708	44708	44708	44521.0	44521.0	44708
unique	69	115	207	8	2	4	8	2	899	1400	587	920	4.0	3.0	4
top	6	Paris	Australian Open	International	Outdoor	Hard	1st Round	3	Federer R.	Lopez F.	1	46	2.0	0.0	Completed
freq	2159	2784	2159	10792	36532	23799	20728	36379	970	369	1007	393	35235.0	27976.0	43015

Describe sur variables catégorielles

Nous pouvons remarquer que le nombre de modalité est très différent d'une variable à l'autre, allant de 2 pour les variables 'Court' et 'Best of' jusqu'à 1400 pour la variable '**Loser**'. S'il était plus aisé de dichotomiser les variables aux nombres de modalité moins élevées, il serait

nécessaire de trouver un autre moyen pour conserver l'information utile des autres variables (au nombre de modalité très élevées).

B. Distribution des variables

a) Valeurs manquantes et nettoyage du dataset

Après avoir vérifié que le jeu de données ne contenait pas de doublons, nous nous sommes concentrés sur les valeurs manquantes.

La variable '**Comment**' a une importance particulière puisqu'elle indique si le match s'est déroulé « normalement » (modalité 'Completed') ou si des « incidents » ont perturbé le match (modalités 'Retired', 'Walkover', 'Disqualified'). Les valeurs manquantes de cette variable donnaient l'information que le match avait été annulé (pour quelques raisons que ce soit) et par conséquent, les paris le concernant également. Nous avons donc décidé de simplement supprimer les entrées du jeu de données dans lesquelles la valeur de la variable 'Comment' était manquante.

Concernant les autres valeurs manquantes, l'illustration ci-dessous nous montre qu'elles concernent certaines variables et qu'elles sont localisées. Il s'agit essentiellement des côtes les plus récentes des bookmakers '**Pinnacle**' et '**Bet 365**'. L'objectif sera donc de retrouver ces cotes en ligne et enrichir la base de données.

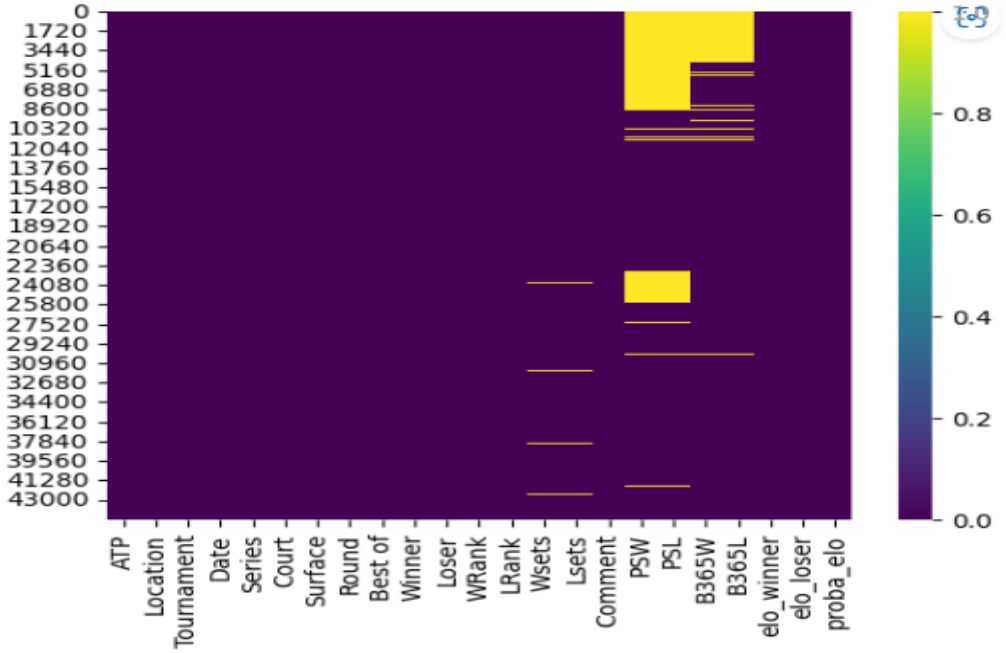


Figure 1 : cartographie des valeurs manquantes

Toujours dans l'objectif de mieux comprendre notre dataset, nous avons procédé à plusieurs visualisations.

b) Visualisations

Par souci de clarté et afin de ne pas rendre indigestible ce rapport nous avons sélectionné les visualisations qui nous paraissaient les plus pertinentes.

Tout d'abord, qui sont les joueurs qui gagnent le plus de match ? Ci-dessous, l'histogramme qui indique les 10 joueurs qui ont remporté le plus de victoires (avec le nombre de victoires).

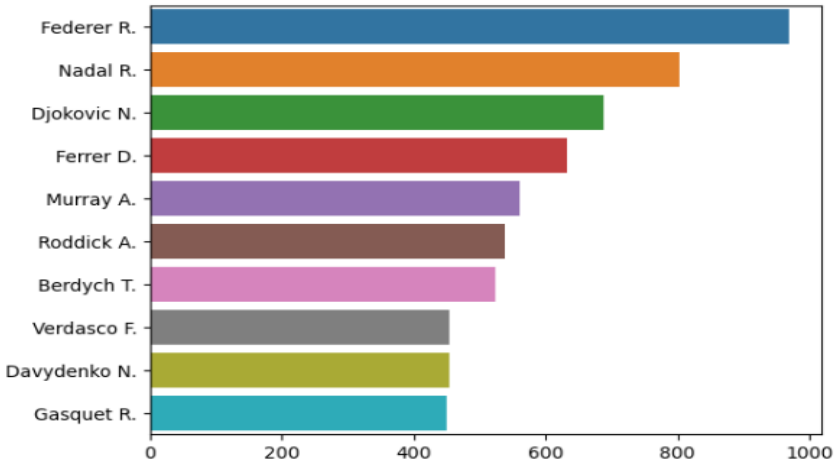


Figure 2 : histogramme des 10 joueurs ayant remporté le plus de victoires

Sans grande surprise, nous retrouvons les grands noms du tennis, comme Federer, Nadal ou Djokovic. Ce qui appuie la cohérence de notre jeu de données.

Nous avons également porté notre attention sur les côtes des bookmakers, qui ont toutes leurs importances ici, puisque c'est elles qui nous permettent des gains financiers ou non.

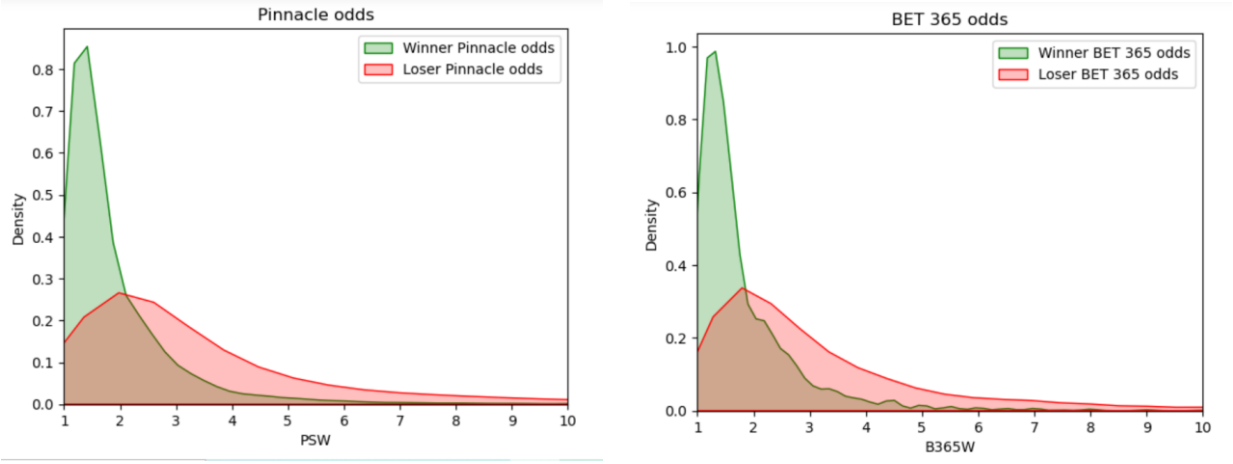


Figure 3 : Distribution des cotes des bookmakers

Un simple coup d'œil permet de voir une distribution suivant une loi de Poisson avec des paramètres différents relativement au gagnant ou perdant. Cependant les courbes des différents bookmakers ont la même allure, bien que les densités soient différentes. Cette

différence peut s'expliquer par la marge prise par chacun des bookmakers (elle est plus petite chez Pinnacle).

