Est-il possible de prédire les résultats de la première ligue anglaise?

Melchior Prugniaud

Master Specialisé Data Science - 2019-2020
ENSAE PARIS TECH
Cours : Introduction au machine learning

2 février 2020

Abstract

A l'aide des résultats de la première ligue anglaise depuis la saison 2009/2010 nous cherchons à prédire les probabilités de l'issu d'un match à l'aide de variables facilement obtenable sur internet. Nous chercherons à observer si des gains sont possibles sur la dernière saison en cours.

1 Analyse descriptives des données

1.1 Les données

Les données proviennent de plusieurs sources différentes. Dans un premier temps, l'ensemble des résultats des matchs de football proviennent du site football-data. Nous avons par la suite décidé de nous concentrer sur la première ligue anglaise (EPL) uniquement, car après quelques recherches sur l'ensemble des ligues européennes,il apparaît que chacune a des caractéristiques propres (par exemple, un nombre de buts beaucoup plus fort en Espagne qu'en France). Et nous avons choisis de prendre uniquement l'EPL étant donnée que les ligues inférieures donnent des résultats assez hasardeux avec une répartition des probabilités proche d'un tiers pour chaque issue. De ce jeu de données, nous ne gardons que les scores et les côtes associées (pour l'évaluation de nos résultats) et ne prenons pas en compte le nombre de fautes commises, de tirs...
Un autre jeu de donnée concernant les notes des équipes de football tiré du jeu FIFA a été scrappé à l'aide de scrapy sur le site Fifa Index. De ce dataset, nous ne tirons que trois variables qui sont le potentiel défensif, offensif et du milieu de terrain de chaque équipe à une date donnée.
Le dernier jeu de donnée que nous avons choisi de prendre est issu du site Transfermarkt. Ce site allemand évalue la valeur des joueurs et donne des informations concernant la taille de ceux-ci ainsi que leur âge.
Enfin, nous aurions voulu ajouter les compositions d'équipes pour pouvoir observer l'influence de la présence d'un joueur dans la performance de l'équipe en mettant en relation les onze de départs avec les notes individuelles de chaque joueur tiré de FIFA. Malheureusement, l'historique des compositions de départs n'est pas une donnée facilement récupérable sur internet. Le site Football lineups permet d'avoir ses informations mais malgré des tentatives de rotations d'IP, changement d'user agent... il n'a pas été possible de récupérer ses informations.[1]

1.2 Analyse des différentes variables

Nous possédons donc un dataset contenant 3659 matchs d'EPL couvrant la période de 2010 à 2019. Nous pouvons déjà remarquer qu'il semble y avoir un net avantage de jouer à domicile avec 46% de victoire à domicile contre uniquement 29% à l'extérieur. Cet effet est assez connu dans l'analyse footballistique et s'appelle le 'home advantage'. Nous avons remarque que cet effet est constant dans le temps sur l'ensemble des saisons. Etant donné qu'une victoire a lieu quand l'équipe A met le plus de but contre l'équipe B, nous avons observé que le nombre de buts à l'extérieur est inférieur au nombre de buts marqué

[1] Cf les tentatives dans les crawlers du projet

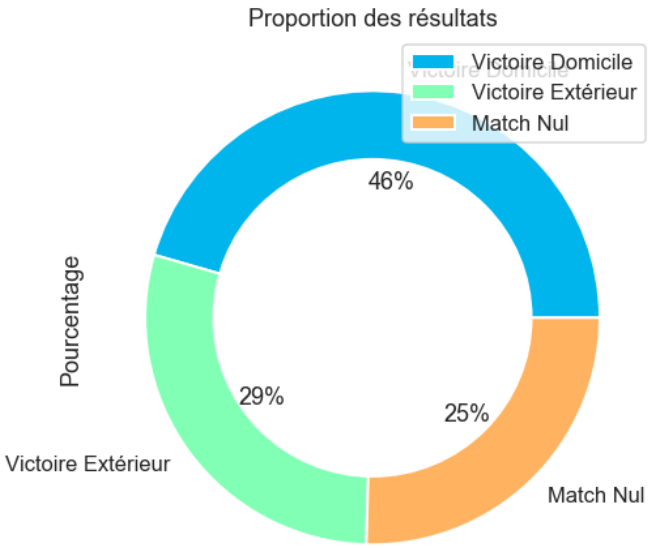

Figure 1: Répartition des résultats

à domicile. En effet, il semble exister un écart de 0.5 but entre les deux. Ces résultats ne varient que très peu au cours de la période étudiée.

