

**Projet tutoré : Data Science**

**“ Analyse des tirs de joueurs NBA ”**

**Réalisé par :**

* Najat BOUTABRATINE
* Sarah BOUKRIS
* Philippe WEISSBECK

**Encadré par :**

* Kalomé [BOTOWAMUNGU](mailto:kalome.b@datascientest.com)

**Institut :**

**DATASCIENTEST**

Formation continue – Promotion FEV24



**Table des matières**

1. Présentation du sujet 1

1.2. NBA 1

1.3. Sujet choisi 2

1.4. Contraintes 2

1.5. Méthodologie 4

2. Travail réalisé 6

2.1. Environnement de travail 6

2.2. Les données 6

2.2.2. La structure 6

2.2.3 Analyse 21

2.2.4 Les erreurs 26

2.3. Pre-processing & feature engineering 26

2.3.1. Problèmes rencontrés 27

2.3.2. Nettoyage des données 27

2.3.3. Choix des joueurs 27

2.3.4. Prétraitement et constitution des Données 34

3. Modélisation 35

3.1. Adaptation Daframe 35

3.2. Rapport de modélisation 35

3.2.1. Modèle LightGBM 37

3.2.2. XGBOOST 42

3.2.3. Réseaux de neurones 46

4. Perspectives d’Amélioration 51

5. Problèmes rencontrés 52

6. Bibliographie 55

1. Présentation du sujet

# **1.1. Data Science**

Ce projet supervisé s'inscrit dans le domaine vaste et en constante évolution de la science des données. Cette discipline contemporaine utilise une combinaison d'outils mathématiques, statistiques et informatiques pour traiter et analyser de manière efficace la quantité massive d'informations générées dans notre société moderne. Plus précisément, nous avons exploré des problématiques liées à l'analyse de données, à l'apprentissage automatique et à la prédiction. Pour cela, nous avons employé une variété d'outils technologiques avancés.

Ce rapport détaillera les défis spécifiques que nous avons rencontrés et les méthodes que nous avons utilisées pour les surmonter. Nous décrirons également les étapes de notre processus, depuis la collecte et la préparation des données jusqu'à l'implémentation et l'évaluation des modèles prédictifs. L'objectif est de fournir une compréhension claire de notre approche méthodologique et des résultats obtenus, tout en mettant en évidence l'importance de la science des données dans le traitement des informations complexes.

1.2. NBA

La NBA (National Basketball Association) est la principale ligue de basketball professionnel aux États-Unis et la plus prestigieuse au monde. Fondée en 1946, la NBA a évolué pour devenir une organisation internationale avec aujourd’hui des joueurs internationaux et plus de 20 millions de fans venus de tous les continents.

La ligue est composée de 30 équipes réparties en deux conférences, l'Est et l'Ouest, chacune subdivisée en trois divisions. Au fil des décennies, la NBA a été le théâtre de performances incroyables et de joueurs légendaires comme Michael Jordan, Magic Johnson, Larry Bird, Kobe Bryant et plus récemment LeBron James et Stephen Curry. Ces athlètes ont non seulement marqué l'histoire du sport par leurs compétences extraordinaires, mais ont également eu un impact culturel significatif.

La NBA est également connue pour son engagement envers des causes sociales et communautaires, utilisant sa plateforme pour promouvoir l'égalité, l'inclusion et diverses initiatives philanthropiques.

Chaque année, la saison régulière est suivie des playoffs, culminant avec les finales NBA, un événement très attendu où les meilleures équipes s'affrontent pour le titre de champion. Grâce à sa popularité croissante et à sa portée mondiale, la NBA continue d'inspirer et de captiver les amateurs de basketball partout dans le monde. En 2022 la NBA a généré plus de 10 milliards de revenus, faisant du basket et de cette ligue un sport et des évènements qu’on ne « lâche pas »

1.3. Sujet choisi

Les sports américains sont très friands de statistiques, et la NBA (National Basketball Association) ne fait pas exception à la règle.

Le développement constant des nouvelles technologies et des outils numériques, permet désormais de suivre en temps réel les déplacements de tous les joueurs sur un terrain de basket. Les données recueillies sont ainsi très nombreuses et riches.

Le but de ce projet est de :

Comparer les tirs (fréquence et efficacité au tir par situation de jeu et par localisation sur le terrain) de 20 des meilleurs joueurs de NBA du 21ème siècle (selon ESPN).

Pour chacun de ces 20 joueurs encore actifs aujourd’hui (de LeBron James à Giannis Antetokounmpo), estimer à l’aide d’un modèle la probabilité qu’à leur tir de rentrer dans le panier, en fonction de différentes métriques.

La difficulté de ce projet réside dans l’élaboration d’un modèle fiable, ainsi que dans le fait que de nombreux paramètres, parfois difficilement quantifiables ou difficiles à trouver, peuvent avoir une influence importante sur l’issue d’un tir : la pression défensive exercée sur le tireur, la force du contact avec un défenseur, l’orientation des appuis du tireur face au panier, la bonne tenue du ballon en main, la qualité de la dernière passe avant le tir, l’état physique du joueur à ce moment, etc.