Sans surprise, la distribution des cotes des gagnants sont comprises essentiellement entre 1 et 2 (plus de 70% d'entre elles), tandis que celles des perdants ont une plus grande amplitude (à peine plus de 30% des côtes sont comprises entre 1 et 2, 25% entre 2 et 3 et environ 15% entre 3 et 4). Il est à noter que certaines côtes peuvent atteindre 10, parfois plus.

Nous pouvons ainsi mettre en avant différentes visualisations concernant le comportement d'autres variables. Néanmoins, nous allons maintenant présenter certaines relations entre variables.

c) Relations entre variables

Lorsque nous avons pensé aux relations entre les variables, la première chose qui nous soit venue à l'esprit est la matrice de corrélation. Elle nous permet de voir la mesure de l'information qu'une variable peut partager avec une autre, l'information redondante en d'autres termes. Ainsi, on observe, tout naturellement que les cotes des gagnants (PSW et B365W) et les côtes (PSL et B365L) des perdants sont fortement corrélées.

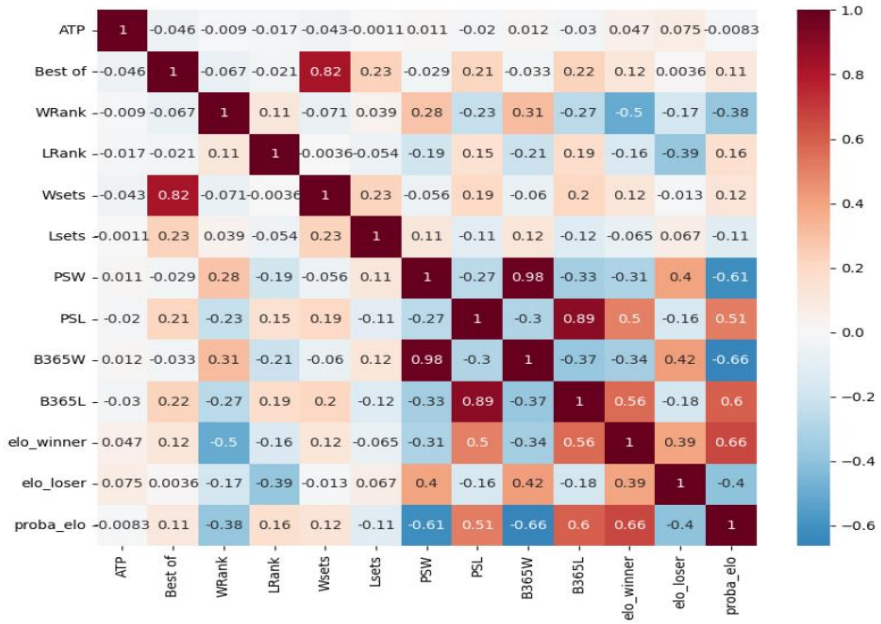


Figure 4 : Matrice de corrélation

En corrigeant les côtes de la marge que les bookmakers y introduisent, on se rend compte que la corrélation est maximale à 1/0.99 ou -1/-0.99 (figure ci-dessous). En d'autres termes, ces variables (côtes corrigées des différents bookmakers) portent la même information. Ce qui n'est pas surprenant lorsque l'on sait les moyens déployés par ces bookmakers afin d'être le plus juste dans leurs prédictions.

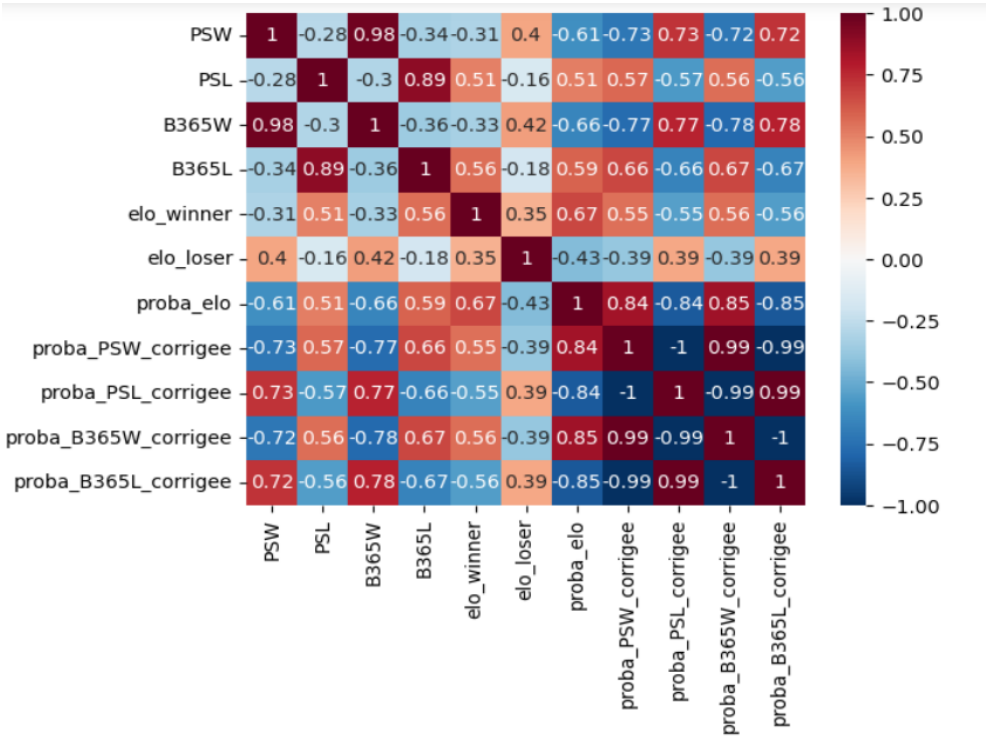


Figure 5 : Matrice de corrélation avec cotes corrigées

La marge fixée par les bookmakers dans les côtes qu'ils proposent est donc un facteur déterminant à prendre en compte dans la stratégie à mettre en œuvre afin de battre ces bookmakers. Ainsi, nous avons extrait ces marges et visualiser sa distribution dans la figure ci-dessous.