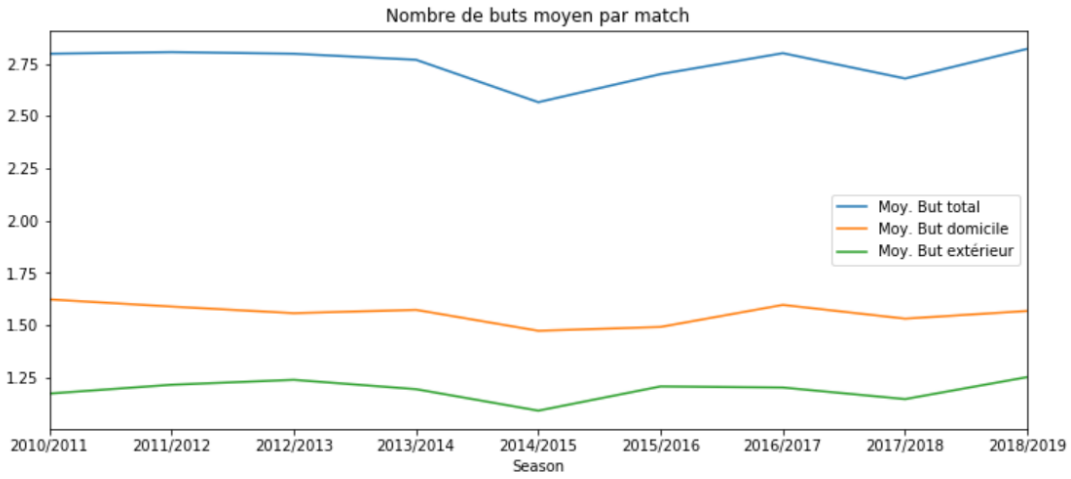

Figure 2: Nombre moyen de buts par lieu et par saison

A l'aide de la figure 3, il est clair que la fréquence du nombre de buts à domicile et de la fréquence des buts extérieurs n'est pas la même, exceptée pour les matchs sans aucuns buts et les matchs avec des grands scores.

L'ensemble de nos analyses sur cette différence de buts tend à nous indiquer que des variables doivent capturer l'avantage de jouer à domicile, mais aussi qu'il est possible de créer des variables qui en fonction du score vont calculer un indicateur donnant un avantage pour l'équipe recevant ou visiteuse.

Dans cette partie, nous allons nous concentrer désormais sur les données issues du site transfermarkt. La principale information de ce dataset concerne la valeur des joueurs, leur âge, leur nationalité et leur taille. Manchester City ayant gagné quatre titres sur la période, il est logique de voir que cette équipe est celle qui a le plus de valeur sur le marché en 2019. Elle est suivie par Liverpool qui est une équipe qui vient de gagner la ligue des champions en 2019. De plus, il semble y avoir une grande différence entre la valeur des équipes du 'big 6' (Manchester City, Manchester United, Liverpool, Chealsea, Arsenal et les Spurs) et les équipes restantes. La valeur moyenne d'une équipe peut donc avoir une influence sur le résultat d'un match. A noter que depuis 2015, la valeur totale des équipes d'EPL a presque doublé atteignant plus de neuf milliards de dollars.

Concernant les nationalités des joueurs, une grande majorité d'entre eux est issus de l'Angleterre