1.4. Contraintes

Le projet de comparer les tirs de 20 des meilleurs joueurs de la NBA du 21ème siècle et d'estimer la probabilité de réussite de leurs tirs présentent plusieurs contraintes et défis à prendre en compte :

* **Contraintes de Données**

1. **Qualité et Complétude des Données** :
   * **Exactitude des Données** : Les données collectées doivent être précises et complètes. Les erreurs ou les données manquantes peuvent affecter la fiabilité du modèle.
   * **Granularité des Données** : Les données doivent être suffisamment détaillées pour capturer toutes les métriques pertinentes (position exacte sur le terrain, défenseur le plus proche, etc.).
2. **Disponibilité des Données Historiques** :
   * **Accès aux Données Passées** : Pour les 20 joueurs sélectionnés, il est crucial d'avoir accès à des données historiques détaillées couvrant plusieurs saisons pour une analyse robuste.
3. **Diversité des Sources de Données** :
   * **Intégration des Sources** : Les données peuvent provenir de différentes sources (statistiques de la NBA, suivi des joueurs, etc.), et leur intégration peut être complexe

* **Contraintes Techniques**

1. **Modélisation et Prédiction** :
   * **Complexité des Modèles** : La création d'un modèle de prédiction fiable nécessite de choisir des algorithmes adaptés et de les ajuster correctement.
   * **Entraînement du Modèle** : Les modèles de machine learning nécessitent des quantités significatives de données d'entraînement et de validation.
2. **Précision des Métriques** :
   * **Multiplicité des Facteurs** : De nombreux facteurs influencent la réussite d'un tir (pression défensive, contact, état physique du joueur, etc.). La difficulté réside dans la quantification de ces facteurs et leur inclusion dans le modèle.

* **Contraintes Analytiques**

1. **Sélection des Joueurs** :
   * **Critères de Sélection** : Déterminer les 20 meilleurs joueurs selon des critères objectifs (comme ceux d'ESPN) et s'assurer qu'ils sont encore actifs.
2. **Comparaison et Analyse** :
   * **Normalisation des Données** : Comparer les performances des joueurs nécessite de normaliser les données pour tenir compte des différences de style de jeu, de position, et de temps de jeu.

* **Contraintes Pratiques**

1. **Déploiement et Utilisation** :
   * **Application Pratique** : Le modèle doit être utilisable en pratique, par exemple pour fournir des insights en temps réel ou pour des analyses post-match.
2. **Interprétation des Résultats** :
   * **Compréhension par les Utilisateurs** : Les résultats du modèle doivent être présentés de manière claire et compréhensible pour les entraîneurs, analystes, et autres utilisateurs finaux.

* **Contraintes Éthiques et Légales**

1. **Confidentialité des Données** :
   * **Protection des Données** : Les données utilisées doivent être traitées de manière à protéger la confidentialité des joueurs et à respecter les réglementations sur la protection des données personnelles.
2. **Équité et Biais** :
   * **Élimination des Biais** : Assurer que le modèle ne présente pas de biais systématiques contre certains joueurs ou équipes.

En tenant compte de ces contraintes, le projet peut fournir des analyses précieuses sur les performances de tir des meilleurs joueurs de la NBA, tout en naviguant dans les complexités inhérentes à la collecte, à l'intégration, et à l'analyse de données de haute qualité.

1.5. Méthodologie

Pour mener à bien ce projet de comparaison des tirs de 20 des meilleurs joueurs de la NBA du 21ème siècle et d'estimation de la probabilité de réussite de leurs tirs, il faut suivre plusieurs étapes méthodiques.

Tout d'abord, il est crucial de définir clairement les objectifs, en comparant la fréquence et l’efficacité des tirs selon la situation de jeu, et en estimant la probabilité de réussite d’un tir en fonction de diverses métriques.

Ensuite, les 20 meilleurs joueurs actuels doivent être sélectionnés en utilisant des critères objectifs tels que ceux d'ESPN.

La première étape essentielle est la collecte de données provenant de sources variées (NbaShotLocation, playByPlay, Players etc …). Une fois collectées, l’étape suivante est la ‘Data Quality’ afin de garantir des données intègres, cohérentes et fiables, les données sont nettoyées afin d’éliminer les doublons et gérer les valeurs manquantes, puis intégrées en utilisant des variables communes comme Player ID et Game ID. On vérifiera également la cohérence de la donnée et de son format.

L’analyse exploratoire des données suit, impliquant des statistiques descriptives et des visualisations pour comprendre les distributions et identifier les tendances. L'analyse et la comparaison des joueurs sont effectuées pour déterminer les différences en termes de fréquence et d’efficacité des tirs dans diverses situations et localisations. Les facteurs clés influençant la réussite des tirs sont identifiés.

La segmentation des données par joueur, situation de jeu et localisation sur le terrain est également nécessaire. La modélisation commence par la sélection d'algorithmes de Machine Learning appropriés, tels que la régression logistique ou les réseaux de neurones, et l’entraînement des modèles sur les données d’entraînement. La validation des modèles suit, en utilisant des métriques telles que la précision et l’AUC-ROC pour évaluer la performance. Les modèles sont ensuite appliqués pour estimer la probabilité de réussite des tirs pour chaque joueur, et ces estimations sont comparées avec les fréquences de tirs réussis observées.