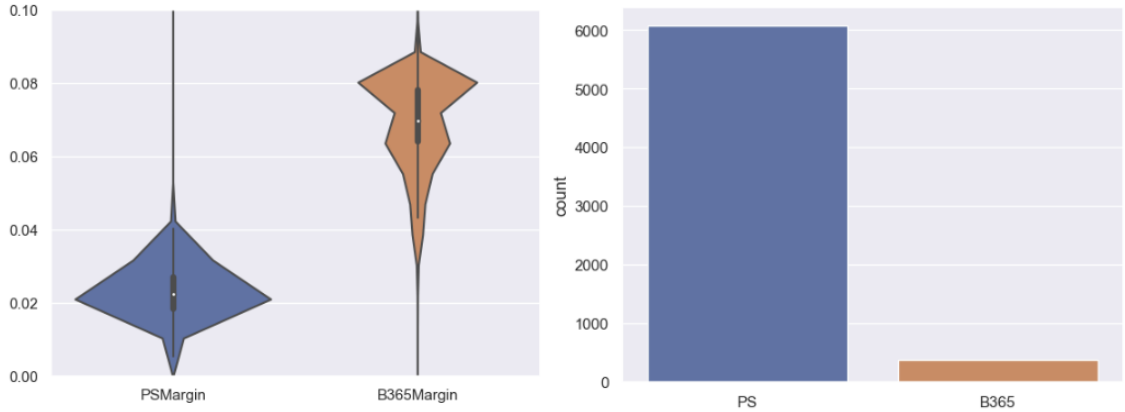


Figure 6 : Répartition de la marge en comparaison au nb de côtes uniques par bookmake

On constate que le bookmaker **B365** prend une marge nettement plus élevée sur l'ensemble des côtes avec une forte densité autour de 8% contre 2% pour Pinacle. On remarque en parallèle que le nombre de côtes uniques est nettement supérieur chez **Pinacle** que chez **B365** (Il y a donc plus de variabilité chez Pinnacle).

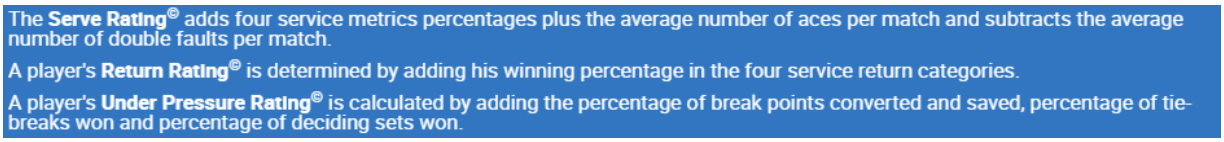
5. Enrichissement du dataset

Suite à la première étape d'exploration, nous avons cherché un moyen d'enrichir notre jeu de données avec les statistiques propres à chaque match et à chaque joueur. Nous avons identifié 2 moyens de le faire :

1) Le jeu de données atp_data, téléchargé à partir du lien suivant : ATP matches dataset | Kaggle, est une extraction des données disponible sur le site : http://tennis-data.co.uk/data.php. Il a été possible de télécharger les données les plus à jour : les matchs de mars 2018 (là où s'arrêtait le dataset Kaggle) à nos jours.
2) Le GitHub suivant réalise le scraping du site ATP Tour : Using Python to scrape ATP World Tour tennis data. Il permet d'extraire les données propres à chaque tournoi, à chaque match et à chaque joueur.

Nous avons donc procédé aux téléchargements des fichiers '.csv' disponibles sur le GitHub de scraping, puis, concaténé toutes les données de 2000 à 2022 et réalisé des jointures sur les deux clés primaires '**tourney_year_id**' et '**match_id**'.
Nous avons ensuite procédé à la jointure du résultat des étapes précédentes avec le dataset Kaggle actualisé avec les matchs les plus récents (de 2018 à nos jours). Le dataset qui résulte de tout cela est composé de 46216 lignes et 220 colonnes.

Nous avons, par la suite, procédé à une phase d'exploration et de nettoyage du dataset pour supprimer les informations redondantes ainsi que les valeurs manquantes. Nous avons également créé trois statistiques définis par le site ATP Tour :



Le résultat des phases précédentes nous donne un dataset de 28666 lignes et 92 colonnes. Pour rappel, une ligne correspond à un match où nous avons les informations relatives aux joueurs gagnant et perdants ainsi que les informations communes aux deux joueurs propres au match et au tournoi.

De plus, les observations de notre dataset étant les résultats de chaque match et les résultats étant censés être ignorés avant match, nous avons apporté de nouvelles modifications à notre dataset :

1) Calcul de la moyenne mobile
    a) Séparer les statistiques de chaque adversaire (les informations communes étant dédoublées pour chacun des deux joueurs) afin d'obtenir un jeu de données une entrée ne contient que les informations relatives à un joueur.
    b) Trier les données par joueurs selon le temps
    c) Calculer la moyenne mobile des statistiques du joueurs (afin de prendre en compte son historique, sa forme lors des modélisations, la fenêtre de cette moyenne étant un hyper-paramètres à déterminer)

d) Déplacer ces nouvelles statistiques au match jouer juste avant (car elles prennent en compte les résultats du match en cours)
e) Re-former le tableau initial avec en entrée les matchs et statistiques des deux adversaires.

2) Création de la variable cible en tirant aléatoirement l'identité des joueurs 1 et 2. Notre dataset contenant les statistiques des gagnant et perdant, nous avons ainsi créé une nouvelle variable cible : '**player_1_win**', qui prend la valeur de 1 si le player_1 est gagnant ou 0 s'il est perdant.

A ce stade, le dataset est de dimension (28666, 100), correspondants aux matchs de 2004 à 2019 (de 2019 à nos jours, nous n'avons pu scrappé que des valeurs manquantes, par manque de temps, nous avons préféré tronquer notre dataset) avec les variables suivantes :

Variables communes

La variable	La signification
year	Année du match
month	Mois du match
day	Jour du match
Series	Catégorie du tournoi (Grand Slam, Masters, International or International Gold…)
Court	Terrains de tennis en intérieur ou en extérieur
Surface	Surface du terrain (clay, hard, carpet or grass)
prize_money_euro	la dotation financière du tournoi
Round	Tour du match (1er tour, 2ème tour… quart de finale, demi-finale, finale)
Best of	Le nombre de sets maximum (3 ou 5)

Variables propres à chacun des 2 joueurs (player_1_*variable* et player_2_*variable*)

La variable	La signification
B365	Cote du bookmaker B365
PS	Cote du bookmaker Pinnacle
name	Prénom et Nom du joueur
rank	Classement ATP du joueur au début du tournoi
height_cm	Taille en cm
weight_kg	Poids en kg
handedness	Gaucher ou droitier
backhand	Revers 1 main ou 2 mains
at_home	Joueur joue à domicile (dans son pays) ou pas (1 ou 0)
age	Âge
nb_year_xp	Nombre d'années d'expérience
match_duration	Durée moyenne par match (Moyenne mobile - MA)
nb_sets	Nombre de sets disputés par match (MA)
nb_games	Nombre de jeux disputés par match (MA)
nb_tiebreaks	Nombre de tie breaks disputés par match (MA)