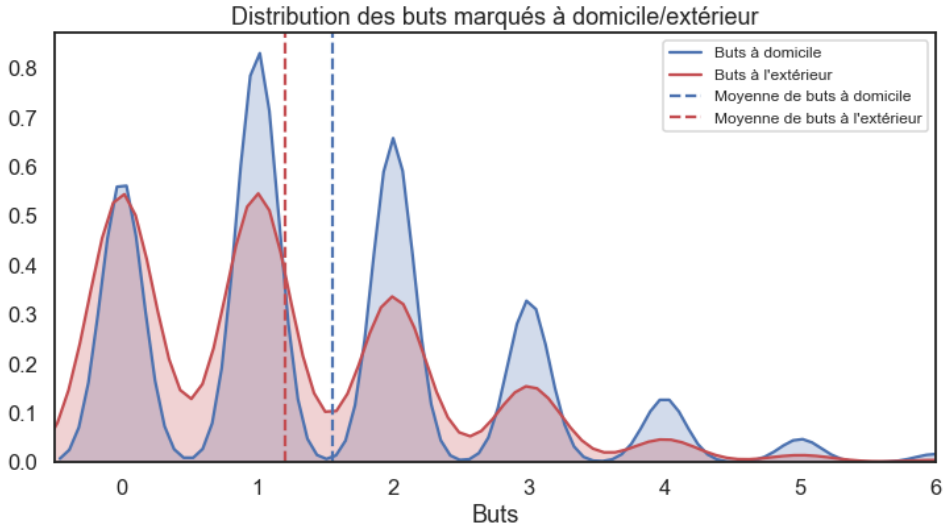

Figure 3: Distribution du nombre de buts

même si en y regardant de plus près, il est possible d'observer que les équipes du bas de classement ou venant de monter possède plus de joueur anglais tandis que dans le haut du classement, il semble y avoir beaucoup plus de joueur étranger. Par exemple, en 2019, dans l'équipe de Liverpool (l'actuel leader) il y a huit anglais et seize étrangers tandis que dans l'équipe de Bournemouth, actuel 17 ème, il y a dix sept joueurs anglais.

Au regard de l'âge moyen de chaque équipe et de la taille moyenne, il ne semble pas que ces variables changent beaucoup le résultat d'un match, car il n'y a pas une très grande dispersion entre les moyennes de chaque équipe. Et en comparant les valeurs moyennes de chaque équipe qui ont été championne d'Angleterre, il ne semble pas exister de tendance déterminante.

2 Traitement des variables

Dans cette partie, nous allons présenter le traitement apporté à nos différentes variables. Le premier traitement concerne la création de variable issue d'une modélisation statistique. En effet, Maher en 1983 avait tenté de modéliser le nombre de buts qu'une équipe pouvait marquer à l'aide d'un GLM avec une loi de poisson. A l'aide de l'analyse de Maher [2], cinq variables vont être crée qui contiendront le nombre de buts espéré de chaque équipe ainsi que les probabilités de trois différents résultats issus des distributions prédictives calculés à l'aide des coefficients. Dans un premier temps, il est possible de voir par exemple, pour la saison 2018/2019, qu'une loi de poisson suit bien le nombre de buts marqué à l'extérieur et à domicile comme le démontre la figure 5. Par contre, le modèle de poisson semble sous-estimer les matchs sans buts que ce soit pour les buts extérieur ou domicile. Et le modèle surestime légèrement les buts à domicile entre 1 et 3.
Ainsi, à l'aide d'une fonction personnalisée, nous allons créer nos cinq variables. Sachant que la méthode de Maher fonctionne mieux à partir de la moitié des matchs joués, nous allons appliquer des conditions pour la valeur des variables :

- HG (Home Goal) : But à domicile
- AG (Away Goal) : But à l'extérieur
- ProbaHW (Probability Home Win) : Probabilté de victoire de l'équipe domicile
- ProbaAW (Probability Away Win) : Probabilté de victoire de l'équipe extérieur
- ProbaD (Probability Draw) : Probabilté de match nul

[2]Maher Modelling association football scores

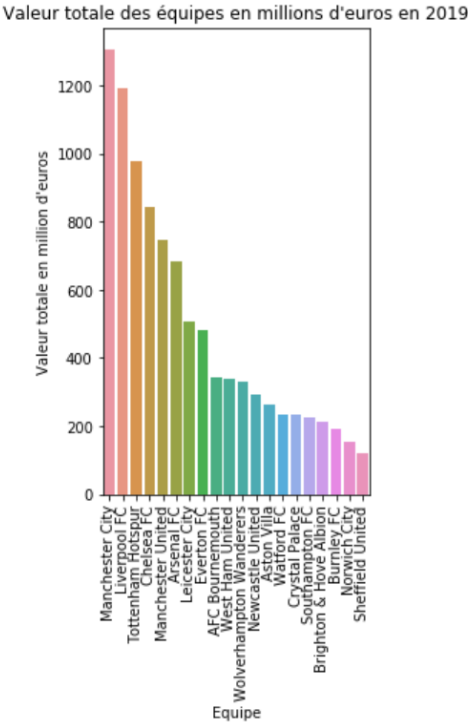

Figure 4: Valeur des équipes en millions d'euros

Algorithm 1: Fonction MaherxGt

Result: [HG,AG,ProbaHW,ProbaD,ProbaAW]
if *Nombre de matchs > 19* **then**
  Méthode de Maher pour exprimer les variables
else
  if *(Nombre de buts = 0) or (Nombre de matchs <4)* **then**
    Utilisation de la moyenne de buts de l'an dernier pour exprimer une loi de poisson
  else
    Utilisation de la moyenne de buts des matchs de cette saison pour exprimer une loi de poisson
  end
end