Ensuite, les résultats sont documentés, incluant les étapes suivies, la méthodologie, et les résultats obtenus, avec des visualisations pour illustrer les comparaisons et les insights. Un rapport final est rédigé, et une présentation est préparée pour partager les résultats avec les parties prenantes. Enfin, le retour d'expérience est recueilli pour identifier les points d'amélioration, permettant de mettre à jour et ajuster le modèle et l’analyse en conséquence. En suivant ces étapes, le projet peut être structuré de manière rigoureuse, garantissant des résultats précis et significatifs.

2. Travail réalisé

2.1. Environnement de travail

Nous avons choisi le langage de programmation Python pour sa vaste collection de bibliothèques dédiées à la data science. Pour garantir la compatibilité avec divers environnements de travail, nous avons opté pour la version Python 3.

Afin de faciliter la collaboration et le partage efficace de fichiers, nous avons utilisé Git avec un dépôt hébergé sur GitHub. Cette approche nous a permis de partager notre progression avec notre encadrant, lui permettant de nous guider efficacement tout au long du projet.

En ce qui concerne notre méthodologie, nous avons d'abord réalisé une analyse approfondie du sujet et des données ensemble. Par la suite, nous avons décidé de diviser notre travail en explorant différentes méthodes. Nous avons approfondi plusieurs approches, comparé nos résultats, et déterminé l’approche la plus efficace (détails à fournir dans la phase de modélisation).

2.2. Les données

### **2.2.1. Les Ressources**

* Dataset des tirs NBA entre 1997 et 2019 :    [1]
* Dataset des actions de chaque match entre 2000 et 2020 : [2]
* Dataset des bilans d’équipe entre 2014 et 2018 : [3]
* Dataset des joueurs de NBA depuis 1950 : [4]

2.2.2. La structure

La première étape de la modélisation du problème a été de formaliser la structure de données.

**NBA shot Locations :**

|  |  |  |
| --- | --- | --- |
| **Nom de la colonne** | **Description** | **Type informatique** |
| Game ID | Identifiant du match | int64 |
| Game Event ID | Identifiant de l'événement du match | int64 |
| Player ID | Identifiant du joueur | int64 |
| Player Name | Nom du joueur | object |
| Team ID | Identifiant de l'équipe | int64 |
| Team Name | Nom de l'équipe | object |
| Period | Période du match | int64 |
| Minutes Remaining | Minutes restantes dans la période | int64 |
| Seconds Remaining | Secondes restantes dans la période | int64 |
| Action Type | Type d'action (type de tir) | object |
| Shot Type | Type de tir | object |
| Shot Zone Basic | Zone de tir de base | object |
| Shot Zone Area | Zone de tir par secteur | object |
| Shot Zone Range | Portée de la zone de tir | object |
| Shot Distance | Distance du tir | int64 |
| X Location | Coordonnée X du lieu de tir | int64 |
| Y Location | Coordonnée Y du lieu de tir | int64 |
| Shot Made Flag | Indique si le tir a été réussi (résultat à prédire) | int64 |
| Game Date | Date du match | int64 |
| Home Team | Équipe à domicile | object |
| Away Team | Équipe à l'extérieur | object |
| Season Type | Type de saison (saison régulière, playoffs, etc.) | object |

**Player :**

|  |  |  |
| --- | --- | --- |
| **Nom de la colonne** | **Description** | **Type informatique** |
| Player | Le nom & Prénom du joueur | object |
| height | La taille du joueur, en centimètres (cm) | float64 |
| weight | Le poids du joueur, en kilogrammes (kg) | float64 |
| collage | L'université ou le collège où le joueur a étudié | object |
| born | L'année de naissance du joueur | float64 |
| birth\_city | La ville de naissance du joueur | object |
| birth\_state | L'état de naissance du joueur | object |

**NBA Season 2000 2019 :**

|  |  |  |
| --- | --- | --- |
| **Nom de la colonne** | **Description** | **Type** |
| EVENTMSGACTIONTYPE | TYPE\_D\_ACTION\_D\_EVENT | int64 |
| EVENTMSGTYPE | TYPE\_D\_EVENT | int64 |
| EVENTNUM | NUMERO\_D\_EVENT | int64 |
| GAME\_ID | ID\_DU\_JEU | int64 |
| HOMEDESCRIPTION | DESCRIPTION\_DOMICILE | object |
| NEUTRALDESCRIPTION | DESCRIPTION\_NEUTRE | float64 |
| PCTIMESTRING | CHAINE\_DE\_TEMPS\_PC | object |
| PERIOD | PERIODE | int64 |
| PERSON1TYPE | TYPE\_DE\_PERSONNE1 | float64 |
| PERSON2TYPE | TYPE\_DE\_PERSONNE2 | int64 |
| PERSON3TYPE | TYPE\_DE\_PERSONNE3 | int64 |
| PLAYER1\_ID | ID\_JOUEUR1 | int64 |
| PLAYER1\_NAME | NOM\_JOUEUR1 | object |
| PLAYER1\_TEAM\_ABBREVIATION | ABREVIATION\_EQUIPE\_JOUEUR1 | object |
| PLAYER1\_TEAM\_CITY | VILLE\_EQUIPE\_JOUEUR1 | object |
| PLAYER1\_TEAM\_ID | ID\_EQUIPE\_JOUEUR1 | float64 |
| PLAYER1\_TEAM\_NICKNAME | SURNOM\_EQUIPE\_JOUEUR1 | object |
| PLAYER2\_ID | ID\_JOUEUR2 | int64 |
| PLAYER2\_NAME | NOM\_JOUEUR2 | object |
| PLAYER2\_TEAM\_ABBREVIATION | ABREVIATION\_EQUIPE\_JOUEUR2 | object |
| PLAYER2\_TEAM\_CITY | VILLE\_EQUIPE\_JOUEUR2 | object |
| PLAYER2\_TEAM\_ID | ID\_EQUIPE\_JOUEUR2 | float64 |
| PLAYER2\_TEAM\_NICKNAME | SURNOM\_EQUIPE\_JOUEUR2 | object |
| PLAYER3\_ID | ID\_JOUEUR3 | int64 |