nb_points	Nombre de points disputés par match (MA)
sets_won	% de sets gagnés par match (MA)
games_won	% de jeux gagnés par match (MA)
tiebreaks_won	% de tie breaks gagnés par match (MA)
points_won	% de points gagnés par match (MA)
serve_rating	Somme des statistiques liés au service (MA)
aces	Nombre d'aces par match (MA)
double_faults	Nombre de double fautes par match (MA)
first_serves_in	% de bon 1er service (MA)
first_serves_total	Nb de 1er service (MA)
first_serve_points_won	% de points gagnés découlant d'un bon 1er service (MA)
first_serve_points_total	Nb de points disputés découlant d'un bon 1er service (MA)
second_serve_points_won	% de points gagnés découlant d'un bon 2ème service (MA)
second_serve_points_total	Nb de points disputés découlant d'un bon 2ème service (MA)
break_points_saved	% de break points sauvés sur le service du joueur (MA)
break_points_serve_total	Nb de break points joués sur le service du joueur (MA)
service_games_played	Nb de jeux de service joués (MA)
return_rating	Somme des statistiques liés au retour de service (MA)
first_serve_return_won	% de points gagnés découlant d'un retour sur 1er service (MA)
first_serve_return_total	Nb de points disputés découlant d'un retour sur 1er service (MA)
second_serve_return_won	% de points gagnés découlant d'un retour sur 2ème service (MA)
second_serve_return_total	Nb de points disputés découlant d'un retour sur 2ème service (MA)
break_points_converted	% de break points gagnés en retour de service (MA)
break_points_return_total	Nb de break points joués en retour de service (MA)
return_games_played	Nb de jeux joués en retour de service (MA)
service_points_won	% de points gagnés en service (MA)
service_points_total	Nb de points joués en service (MA)
return_points_won	% de points gagnés en retour de service (MA)
return_points_total	Nb de points joués en retour de service (MA)
under_pressure_rating	Somme de break_points_saved, break_points_converted et tie_breaks_won (MA)
win_rate	% de matchs gagnés (MA)
player_1_win	variable cible (1 ou 0)

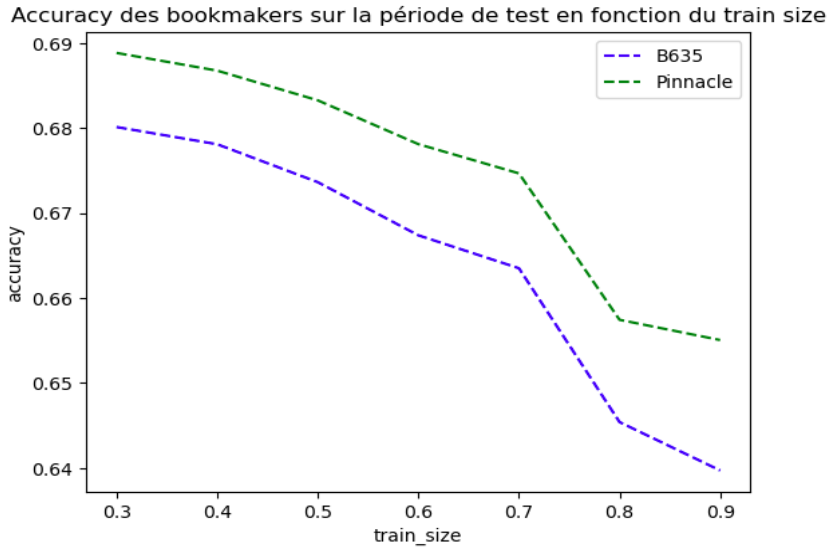
Enfin, avant de lancer les modélisation, nous avons eu recours à de nouvelles étapes de pré-processing :

- Encoder '**player_1_name**' et '**player_2_name**'
- Dichotomiser les variables catégorielles
- Séparer le dataset en données d'entraînement et de test tout en respectant la chronologie des matchs : les données de test correspondent aux matchs les plus récents, les données d'entraînement aux matchs les plus anciens.
- Normaliser et standardiser les données d'entraînement et de test

6. Modélisations

Pour rappel, notre projet est un problème de classification binaire : prédire si le joueur 1 est gagnant ou non. La variable cible ayant été construite sur la base d'un tirage au sort (50% player_1_win = 1 et 50% player_1_win = 0), notre dataset est parfaitement équilibré.

La métrique de performance principale utilisée pour comparer nos modèles est l'accuracy qui correspond au taux de bonnes prédictions du modèle. Nous pouvons comparer cette métrique aux taux de bonnes prédictions des bookmakers (book_accuracy). Nous disposons des cotes établies par les bookmakers B365 et Pinnacle sur les 28 666 matchs du dataset final. La book_accuracy de B365 sur l'ensemble des 28 666 matchs est de 68.06%, celle de Pinnacle est de 69.01%. Voici l'évolution de la book_accuracy du jeu de test (période la plus récente) en fonction de la taille du jeu d'entraînement :



Pour rappel, les données de test correspondent aux matchs les plus récents et les données d'entraînement aux matchs les plus anciens.

A l'observation de ce graphe, nous pouvons constater d'une part que plus le jeu d'entraînement est grand, moins bonne est l'accuracy, et d'autre part que le bookmaker Pinnacle obtient une meilleure prédiction que B365. Ces constats nous permettent d'émettre l'hypothèse suivante : les modèles des bookmakers perdent de leurs capacités à généraliser lorsque l'on prend des périodes trop grandes pour l'entraînement. On peut aisément penser que ceci est dû à une tendance présente dans notre jeu de données : le niveau global des joueurs augmente avec le temps et les moyens alloués également. On ne joue pas au tennis comme il y a 30 ans. Avec le temps, les matchs sont plus serrés et l'issue d'un match est plus incertaine.

Par souci de simplification, nous avons décidé que le jeu d'entraînement fera 70% de l'ensemble des données pour chacune de nos modélisations. Nous répondrons à la question : 'Avons-nous battu les bookmakers ?' par l'affirmative, si nous parvenons à dépasser les

accuracy de nos deux bookmakers calculées à partir du jeu de test (**30%** de l'ensemble des données), soit 8600 matchs, à savoir :

- **66.35% pour B365 et,**
- **67.47% pour Pinnacle.**

Nous avons répartis les principaux modèles de classification entre les différents membres de l'équipe. La stratégie de modélisation est la suivante :

- Modélisations naïves
- Optimisation des hyperparamètres
- Modélisation d'un Voting Classifier avec les meilleurs modèles
- Modèle de deep learning

A. Premières modélisations naïves

Afin de mesurer l'impact de (1) l'enrichissement du dataset et (2) l'influence des bookmakers, nous avons procédé à des modélisations naïves (sans aucune paramétrisation) sur trois datasets :

- un dataset avant enrichissement (dataset_1)
- un dataset après enrichissement **sans** les cotes des bookmakers avec différentes moyennes mobiles [10, 30, 50, 100, 150, 200] (dataset_2)
- un dataset après enrichissement **avec** les cotes des bookmakers avec différentes moyennes mobiles [10, 30, 50, 100, 150, 200] (dataset_3)

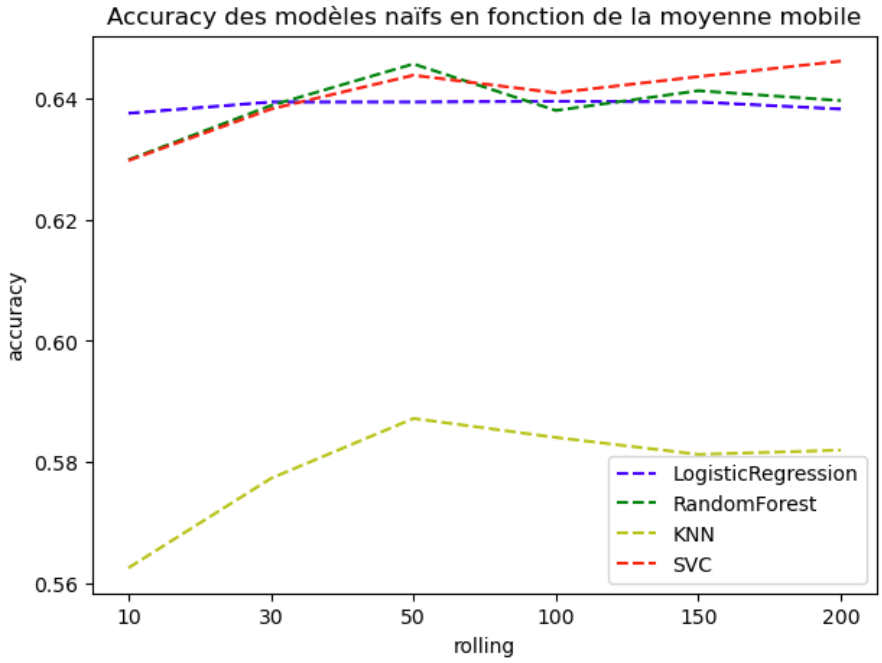
Nous avons entraîné 4 modèles de classification : KNN, Random Forest (RF), Régression Logistique (LR) et Support Vector Classifier (SVC), sans aucune paramétrisation. Voici les accuracy obtenues sur test :