Un ensemble d'autres variables est crée à partir du score et de la date. En effet, les variables suivantes sont créées :

- WRH/WRA-DRH/DRA-LRH/LRA: Taux de victoire - match nul - défaite de l'équipe recevante/visiteuse
- PMHT/PMAT : Nombre de match joué de l'équipe domicile/extérieur
- RTHT/RTAT : Temps de repos entre deux matchs
- EFH/EFA : Etat de forme de l'équipe à domicile/extérieur sur les cinq derniers matchs
- HFA : Avantage de jouer à domicile, calculé avec la différence de buts mis à domicile et les buts pris à domicile divisé par le nombre de match joué à domicile.
- STHT/STAT : Classement des différentes équipes
- WRAH/WRAA : Taux de victoire entre les deux équipes
- USHT/USAT : Invincibilité en championnat pour les deux équipes
- Ainsi qu'un ensemble de variable calculant le nombre de buts sous différents aspects.

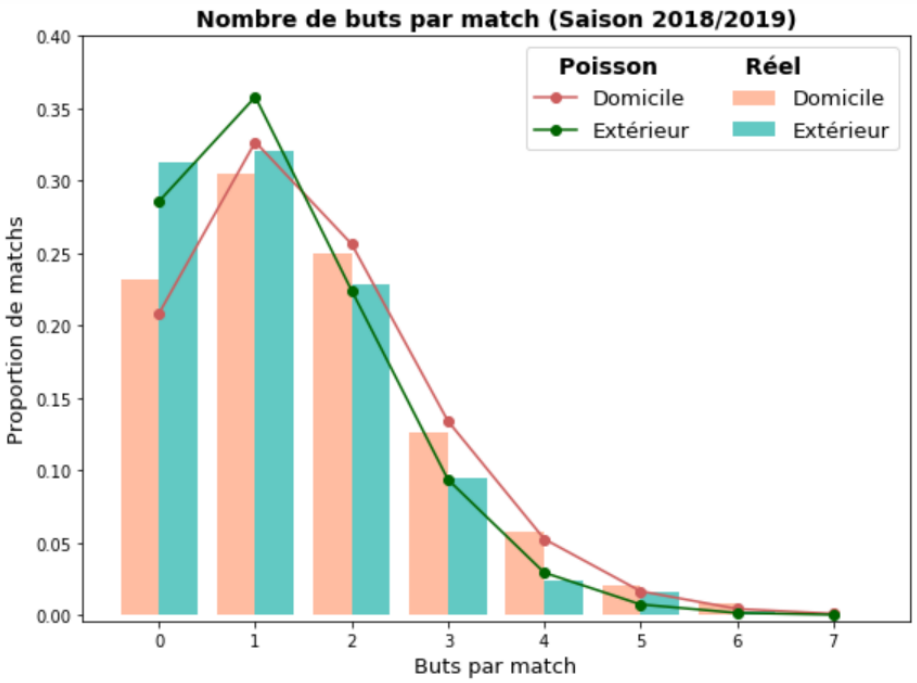

Figure 5: Nombre de buts par match

Six dernières variables sont créées à partir d'un système d'ELO ranking amélioré crée par Constantinou[3] qui s'appelle le pi rating. Ce système possède des meilleurs résultats que le système d'ELO classique et nous l'avons donc incorporé en tant que variable donnant la force de chaque équipe à domicile, à l'extérieur et en général. Par la suite, nous verrons si toutes ces variables sont bien utilisable.

Enfin, à partir de nos deux autres dataset, nous récupérons les notes collectives de chaque équipe à la date la plus proche du match ainsi que les tailles et âge moyen puis le nombre maximal (top 1 et 2) de joueur de même nationalité dans un club. Il est ainsi possible d'obtenir les particularités de chaque équipe. Dans le graphique ci-dessous, nous avons normalisé les données pour observer les différences notables entre l'actuel premier d'EPL et le dernier.