|  |  |  |
| --- | --- | --- |
| PLAYER3\_NAME | NOM\_JOUEUR3 | object |
| PLAYER3\_TEAM\_ABBREVIATION | ABREVIATION\_EQUIPE\_JOUEUR3 | object |
| PLAYER3\_TEAM\_CITY | VILLE\_EQUIPE\_JOUEUR3 | object |
| PLAYER3\_TEAM\_ID | ID\_EQUIPE\_JOUEUR3 | float64 |
| PLAYER3\_TEAM\_NICKNAME | SURNOM\_EQUIPE\_JOUEUR3 | object |
| SCORE | SCORE | object |
| SCOREMARGIN | ECART\_DE\_SCORE | object |
| VISITORDESCRIPTION | DESCRIPTION\_VISITEUR | object |
| WCTIMESTRING | CHAINE\_DE\_TEMPS\_WC | object |

**NBA Season 2019 2020 :**

|  |  |  |
| --- | --- | --- |
| **Nom de la Colonne** | **Description** | **Type informatique** |
| URL |  | object |
| GameType | Type de Match (Saison Régulière / Playoff) | object |
| Location | Lieu du Match | object |
| Date | Date du Match | object |
| Time | Heure du Match | object |
| WinningTeam | Equipe victorieuse | object |
| Quarter | Quart-temps | int64 |
| SecLeft | Secondes restantes | int64 |
| AwayTeam | Equipe extérieure | object |
| AwayPlay | Action adverse | object |
| AwayScore | Score équipe extérieure | int64 |
| HomeTeam | Equipe Domicile | object |
| HomePlay | Action domicile | object |
| HomeScore | Score équipe domicile | int64 |
| Shooter | Tireur | object |
| ShotType | Type de tir | object |
| ShotOutcome | Résultat tir | object |
| ShotDist | Distance tir | float64 |
| Assister | Passeur | object |
| Blocker | Contreur | object |
| FoulType | Type de faute | object |
| Fouler | Joueur ayant fait la faute | object |
| Fouled | Joueur ayant subi la faute | object |
| Rebounder | Rebondeur | object |
| ReboundType | Type de Rebond | object |
| ViolationPlayer |  | object |
| ViolationType |  | object |
| TimeoutTeam | Equipe demandant le temps mort | object |

|  |  |  |
| --- | --- | --- |
| FreeThrowShooter | Tireur Lancer Franc | Object |
| FreeThrowOutcome | Résultat Lancer Franc | object |
| FreeThrowNum | Numéro Lancer Franc | object |
| EnterGame | Joueur entrant | object |
| LeaveGame | Joueur sortant | object |
| TurnoverPlayer | Joueur perdant le ballon | object |
| TurnoverType | Type de perte de balle | object |
| TurnoverCause | Cause perte | object |
| TurnoverCauser | Joueur à l'origine | object |
| JumpballAwayPlayer | Joueur Entre-Deux extérieur | object |
| JumpballHomePlayer | Joueur entre-deux domicile | object |
| JumpballPoss | Joueur récupérant le ballon | object |