Modèle	LR	RF	KNN	SVC
dataset_1	62.49%	62.76%	55.71%	61.58%
dataset_2	63.95% (MA: 100 matchs)	64.57% (MA: 50 matchs)	58.72% (MA: 50 matchs)	64.62% (MA: 200 matchs)
gain_enrichissement (dataset_2 - dataset_1)	*1.46%*	*1.81%*	*3.01%*	*3.04%*
dataset_3	67.06% (MA: 10 matchs)	**67.21%** (MA: 100 matchs)	59.03% (MA: 100 matchs)	66.7% (MA: 200 matchs)
gain_bookmakers (dataset_3 - dataset_2)	*3.11%*	*2.64%*	*0.31%*	*2.08%*

Nous pouvons constater que l'enrichissement du dataset grâce au webscraping du site officiel ATP Tour a permis de gagner entre 1.46% et 3.04% en fonction des modèles. Sans surprise, les gains lorsque nous intégrons les côtes des bookmakers, sont conséquents (+3.11% pour RF). A ce stade, la **meilleure accuracy obtenue est de 67.21%** avec le modèle RF. Nous remarquons que les modélisations sur le dataset_3 donnent des **résultats déjà très encourageants** puisque les **accuracy obtenues pour LR, RF et SVC dépassent celle du bookmaker B365**.

Les accuracy obtenues sur le dataset_3 serviront d'éléments de comparaison pour constater la progression dans la définition de nos futurs modèles.

Afin de limiter le nombre de modélisations à effectuer, nous cherchons à garder une seule valeur pour la moyenne mobile. Ce choix a été fait grâce au graphique des accuracy sur le dataset_2 :



Nous remarquons que l'accuracy en fonction de la moyenne mobile est relativement stable pour les modèles LR, RF et SVC. L'accuracy obtenue est légèrement plus élevé pour une moyenne mobile sur 50 matchs pour les modèles RF, SVC et KNN. Nous garderons donc cette valeur là pour les futures modélisations.

B. Optimisation des hyperparamètres

Nous avons procédé à la recherche des hyperparamètres donnant les meilleurs résultats pour l'ensemble des méthodes de classification suivantes : Régression Logistique (LR), Random Forest (RF), KNeighbors Classifier (KNN), Support Vector Classifier (SVC), Gradient Boosting Classifier (GBC) et XGBoost (XGB).

Définition des grilles d'hyper-paramètres:

- LR:[{'solver': ['lbfgs'], 'C': [0.001, 0.005, 0.01, 0.05, 0.1, 0.5], 'penalty':['l2']}, {'solver': ['liblinear'], 'C': [0.001, 0.005, 0.01, 0.05, 0.1, 0.5], 'penalty':['l1', 'l2']}]
- RF:{'n_estimators' : [50,75,100,200], 'criterion' : ['entropy','gini'], 'max_depth' : [3,4,5,6],'random_state' : [10]}
- KNN:{'n_neighbors': list(range(200, 1801, 200)), 'metric':['manhattan']}
- SVC: {'kernel': ['rbf', 'linear', 'poly', 'sigmoid'], 'C': [0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5], 'gamma': [0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 'scale', 'auto'], 'degree' : [2, 3]}
- GBC: {'n_estimators':range(40,101,20), 'learning_rate':[0.1, 0.2, 0.5], 'max_depth':[1, 3]}
- XGB: {"subsample":[0.5, 0.75, 1], "colsample_bytree":[0.75, 1], "max_depth":[2, 6, 12], "min_child_weight":[1,5,15], "learning_rate":[0.3, 0.1, 0.03], "n_estimators":[50, 75, 100]}

Cross-validation temporelle :

Pour rappel, nos données suivent un ordre chronologique: des matchs les plus anciens aux matchs les plus récents de 2004 à 2019. Il ne fait donc pas sens d'effectuer un tirage aléatoire des matchs pour constituer nos échantillons d'entraînement et de validation. Il faut garder cet ordre chronologique lors de la validation croisée. Nous avons donc choisi de réaliser une cross-validation temporelle à l'aide de la fonction TimeSeriesSplit qui sépare le jeu d'entraînement et celui de validation en gardant l'aspect temporel :

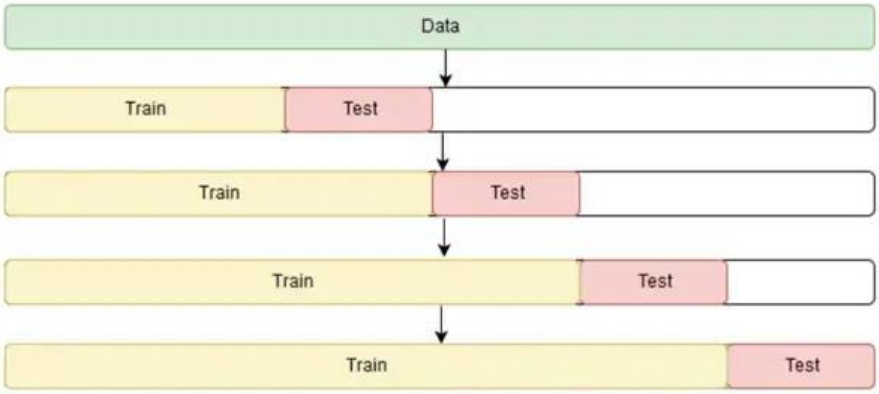


Schéma de la validation croisée temporelle

Nous avons choisi arbitrairement de séparer l'ensemble d'entraînement en 5 sous-ensembles entraînement-validation (argument n_split) : TimeSeriesSplit(n_split=5). Une amélioration possible aurait été d'intégrer cette variable dans l'optimisation des hyper-paramètres.

Voici les modèles ayant donnés les meilleurs résultats sur l'ensemble de test :

Modèle	LR	RF	KNN	SVC	GBC	XGB
Test_accuracy	**67.55%**	67.37%	65.77%	66.65%	67.42%	67.37%

Best_estimator	LogisticRegression(C=0.005, max_iter=2000, penalty='l1', random_state=42, solver='liblinear')	RandomForestClassifier(criterion='entropy', max_depth=5, n_estimators=75, random_state=10)	KNeighborsClassifier(metric='manhattan', n_neighbors=600)	SVC(gamma=0.005, kernel='rbf', random_state=42)	GradientBoostingClassifier(learning_rate=0.1,max_depth=1, n_estimators=60, random_state=0)	XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1, learning_rate=0.1, max_depth=2, min_child_weight=15,n_estimators=50,n_jobs=-1, ,random_state=0, subsample=1, tree_method='exact')

Le meilleur résultat est obtenu avec le modèle de Régression Logistique (67.55%). Ce dernier résultat est supérieur à la book accuracy des 2 bookmakers B365 (66.35%) et Pinnacle (67.47%). Les modèles RF, GBC et XGB donnent également des résultats très encourageants. Nous garderons ces 4 modèles pour construire un Voting Classifier.