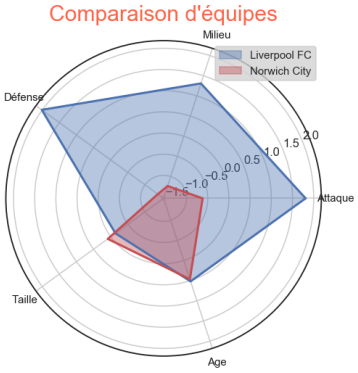

Figure 6: Comparaison entre Liverpool et Norwich City

3 Modélisations

Pour appliquer les différents algorithmes nous avons divisé notre jeu de données en deux. Les données de 2010 à 2019 permettent d'entraîner et de tester l'algorithme, et les données de la saison en cours vont

[3]ANTHONY C. CONSTANTINOU, NORMAN E. FENTON : Determining the level of ability of football teams by dynamic ratings based on the relative discrepancies in scores between adversaries

nous permettre de vérifier si nous réalisons des gains à l'aide des résultats.

Dans un premier temps, différents modèle sans optimisation seront réalisé puis une analyse de nos variables et de l'influence qu'elles peuvent avoir sur le résultat d'un match sera réalisé. Nous allons donc réduire le nombre de variables en fonction de l'importance qu'elles apportent aux résultats, mais aussi si elles sont corrélées entre elles pour éviter d'apporter la même information sous différentes formes.

3.1 Premières modélisations

Une première approche serait de lancer un ensemble d'algorithme possible en ne réalisant aucune sélection ou analyse supplémentaire sur nos variables. C'est ce que nous allons faire dans cette partie, qui nous servira de base de comparaison pour les futurs ajouts ou suppressions.

Mais tout d'abord, nous allons déterminer une mesure permettant d'observer la qualité de prédiction de nos variables. Pour se faire, nous allons utiliser la mesure de De Finetti basé sur la distance du même nom.

$$DeFinettiDistance = ||Pred - true||^2 \tag{1}$$

Avec :

$$Pred = (\mathbb{P}_{win}, \mathbb{P}_{draw}, \mathbb{P}_{loose})$$

Et :

$$true = (win, draw, loose)$$

Par exemple, pour un match avec une prédiction telle que pred = (0.5,0.25,0.25) et un résultat conduisant à une victoire, la distance de De Finetti est :

$$Df = (0.5 - 1)^2 + (0.25 - 0)^2 + (0.25 - 0)^2 = 0.375$$

Par la suite pour calculer la mesure de DeFenetti il suffit de sommer l'ensemble des distances et de diviser cette somme par le nombre de prédictions réalisées. Une autre mesure de performance que nous pouvons utiliser est la perte logistique qui prend des valeurs entre 0 et l'infini. Plus la valeur est proche de zéro plus notre classifier réalise des prédictions fiables.

Comme dit auparavant, nous allons lancer différents algorithmes de classification pour observer les résultats obtenus. Mais nous n'allons pas tous les optimiser, car certains sont plus efficaces que les autres dès le départ et ils nous seviront de base de comparaison. Et d'après plusieurs études qui ont déjà été réalisées, il semblerait que certains sont plus efficaces que d'autres notamment les méthodes de réseaux de neurones (que nous ne couvriront pas ici), les SVM et l'XGBOOST.

Les premiers résultats que nous obtenons sans avoir faits d'optimisations sont les suivants :

	Mesure de DeFinetti	LogLoss	Accuracy
Régression Logistique	0.556	0.942	0.555
Random Forest	0.596	1.872	0.529
XGBOOST	0.516	0.875	0.59
KNN	0.666	4.599	0.463
Naives Bayes	0.755	2.739	0.546
Arbre de décision	1.087	18.127	0.451
Svm	0.635	1.055	0.475

Figure 7: Résultats bruts sans optimisation

Comme dit auparavant, nous remarquons que les résultats observés par d'autres analyses sont en partie confirmés. Le modèle le plus efficace est le modèle XGBOOST avec une mesure de DeFinetti proche de 0.5 et une précision d'environ 60%. Néanmoins la régression logistique donne des résultats plutôt satisfaisant aussi. Ainsi, pour le moment, nous retenons les SVM, la régression logistique, les forêts aléatoires et l'XGBOOST.