**NBA Players stats since 1950**

|  |  |  |
| --- | --- | --- |
| **Nom de la colonne** | **Description** | **Type** |
| Year | L'année de la saison NBA. | float64 |
| Player | Le nom du joueur. | object |
| Pos | La position du joueur sur le terrain (ex. : PG pour meneur, SG pour arrière, etc.). | object |
| Age | L'âge du joueur pendant la saison. | float64 |
| Tm | L'équipe pour laquelle le joueur a joué cette saison. | object |
| G | Le nombre de matches joués par le joueur. | float64 |
| GS | Le nombre de matches où le joueur a été titulaire. | float64 |
| MP | Le nombre total de minutes jouées par le joueur. | float64 |
| PER | Une statistique avancée qui mesure la performance d'un joueur par minute. | float64 |
| TS% | Une mesure de l'efficacité au tir prenant en compte les tirs à trois points et les lancers francs. | float64 |
| 3PAr | Le ratio des tirs à trois points par rapport au total des tirs. | float64 |
| FTr | Le ratio des lancers francs par rapport au total des tirs tentés. | float64 |
| ORB% | Le pourcentage de rebonds offensifs captés par le joueur lorsqu'il est sur le terrain. | float64 |
| DRB% | Le pourcentage de rebonds défensifs captés par le joueur lorsqu'il est sur le terrain. | float64 |
| TRB% | Le pourcentage de rebonds totaux captés par le joueur lorsqu'il est sur le terrain. | float64 |
| AST% | Le pourcentage de passes décisives faites par le joueur par rapport aux tirs réussis de ses coéquipiers | float64 |
| STL% | Le pourcentage d'interceptions faites par le joueur lorsqu'il est sur le terrain. | float64 |
| BLK% | Le pourcentage de tirs adverses contrés par le joueur lorsqu'il est sur le terrain. | float64 |
| TOV% | Le pourcentage de balles perdues par le joueur par rapport à ses possessions totales. | float64 |
| USG% | Le pourcentage des actions d'équipe conclues par le joueur lorsqu'il est sur le terrain. | float64 |
| blanl | Variable vide, utilisée comme espace réservé. | float64 |
| OWS | Une estimation du nombre de victoires apportées par le joueur grâce à sa contribution offensive. | float64 |
| DWS | Une estimation du nombre de victoires apportées par le joueur grâce à sa contribution défensive. | float64 |
| WS | Le nombre total de victoires auxquelles le joueur a contribué. | float64 |
| WS/48 | Le nombre de Win Shares par 48 minutes jouées. | float64 |
| blank2 | Variable vide, utilisée comme espace réservé. | float64 |
| OBPM | L'impact du joueur sur le score de son équipe pour 100 possessions en attaque. | float64 |
| DBPM | L'impact du joueur sur le score de son équipe pour 100 possessions en défense. | float64 |
| BPM | L'impact total du joueur sur le score de son équipe pour 100 possessions. | float64 |
| VORP | Une estimation de la valeur du joueur par rapport à un joueur remplaçant moyen. | float64 |
| FG | Le nombre de tirs réussis par le joueur. | float64 |
| FGA | Le nombre de tirs tentés par le joueur. | float64 |
| FG% | Le pourcentage de tirs réussis par rapport aux tirs tentés. | float64 |
| 3P | Le nombre de tirs à trois points réussis. | float64 |
| 3PA | Le nombre de tirs à trois points tentés. | float64 |
| 3P% | Le pourcentage de tirs à trois points réussis par rapport aux tirs tentés. | float64 |
| 2P | Le nombre de tirs à deux points réussis. | float64 |
| 2PA | Le nombre de tirs à deux points tentés. | float64 |
| 2P% | Le pourcentage de tirs à deux points réussis par rapport aux tirs tentés. | float64 |
| eFG% | Une mesure de l'efficacité au tir qui prend en compte le fait que les tirs à trois points valent | float64 |
| FT | Le nombre de lancers francs réussis. | float64 |
| FTA | Le nombre de lancers francs tentés. | float64 |
| FT% | Le pourcentage de lancers francs réussis par rapport aux lancers tentés. | float64 |
| ORB | Le nombre de rebonds offensifs captés par le joueur. | float64 |
| DRB | Le nombre de rebonds défensifs captés par le joueur. | float64 |
| TRB | Le nombre total de rebonds captés par le joueur. | float64 |
| AST | Le nombre de passes décisives effectuées par le joueur. | float64 |
| STL | Le nombre d'interceptions effectuées par le joueur. | float64 |
| BLK | Le nombre de tirs contrés par le joueur. | float64 |
| TOV | Le nombre de balles perdues par le joueur. | float64 |
| PF | Le nombre de fautes personnelles commises par le joueur. | float64 |
| PTS | Le nombre de points marqués par le joueur. | float64 |

**Games\_details :**

|  |  |  |
| --- | --- | --- |
| **Nom de la Colonne** | **Description** | **Type informatique** |
| GAME\_ID | Identifiant Match | int64 |
| TEAM\_ID | Identifiant Equipe | int64 |
| TEAM\_ABBREVIATION | Abréviation Equipe | object |
| TEAM\_CITY | Lieu Match | object |
| PLAYER\_ID | Identifiant joueur | int64 |
| PLAYER\_NAME | Nom joueur | object |
| NICKNAME | Surnom Joueur | object |
| START\_POSITION | Position départ | object |
| COMMENT |  | object |
| MIN | Minutes jouées | object |
| FGM | Tirs réussis | float64 |
| FGA | Tirs tentés | float64 |
| FG\_PCT | % réussite | float64 |
| FG3M | 3 pts réussis | float64 |
| FG3A | 3 pts tentés | float64 |
| FG3\_PCT | % réussite 3 pts | float64 |
| FTM | Lancers Francs réussis | float64 |
| FTA | Lancers Francs tentés | float64 |
| FT\_PCT | % réussite Lancers Francs | float64 |
| OREB | Rebonds Offensifs | float64 |
| DREB | Rebonds Défensifs | float64 |
| REB | Rebonds | float64 |
| AST | Passes décisives | float64 |
| STL | Interceptions | float64 |
| BLK | Contre | float64 |
| TO | Perte de balle | float64 |
| PF | Fautes personnelles | float64 |
| PTS | Points marqués par le joueur | float64 |
| PLUS\_MINUS | Ecarts points | float64 |