C. Voting Classifier

Afin de se servir de la force de chacun des modèles, nous avons pensé à modéliser un Voting Classifier intégrant chaque modèle retenu précédemment lors de la phase d'optimisation des hyperparamètres.

Nous avons testé l'ensemble des combinaisons possibles pour la constitution du 'collège d'experts'. Nous avons également testé ces différentes combinaisons en appliquant les deux modes de vote 'hard' et 'soft. Voici les résultats :

Accuracy	LR+ RF+ GBC+ XGB	LR+ RF+ GBC	LR+ RF+ XGB	LR+ GBC+ XGB	RF+ GBC+ XGB	LR+ RF	LR+ GBC	LR+ XGB	RF+ GBC	RF+ XGB	GBC+ XGB
Vote 'hard'	67.44%	67.41%	67.48%	67.53%	67.40%	67.53%	**67.58%**	67.42%	67.34%	67.31%	67.35%
Vote 'soft'	67.43%	67.43%	67.48%	67.44%	67.28%	67.38%	67.49%	67.49%	67.22%	67.22%	67.42%

La meilleure accuracy obtenue est celle du Voting Classifier regroupant les modèles LR et GBC avec le mode de vote 'hard'. Elle s'élève à 67.58%, ce qui correspond à 5812 matchs de bonnes prédictions sur 8600 matchs du jeu de test contre 5802 pour le meilleur Bookmaker : Pinnacle.

D. Modèle Deep Learning

Nous avons testé plusieurs architectures de réseaux de Neurones pour évaluer le taux de bonnes prédictions grâce à un modèle de deep learning. Le meilleur score obtenu est de 67.17%. La quantité de données n'est sûrement pas assez élevée pour qu'un modèle de deep learning donne des résultats plus intéressants que les modèles de machine learning classiques.

7. Stratégie de paris

En faisant des prédictions de probabilités (méthode predict_proba), nous pouvons établir une stratégie par nos meilleurs modèles considérée comme rentable. Pour ceci, il suffit de calculer l'espérance de gain net (notée EV_net) de chaque côte comme ceci :

EV_net = côte * y_proba - 1

Dans chaque situation, nous avons choisi la côte la plus haute parmi celles proposées par les deux bookmakers (Pinacle et Bet365)

Si EV_net > 0 : Le pari est considéré comme rentable par le modèle.

Si EV_net < 0 : Le pari est considéré comme mauvais par le modèle.

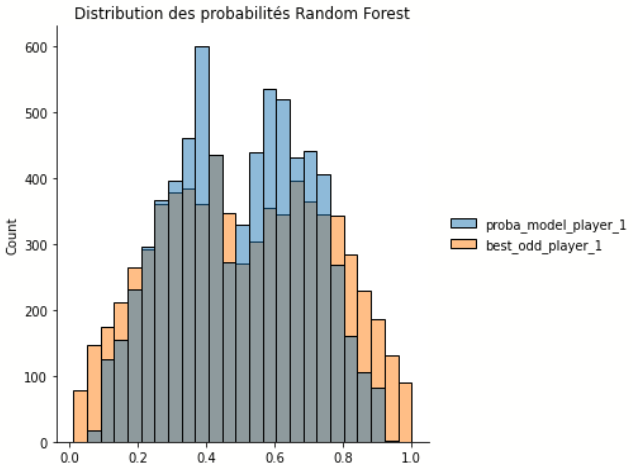
Notre stratégie consistera à parier la même somme (1€) dès lors que nous considérons judicieux de parier. Nous travaillerons sur l'ensemble de Test de taille 30%, soit 8600 matchs.

Les trois métriques clé que nous allons suivre pour choisir la meilleure stratégie sont :

- Le résultat net en € : L'objectif final est de trouver la stratégie qui gagne le plus d'argent sur le long terme
- Le volume de côtes retenues en % des côtes proposées : Plus le modèle est capable de parier sur un volume important, plus il sera considéré comme performant pour deux raisons :
    - Avoir un nombre important d'occurrences permet de converger plus rapidement vers son espérance de gain et donc de limiter la part de hasard inhérente aux paris sportifs.
    - Les sites de paris sportifs proposent différents systèmes de récompenses basés principalement sur le volume de paris réalisés.
- Le ROI (en %) : Cette métrique reste très importante dans la mesure où elle permet de situer l'intérêt d'investir de l'argent sur ce modèle en comparaison avec d'autres placements financiers.

A. Stratégie avec Random Forest

Nous obtenons un DataFrame contenant 7079 côtes considérées comme rentables. Sur ces matchs supposés rentables, on observe que la distribution des probabilités de victoires du modèle est plus dense au centre que la distribution des probabilités de victoires des bookmakers. L'interprétation concrète est que le modèle considère beaucoup de matchs comme plus "équilibrés" que les bookmakers, et va par conséquent avoir tendance à choisir des côtes élevées.



En pariant sur toutes les côtes considérées comme rentables par le modèle Random Forest, nous obtenons un bénéfice net de 107,02€ (+1,51% de rendement)

En misant uniquement sur les côtes rentables situées entre 2.15 et 6.95, nous avons maintenant 4570 côtes (53,14% des matchs) considérées comme rentables, et le bénéfice net est de 227.26€. Nous avons un rendement à 4,97%. Cependant, il est possible que le fait d'avoir trié en fonction des valeurs des côtes entraîne un biais : nous avons choisi ces côtes parce qu'on obtient ainsi le meilleur résultat.

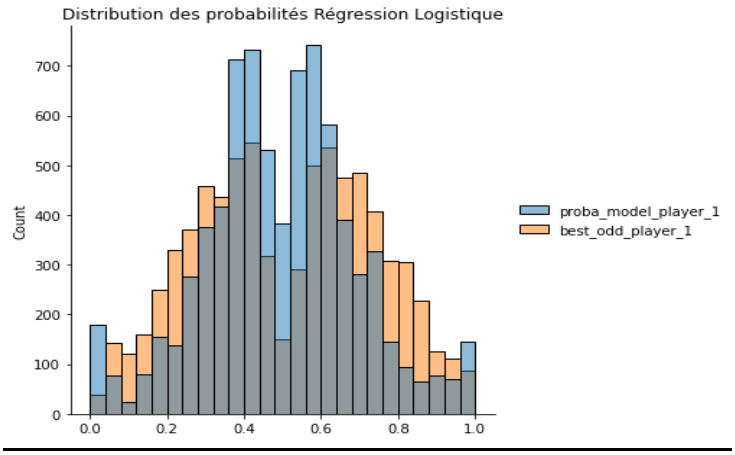
Si nous voulons nous assurer de n'avoir aucun biais, nous pouvons miser sur toutes les côtes que Random Forest considère comme rentables et garder un rendement de 1,51%, tout en étant capables de miser sur 82,3% des matchs proposés, assurant ainsi un large volume limitant le risque sur le court terme.

En essayant de jouer sur l'EV_net minimal requis pour obtenir de meilleurs résultats, nous obtenons une stratégie certes plus rentable, mais celle-ci nous limite fortement sur le nombre de matchs profitables.