Nous allons désormais 'tuner' ces algorithmes à l'aide de la GridSearchCV de sklearn pour voir si nous pouvons améliorer nos premiers résultats car pour le moment, notamment pour les SVM la matrice de confusion permet de voir que celui ci-prédit à tort presque uniquement que des victoires à domicile. A l'aide des différentes documentations de scikit-learn, nous pouvons choisir des critères que nous tentons d'optimiser. Par exemple, pour la régression logistique, différentes pénalisations pour les variables sont sélectionnées en fonction des différents algorithmes utilisés pour résoudre le problème d'optimisation de la descente de gradient. L'optimisation via la GridSearchCV nous donne comme solver 'liblinear' avec la

pénalité 'l1' qui équivaut à la magnitude de tous nos coefficients. L'optimisation que nous avons réalisée permet de réduire la distance de DeFinetti de presque 0.3, mais fait aussi diminuer de 0.1 la qualité de nos prédictions. Pour les autres algorithmes que nous avons optimisés, le résultat est visible dans le tableau ci-dessous. A noter que pour l'XGBOOST, nos tentatives d'optimisation ne semblent pas bien fonctionner car nous perdons des points dans nos différentes métriques.

3.2 Analyse et sélection des variables

Ayant pour le moment 61 variables explicatives, il nous faudra dimuner ce nombre afin de réduire l'overfitting et d'augmenter le résultat de nos prédictions. De plus, même si ce n'est pas notre cas car notre jeu de données est assez petit, il est important de réduire le nombre de variables pour diminuer le temps de traitement lorsque l'on est en situation de production pour une entreprise. Le coefficient de corrélation de Pearson est utilisé pour produire une matrice de corrélation permettant d'observer les variables ayant un lien trop fort entre elles. Pour observer les corrélations fortes entre nos variables, nous nous aidons d'un visuel graphique.

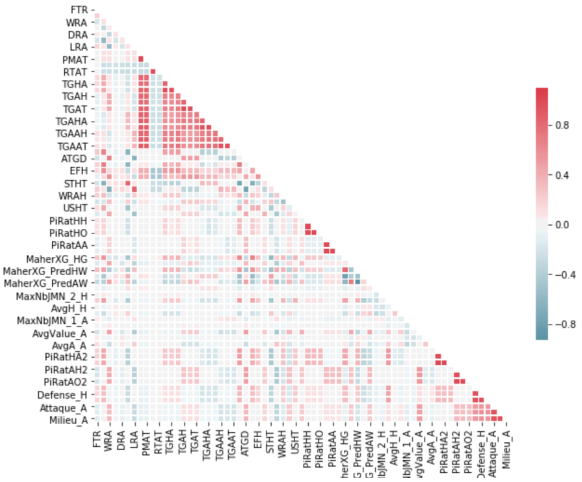

Figure 8: Matrice de corrélation

Dès lors, d'importantes corrélations apparaissent entre nos différentes variables. De ce fait, il semblerait que certains apportent la même information ce qui peut fausser nos prédictions. Nous décidons à partir de cette première analyse de supprimer les variables ayant une corrélation avec d'autres proches de plus ou moins un. En plus de regarder la corrélation entre nos variables, nous allons observer l'importance de celles-ci à notre modélisation à l'aide du module SelectFromModel de scikit-learn. Nous appliquons cette méthode à différents algorithmes : la régression logistique, Random Forest, XGBOOST et lightGBM et ajoutons la méthode RFE (Recursive Feature Elimination) de scikit-learn qui permet de construire des modèles grâce à différentes combinaisons de variable. Au final, nous ne gardons que les variables présente au moins une fois dans les différentes méthodes de sélection de variables, ce qui nous donne un modèle réduit de 16 variables.

Feature	RFE	RegLog	XGB	Random Forest	LightGBM	Total
PiRatHO	True	True	True	True	True	5
PiRatAO	True	True	True	True	True	5
AvgValue_H	True	True	True	True	True	5
MaherXG_HG	True	True	False	True	True	4
AvgValue_A	True	True	True	True	False	4
WRH	True	True	True	False	False	3
MaherXG_AG	True	True	False	False	True	3
WRAH	True	True	False	False	False	2
WRAA	True	True	False	False	False	2
WRA	True	True	False	False	False	2
AvgA_H	False	True	False	False	True	2
TGAAT	False	False	False	False	True	1
STAT	False	False	True	False	False	1
HTGD	False	False	True	False	False	1
EFA	False	False	True	False	False	1
AvgA_A	False	True	False	False	False	1