**Seasons\_stats :**

|  |  |  |
| --- | --- | --- |
| **Nom de la Colonne** | **Description** | **Type informatique** |
| Unnamed:0 | Index | int64 |
| Year | Saison | float64 |
| Player | Nom Joueur | object |
| Pos | Position Joueur | object |
| Age | Age | float64 |
| Tm | Equipe | object |
| G | Matchs | float64 |
| GS | Matchs débutés | float64 |
| MP | Minutes jouées | float64 |
| PER | Efficacité joueur | float64 |
| TS% | True Shooting % (véritable efficacité au tir) | float64 |
| 3PAr | Réussite à 3 pts | float64 |
| FTr | Réussite LF | float64 |
| ORB% | % Rebonds offensifs | float64 |
| DRB% | % Rebonds défensifs | float64 |
| TRB% | % total rebonds | float64 |
| AST% | % Passes décisives | float64 |
| STL% | % interceptions | float64 |
| BLK% | % contres | float64 |
| TOV% | % pertes de balles | float64 |
| USG% | % de ballons utilisés | float64 |
| blanl | vide | float64 |
| OWS | Offensive Win Shares | float64 |
| DWS | Défensive Win Shares | float64 |
| WS | Win Shares | float64 |
| WS/48 | Win Shares per 48 minutes | float64 |
| blank2 | Entrée vide | float64 |
| OBPM | Offensive Box Plus/Minus | float64 |
| DBPM | Défensive Box Plus/Minus | float64 |
| BPM | Box Plus/Minus | float64 |
| VORP | Value Over Replacement Player | float64 |
| FG | Tirs réussis | float64 |
| FGA | Tirs tentés | float64 |
| FG% | % Réussite | float64 |
| 3P | Idem 3 pts | float64 |
| 3PA | Idem 3 pts | float64 |
| 3P% | Idem 3 pts | float64 |
| 2P | Idem 2 points | float64 |
| 2PA | Idem 2 points | float64 |
| 2P% | Idem 2 points | float64 |
| eFG% | % effectif | float64 |
| FT | Lancers Francs | float64 |
| FTA | Tentatives LF | float64 |
| FT% | % Réussite LF | float64 |
| ORB | Rebonds offensifs | float64 |
| DRB | Rebonds défensifs | float64 |
| TRB | Total rebonds | float64 |
| AST | Passes décisives | float64 |
| STL | Interceptions | float64 |
| BLK | Contres | float64 |
| TOV | Pertes de balle | float64 |
| PF | Fautes personnelles | float64 |
| PTS | Points | float64 |

**Ranking :**

|  |  |  |
| --- | --- | --- |
| **Nom de la Colonne** | **Description** | **Type informatique** |
| TEAM\_ID | Identifiant Equipe | int64 |
| LEAGUE\_ID | Identifiant ligue | int64 |
| SEASON\_ID | Identifiant saison | int64 |
| STANDINGSDATE | Date Match | object |
| CONFERENCE | Conférence | object |
| TEAM | Equipe | object |
| G | Matchs | int64 |
| W | Victoires | int64 |
| L | Défaites | int64 |
| W\_PCT | Pourcentage de victoires | float64 |
| HOME\_RECORD | Résultat à domicile | object |
| ROAD\_RECORD | Résultat à l'extérieur | object |
| RETURNTOPLAY |  | float64 |

2.2.3 Analyse

Une fois la structure bien établie nous nous sommes penchés sur le contenu des don- nées. Afin d’orienter notre approche sur notre objectif nous avons analysé les différentes variables.

Une image contenant ligne, diagramme, cercle, capture d’écran

Description générée automatiquement

Figure  : Relation entre nos jeux de données

Une image contenant texte, capture d’écran, Rectangle, diagramme

Description générée automatiquement

Figure  : Relation des ShotMadeFlag

On constate que la distribution de cette variable est équilibrée dans notre DataFrame. Environ 52% des tirs sont ratés pour 48% de réussis. Un fort déséquilibré dans cette variable aurait posé des problèmes et aurait nécessité d’autres étapes de type rééquilibrage car les algorithmes classiques sont moins efficaces en cas de forte disparité.

Une image contenant texte, diagramme, ligne, Tracé

Description générée automatiquement

Figure  : Distribution du pourcentage de réussite aux tirs

Une image contenant texte, capture d’écran, diagramme, Tracé

Description générée automatiquement

Figure  : Répartition des tirs par distance de tir

Une image contenant capture d’écran, texte, diagramme, Tracé

Description générée automatiquement

Figure  : Répartition des tirs par type de tir

Une image contenant texte, capture d’écran, nombre, diagramme

Description générée automatiquement

Figure Répartition des tirs en fonction de la zone

Le constat est ici assez évident et confirme nos intuitions, plus le tir est effectué proche du panier, plus il est probable qu’il soit réussi.

La Restricted Area étant la zone sous le panier, on y voit que la part des tirs réussis est bien plus important que celle des tirs échoués.

Le constat est inverse pour la zone « Above the Break 3 » qui correspond à tous les 3 points hors ceux effectués dans les coins.