Il est par exemple possible de miser sur les EV_net > 0.16, ce qui nous donne un bénéfice de 95.07€ sur l'ensemble de Test (sur 2464 matchs). Nous aurions alors un rendement de 3,86% applicable à 28,65% des matchs. De même, ce seuil est choisi arbitrairement.

B. Stratégie avec la Régression Logistique

Nous obtenons un DataFrame contenant 7691 côtes considérées comme rentables (89,4% des matchs). Sur ces matchs supposés rentables, on observe que la distribution des probabilités de victoires du modèle est plus dense au centre que la distribution des probabilités de victoires des bookmakers, mais également aux extrémités avec quelques rares cas où la Régression Logistique semble plus catégorique que les bookmakers sur l'issue d'un match.



En pariant sur toutes les côtes rentables, nous obtenons un bénéfice net de 83,31€, soit un rendement de 1,08%.

Si nous voulons optimiser ces résultats en filtrant par côtes, on se rend compte que la Régression Logistique, au contraire de Random Forest, semble très bien performer sur des côtes "moyennes", cependant une fois de plus c'est sur les outsiders qu'on obtient les meilleurs résultats.

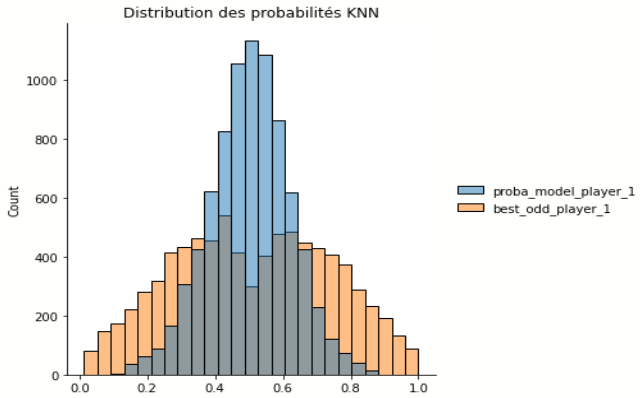
En gardant uniquement les côtes entre 2,05 et 6,93, la Régression Logistique obtient un résultat net de +158,62€ sur 6662 matchs, soit un 2,38% de rentabilité sur 77,5% des matchs du jeu de Test. Modifier le seuil d'EV n'est pas rentable pour ce modèle.

C. Stratégie avec Gradient Boosting Classifier

Nous obtenons un DataFrame contenant beaucoup moins de côtes considérées comme rentables, seulement 874 (10,2% des matchs). De plus, sur cet échantillon, le modèle est perdant (-41.91€), avec une rentabilité de -4,80%. Nous n'allons pas explorer ce modèle plus en détail.

D. Stratégie avec K Plus Proches Voisins

Nous obtenons un DataFrame de 8188 côtes rentables (95,2% des matchs), avec comme pour Random Forest mais de manière encore plus marquée, une tendance à considérer que les matchs sont beaucoup plus équilibrés que ce qu'estiment les bookmakers. Il en résulte que pour quasiment tous les matchs, KNN parie pour l'outsider dès lors que les côtes ne sont pas trop proches.

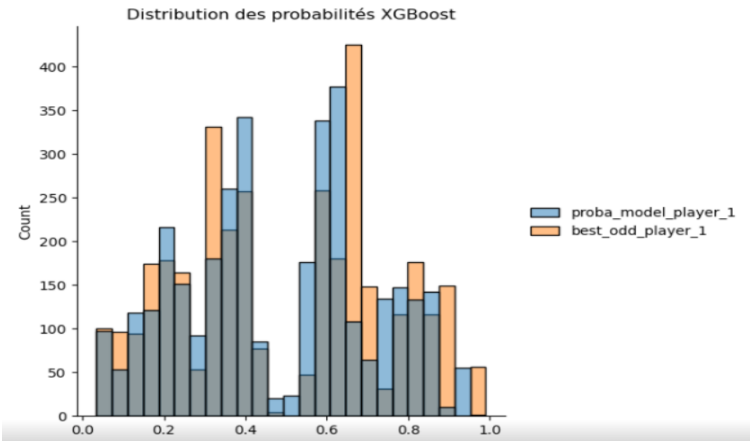


En pariant sur toutes les côtes rentables selon KNN, on obtient un résultat net de 120,94€ (+1,48%). C'est le meilleur résultat observé parmi nos modèles simples.

Là aussi, le résultat optimal vient lorsque l'on limite les côtes à celles entre 2,10 et 6,93. On arrive alors à un résultat net de + 214,63€ sur 6405 matchs (+3,35% sur 74,5% des matchs disponibles)

E. Stratégie avec le XGBoost

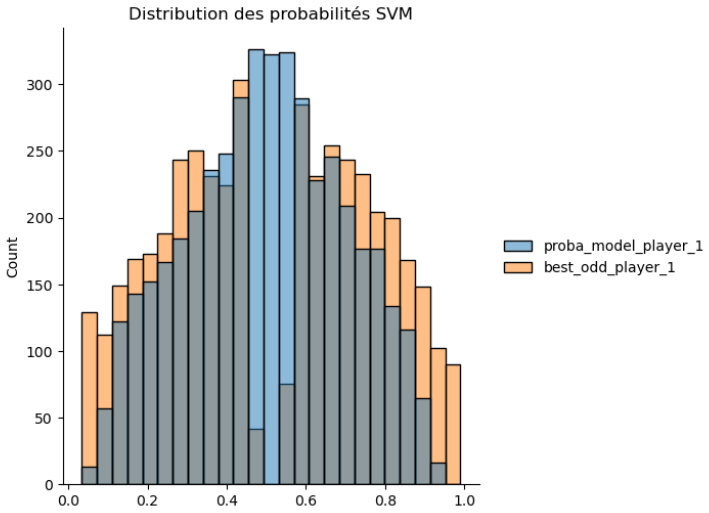
Nous obtenons un DataFrame contenant 3443 côtes considérées comme rentables (40,02% des matchs). Sur ces matchs supposés rentables, on observe que la distribution des probabilités de victoires du modèle est plus dense aux extrémités que la distribution des probabilités de victoires des bookmakers.



En pariant 1 euro sur toutes les cotes que le modèle considère comme rentables sur l'ensemble test, nous gagnons au total 110.67 euros

F. Machine à vecteurs de support

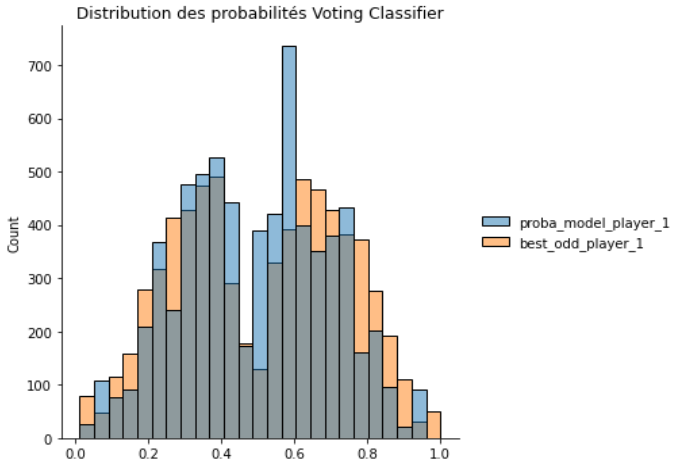
Nous avons un DataFrame de 4446 côtes profitables (51,7% de tous les matchs). On constate sur le graphe ci-dessous que la distribution des probabilités de victoires du modèle est plus dense au centre que la distribution des probabilités de victoires des bookmakers.