Figure 9: Variables sélectionnées

3.3 Seconde modélisations

Nous recommençons les étapes précédentes avec cette fois-ci les variables restantes pour obtenir dans un premier temps des résultats sans optimisation et ensuite des résultats avec une GridSearchCV. La première chose que nous remarquons est que globalement la distance de DeFinetti diminue pour presque tous nos prédicteurs. Il semblerait donc que la sélection de variables permet d'obtenir de meilleurs résultats. Au final, après les différentes modélisations, nous pouvons réaliser les différentes matrices de confusions pour pouvoir observer la précision en fonction des différents labels. En effet, en regardant la matrice de confusion ci-dessous, nous pouvons observer que le classifieur XGBOOST aura tendance à sur-évaluer les victoires à domicile (ce qui est aussi logique au vu des statistiques descriptives étant donné qu'il y a presque 50% de victoires pour les équipes recevantes). De plus, le football reste un sport pratiqué par des humains et il est normal d'y retrouver des aberrations d'un point de vue statistique.

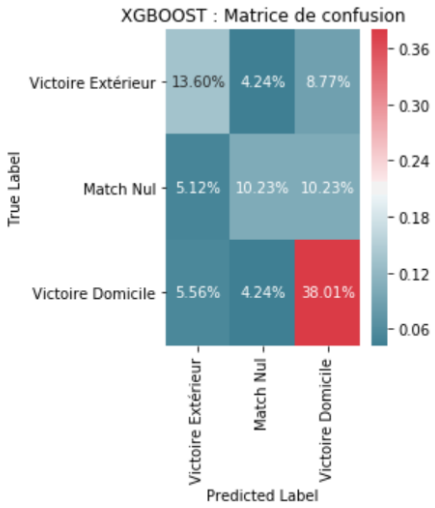

Figure 10: Matrice de confusion modèlé réduit

Au final, le modèle que nous choisissons est le XGBOOST réduit qui nous servait de base de comparatif. Il permet d'obtenir une mesure de DeFinetti égale à 0.519 avec un taux de précision de 61.8% sur le test set.

4 Résultats

Le modèle que nous sélectionnons donc est l'XGBOOST avec les paramètres choisi par l'algorithme et non une GridSearchCV. Pour mesurer la bonne qualité de nos prédictions nous avons la mesure de DeFinetti qui va observer l'écart entre une prédiction et le résultat. Mais nous allons aussi observer le montant d'argent que nous pouvons gagner (ou perdre) en pariant sur la plus haute probabilité calculé par notre modèle choisi.

Un premier modèle très simple consisterait à placer deux euros sur chaque match en choisissant la probabilité la plus élevée. Un second, un peu plus complexe, consisterait à ne parier que sur les matchs où la probabilité calculée est plus élevée que celle du bookmaker. Enfin, le troisième modèle utilise le critère de Kelly et un dépôt d'argent (bankroll) initial pour calculer des gains. Le critère de Kelly permet de calculer le montant de la mise à faire sur chaque match. Il se calcule de la manière suivante :

$$K_f = \frac{(bp - q)}{b}$$

Avec b la cote du bookmaker à laquelle on soustrait 1, p la probabilité associée à l'issu du match que l'on a calculé et q égale à 1 moins p. Grâce au critère de Kelly, on obtient une fraction de la bankroll à miser sur le match, sachant que si ce critère est inférieur à 0 alors il ne faut pas placer de paris car le risque pris n'en vaut pas la peine.

Les graphiques ci-dessous nous donnent les résultats de l'ensemble des trois stratégies de management de pari. Le nombre de matches total dans le jeu de données depuis le début de la saison 2019/2020 est de 239.

C'est à partir d'ici qu'il semblerait y avoir un problème avec les résultats. Autant, avoir un taux de bonnes prédictions entre 50 et 60% ne semblent pas être si extraordinaire, il semblerait toutefois que les gains possibles avec les différentes modélisations soient idylliques et beaucoup trop rentable. Malheureusement, n'ayant sans doute pas assez de recul, je n'arrive pas à détecter où est le problème dans mes évaluations de gains. En effet, rien que pour la première stratégie, qui semble très naïve, il semblerait que l'on soit gagnant de 95 euros en plaçant simplement 2 euros sur chaque match avec une proportion de plus de 55% de bonnes prédictions.