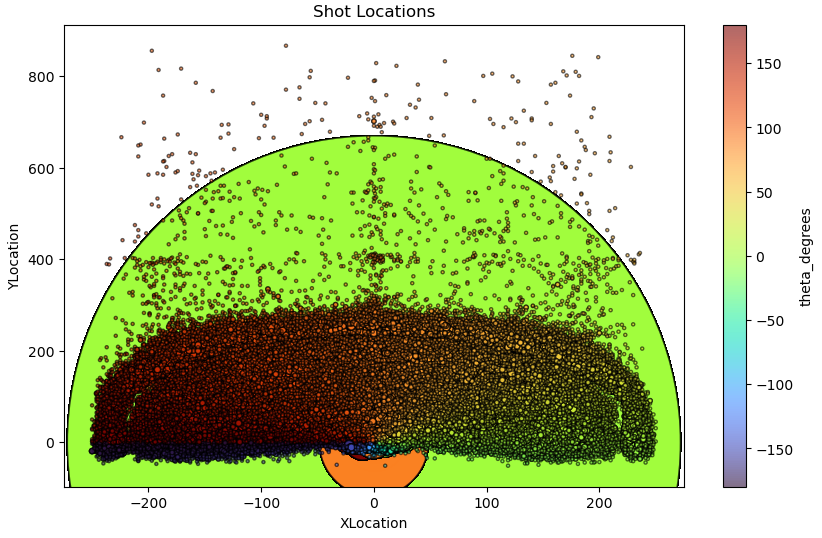


Figure  : Cartographie des tirs

Plus le point est large, plus la fréquence est élevée. La couleur est ici définie en fonction de l’angle du tir calculé à partir des coordonnées X et Y.

Une image contenant texte, capture d’écran

Description générée automatiquement

Figure  : Cartographie des tirs en identifiant les réussis et les manqués

(Nous pouvons remarquer qu’il y a certains angles favorables à un panier réussi que ce soit gauche ou droite avec une symétrie à 0)

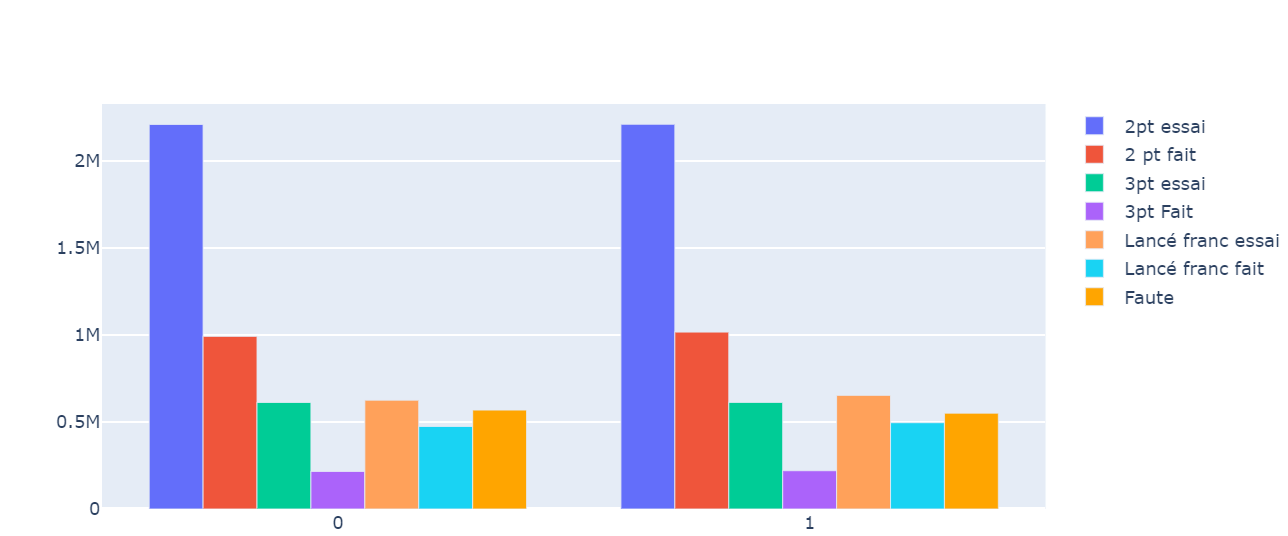


Figure  : Comparaison de différents évènements selon si le match  
 se joue à domicile (0) ou à l’extérieur (1)

Nous pouvons constater qu’il n’y a aucune relation entre le fait que le match se déroule à l’extérieur ou à domicile et les différents évènements intervenant lors d’un match, tel que la réussite ou non d’un tir, le nombre d’essai, et le nombre de fautes.

Une image contenant texte, capture d’écran, ligne, Tracé

Description générée automatiquement

Figure  : Relation entre le temps de jeu et l’efficacité des joueurs

2.2.4 Les erreurs

- Le nom de quelques colonnes était erroné

- Les données relatives au joueur Ray Allen étaient erronées dans la table

2.3. Pre-processing & feature engineering

Une fois le problème bien défini et la structure de données clairement établie, nous avons pu entamer la phase pratique. L'objectif de cette étape de modélisation était de développer un ensemble d'outils à la fois suffisamment génériques pour exploiter les données avec diverses méthodes de data science et suffisamment performants pour obtenir des résultats en un temps raisonnablement court.