En pariant sur toutes les côtes considérées comme rentables par le modèle, nous obtenons un bénéfice net de 132,57€ (+2,98% de rendement)

G. Stratégie avec Voting Classifier (LR + RF + XGBoost)

Nous avons un DataFrame de 6922 côtes profitables (80,5% de tous les matchs), et une distribution des probabilités moins centrée que pour la plupart des autres classifieurs. On remarque aussi que les matchs profitables semblent de nouveau souvent ceux avec des côtes relativement déséquilibrées mais pas trop.



Notre stratégie habituelle nous donne une profitabilité plus importante qu'avec les modèles précédents : +162,92€ (+2,35%).

En limitant nos paris aux côtes comprises entre 2,15 et 31,5, on obtient un profit de +220,17€ (+3,49%) sur 6309 matchs (73,3% des matchs disponibles)

Encore une fois, mettre un seuil minimal sur EV_net n'est pas profitable.

N'ayant pas de possibilité de prouver que la valeur des côtes a une incidence sur la fiabilité du modèle, et pour éviter tout biais potentiel nous sélectionnons simplement le modèle ayant obtenu les meilleurs résultats pour tous les paris possibles avec EV_net > 0.

H. Tableau récapitulatif des résultats

Classifier	Rand. Forest	Reg Logistique	Gr. Boosting	KNN	**Voting**	**XGBoost**	**SVM**
Côtes retenues	7079	7691	874	**8188**	6922	3443	4446
Côtes retenues (%)	82,3%	89,4%	10,2%	**95,2%**	80,5%	40,02%	51,7%
Résultat net (€)	107,02€	83,31€	-41,91€	120,94€	**162,92€**	110,67€	132,57€
ROI (%)	+1,51%	+1,08%	-4,80%	+1,48%	+2,35%	**3,21%**	2,98%

Pour notre stratégie de pari, nous décidons donc d'employer le Voting Classifier avec Régression Logistique, Random Forest et XGBoost, en pariant à chaque fois le même montant sur chaque match dont l'espérance de gain est positive. Il s'agit du modèle nous permettant de gagner le plus d'argent grâce à la combinaison d'un bon ROI et un volume de côtes rentables très satisfaisant.

8. Conclusion

En conclusion et pour récapituler les actions mises en œuvre pour mener à bien ce projet, nous avons, dans un premier temps, explorer la base de données à disposition sur le site kaggle.

Très rapidement, nous nous sommes rendu compte que les informations qui figuraient dans ce jeu de données pouvait être enrichies. Par conséquent, nous avons collecté de nouvelles données en utilisant notamment un code de web scraping disponible sur GitHub. Par la suite, nous avons nettoyé ce jeu de données des informations superflues et mis en commun certaines variables redondantes dans le but de réduire le nombre de variables explicatives tout en gardant le maximum d'information pertinente.

L'étape suivante a été de re-manipuler les entrées de notre tableau de données afin que, pour chaque entrée, nous ayons la moyenne mobile des X dernières variables (correspondants au X derniers matchs) tout en excluant les résultats (variables) du match relatif à l'entrée. Et ce, pour chacun des deux joueurs. Dorénavant, une entrée ne correspond plus aux résultats (statistiques des joueurs pendant le match essentiellement) d'un match mais à la moyenne des résultats de chaque joueur avant match sur X matchs. Il est à noter que cette variable X est un hyper-paramètre que nous avons finalement fixé à 50 (taille de fenêtre qui maximise l'accuracy).

Enfin, nous avons également créé notre variable cible. Pour ce faire, nous avons procédé comme suit : Nous avons tiré au sort les noms des joueurs 1 et 2 pour chaque match (auparavant, nous avions le gagnant et le perdant), puis, nous avons créé notre variable cible, binaire, qui répond à la question 'le joueur 1 a-t-il gagné le match ?'.

Une fois la partie de pré-processing terminée, nous avons lancé diverses modélisations à la recherche du modèle de classification qui nous donnera la meilleure estimation, suivant une méthode en entonnoir, c'est-à-dire, en commençant par des modélisations naïves puis en affinant les hyper-paramètres puis en confrontant chacun des modèles entre-eux. Ainsi, le modèle qui nous a donné le meilleur résultat est une régression logistique, avec une accuracy à 67,55% (LogisticRegression(C=0.005, max_iter=2000, penalty='l1', random_state=42, solver='liblinear').

Néanmoins, les décisions collégiales étant souvent les meilleures (intelligence collective oblige), nous avons fait appel à un Voting Classifier, en prenant le soin de changer notre groupe d'experts (nos modèles) toujours dans le but d'aller chercher de meilleures estimations. Nous avons ainsi porté la meilleure accuracy à 67,58% (LR + GBC, mode 'hard').

Nous avons donc réussi à trouver un modèle qui nous donne une meilleure accuracy que celles des bookmakers B365 (66,35%) et Pinnacle (67,47%) !

Nous avons également implémenté un modèle deep learning qui a fourni de moins bon résultat (Manque d'observations ?).

Enfin, nous avons mis en place une stratégie de pari simple mais efficace qui consiste à ne parier que si notre modèle considère ce pari rentable. Ainsi, nous minimisons le risque de

perdre notre mise. Cette stratégie a été testé non par sur le meilleur modèle mais sur l'ensemble des modèles. Nous ne cherchons plus à avoir la meilleure estimation mais les plus grands gains, ainsi la mise en place de cette stratégie redistribue les cartes entre nos modèles.

Nous avons réussi à atteindre un retour sur investissement de 3,21% avec le modèle XGBoost. Néanmoins, nous préférons le Voting Classifier, avec un retour sur investissement de 2,35% mais un volume de pari plus élevé (6922 contre 3433 pour XGboost), pour la simple et bonne raison que cela permet de limiter le hasard et profiter des systèmes de fidélité des bookmakers (récompenses sous forme de paris gratuits, cashback…etc.). Ce qui n'est pas négligeable.

Nous pouvons donc affirmer que nous avons bel et bien battu les bookmakers.

De nombreux axes d'améliorations sont possibles et réalisables dans le cadre de travaux futurs :

- Fine-tuning des modèles sur différentes périodes de test
- Actualisation du dataset avec l'intégration des matchs les plus récents (scraping des cotes établies par d'autres bookmakers, scraping et enrichissement du dataset avec encore plus de matchs et plus de statistiques), et re-manipulation (effectuer une sélection plus avancée des variables, effectuer une transformation plus fine des variables)
- Prendre en compte le classement moyen ATP des adversaires avec le calcul des moyennes mobiles (Gagner 15 balles de break face au numéro 1 mondiale n'est pas la même chose que face à un joueur classé 250) et le classement Elo des joueurs.
- Implémenter un modèle de deep learning, notamment un réseau de neurones récurrents qui prendrait en entrée l'historique de chacun des matchs des joueurs et donnerait le vainqueur (Problématique : les nouveaux joueurs ayant peu de match à leur actif)
- Approfondir notre stratégie de paris en modulant la taille des mises et en réinvestissant les bénéfices.

9. Références

- ATP matches dataset | Kaggle
- https://towardsdatascience.com/making-big-bucks-with-a-data-driven-sports-betting-strategy-6c21a6869171
- (PDF) Exploiting sports-betting market using machine learning (researchgate.net)
- https://fr.wikipedia.org/wiki/Pari_sportif
- Tennis results and betting odds data for tennis betting system development
- http://tennis-data.co.uk/alldata.php
- https://github.com/serve-and-volley/atp-world-tour-tennis-data
- https://www.researchgate.net/publication/331218530_Exploiting_sports_betting_market_using_machine_learning
- https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4