Par contre, en ne misant que sur les matchs où les cotes calculées sont supérieures à celles des bookmakers, nous ne parions que sur 82 matches avec un taux de réussite approchant les 60%. Malgré cela, le gain moyen pour ces matchs là est de 44 cents par pari tandis que les pertes moyennes sont de 97 cents pour une bankroll de 100 euros en ne misant qu'un pourcent de la bankroll à chaque fois. Au final, 10.43 euros sont perdus. En appliquant cette stratégie et une mise de 2 euros sur chacun des matchs, la perte s'élève cette fois-ci à 21.56 euros.

Enfin, après avoir tester le modèle sur la période mi février à mars 2020 (début du confinement et arrêt des matchs), on observe un ROC d'environ 82% qui est tout de même à nuancer car il y a une grande période de perte sur une suite de matchs mal pronostiqué.

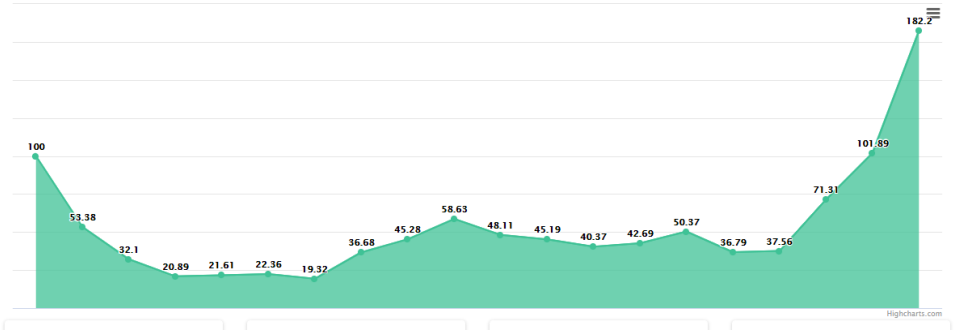

Figure 11: Résultats avec la méthode de Kelly pour la période Février-Mars 2020

5 Conclusion

Avant de commencer toute modélisation, il était évident qu'il semblait difficile, avec aussi peu de données et des méthodes de machine learning assez basiques, d'obtenir un taux de bonnes prédictions de plus de 66%. Le but était de faire environ 60% de bonnes prédictions, car il s'agit de ce que l'on trouve comme résultats de prédictions sur les différents projets du même genre.

Néanmoins, l'intérêt était d'ajouter un outil d'aide à la prise de décision pour placer un pari sur un match qui avec des données facilement récupérables pourrait augmenter les gains.

Il est évident que des axes d'améliorations sont possible. En effet, l'un des premiers serait de récupérer les compositions d'équipes pour observer la présence ou non d'un joueur. A cela s'ajoute la possibilité par la suite de mettre plus de variables propres à chaque joueurs permettant peut être de capter plus d'informations et d'augmenter les performances de notre algorithme.

Un autre axe serait de rendre l'ensemble des manipulations plus facile d'utilisation en parsant mieux les données, en optimisation les différents temps de calculs et en présentant tout ceci dans une belle interface même si cela reste une interface console.

Ensuite, les variables construites avec le pi rating et notamment celle avec la modélisation de Maher peuvent être amélioré en utilisant des techniques plus complexe d'ELO rating et de modélisation statistique. En effet, des auteurs comme Dixon et Coles ou bien Dimitris Karlis et Ioannis Ntzoufras proposent d'autres solutions comme des lois de Skellam pour modéliser une différence de buts et donc l'issu d'un match.

De plus, il serait important de pouvoir mieux capter l'effet du temps sur le jeu de données, car certaines équipes performent mieux une année (Leicester par exemple en 2015) que d'autres. Un autre

axe d'amélioration concerne le placement des paris et la gestion des risques sur celui-ci. Nos méthodes pour évaluer les gains sont restés assez basiques, mais en dérivant la théorie du portefeuille de Markovitz, il doit être possible de manager au mieux son risque pour le réduire, mais aussi augmenter les gains.

Enfin, il semblerait que les réseaux de neurones soient désormais une modélisation assez utilisée dans ce genre de modélisation. C'est une piste qu'il faudrait étudier pour voir s'il est possible d'augmenter les performances en entrainant un réseau.

Toutes ces pistes peuvent très probablement augmenter le pourcentage de bonne prédiction et diminuer la distance de DeFinetti et sont donc à développer au cours des prochains mois.