Pour ce faire, nous avons utilisé Python ainsi que certaines de ses nombreuses bibliothèques. Les plus notables sont les suivantes :

* Pandas : Utilisé pour la manipulation et l’analyse de données.
* NumPy : Utilisé pour la manipulation de matrice ou pour effectuer des fonctions mathématiques
* Seaborn : Outil de visualisation
* Matplotlib : Outil de visualisation
* Sklearn : Bibliothèque destinée à l’apprentissage automatique

2.3.1. Problèmes rencontrés

- Trouver les 20 meilleurs joueurs qui peuvent être actif dans notre jeu de données

2.3.2. Nettoyage des données

Un aspect essentiel de la science des données est le nettoyage minutieux des données. Cette étape est cruciale pour garantir la qualité et la fiabilité des analyses futures. Ainsi, nous avons mis en place une procédure de nettoyage rigoureuse et exhaustive, basée sur l'analyse préliminaire des données. Cette procédure comprenait la détection et la correction des valeurs aberrantes, la gestion des valeurs manquantes, et l'harmonisation des formats de données. Nous avons également veillé à éliminer les doublons et à corriger les incohérences éventuelles. En procédant ainsi, nous avons assuré que les données utilisées pour les étapes suivantes de modélisation et d'analyse étaient de la plus haute qualité possible. Cette attention au détail dans le nettoyage des données a été fondamentale pour obtenir des résultats précis et significatifs dans nos analyses de data science.

2.3.3. Choix des joueurs

Pour construire notre DataFrame final, nous avons dans un premier temps identifié une liste de 20 joueurs pertinents.

Différentes méthodes ont été testées :

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

Figure : Les 20 joueurs ayant joué le plus de match

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquementUne image contenant texte, capture d’écran, affichage, logiciel

Description générée automatiquementUne image contenant texte, capture d’écran, Caractère coloré

Description générée automatiquementUne image contenant texte, capture d’écran, diagramme, ligne

Description générée automatiquement

Une image contenant texte, capture d’écran, ligne, Police

Description générée automatiquement

Figure  : 20 meilleurs joueurs ayant marqué le plus de points

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

Figure  : Top 20 meilleurs joueurs qui ont mis le plus de paniers

Puis cette liste a finalement été construite « à dire d’expert », et ressemble aux listes que nous avons pu trouver.

Cette liste contient 20 joueurs iconiques de la période, jouant à des postes différents et dont le nombre de tir présents dans les DataFrames offre une volumétrie suffisante pour de futures modélisation.

La liste est la suivante :

Une image contenant texte, Police, logiciel, Logiciel multimédia

Description générée automatiquement

Figure  : Top 20 meilleurs joueurs choisis

Comme nous pouvons le voir sur la capture ci-dessus, cela nous permet de conserver environ 400k lignes avant les différents retraitements que nous allons opérer.

Aux données existantes, nous avons souhaité inclure un certain nombre de champs calculés afin d’enrichir notre jeu de données.

Nous avons donc ajouté les indicateurs suivants :

* LastShotMadeFlag : Booléen indiquant si le joueur a mis son précédent tir lors du match. C’est une sorte d’indicateur de confiance, si le joueur enchaîne les tirs réussis, on peut penser que la probabilité qu’il réussisse le suivant est accrue.
* Theta\_degrees : Angle du tir, créé à partir des coordonnées X et Y.
* HomeGame : Booléen indiquant si le match est joué à domicile ou non. Permet d’inclure l’aspect « environnemental » tel que le niveau sonore de la foule par exemple.
* SecLeft : Temps restant dans le match. Les matchs étant normalisés à 4 Quart-temps, cette variable semble a priori plus facilement compréhensible par l’algorithme qu’un croisement Quart-temps / Minutes restantes / secondes restantes.
* ActionTypeRange : Le nombre de modalités étant trop important dans la variable initiale ActionType, nous avons normalisé cette variable en gardant les 4 plus fréquentes, puis, en rangeant les autres variables dans un champ ‘Autre’.
* FOULER : Si le tir est précédé d’une faute, nom du joueur ayant effectué la faute
* REBOUNDER : Si le tir est précédé d’un rebond, nom du joueur ayant pris le rebond
* TURNOVER : Si le tir est précédé d’une perte de balle, nom du joueur ayant perdu la balle
* PlayerTmScore: Score de l’équipe à laquelle appartient le joueur.
* RebondOffensif / RebondDefensif : Ces deux champs sont déterminés depuis les variables homedescription et visistordescription du jeu de données playbyplay. On fait une analyse des lignes au-dessus et en dessous de celle d'un tir, pour voir s’il existe un événement qui a contribué à la réussite ou au ratage du tir

Quelques opérations ont été appliquées ensuite dans le but de rendre le jeu de données plus consistant.

Dans un premier temps, nous avons observé que les données relatives au joueur Ray Allen étaient erronées dans la table que nous utilisons pour récupérer la taille et le poids des joueurs, nous avons donc appliqué un correctif sur sa taille et sur son poids.

Le jeu de données playbyplay contient l’intégralité des actions qui surviennent au cours d’un match. Nous avons donc uniquement filtré sur ce qui nous intéressait, c’est-à-dire les tirs, qu’ils soient réussis ou manqués. Cependant, les variables SCORE\_MARGIN et SCORE ne sont renseignées que lorsque le score change. Aussi, afin de rendre notre jeu de données plus consistant, nous avons alimenté les champs vides avec la dernière valeur connue lors du match (méthode ffill). La première ligne étant initialisé à SCORE = ‘0 – 0’ et SCORE\_MARGIN = 0 (égalité).

Afin d’éviter d’avoir à répéter ces éléments à chaque csv, nous avons défini une fonction qui l’applique automatiquement.

Enfin, nous avons pris le parti de ne pas conserver les données issues de prolongations, ces données étant à la fois très spécifiques et très marginales (< 0,1%).

A la fin de notre pre-processing, notre DataFrame est le suivant :



Figure  : Les colonnes du DataFrame

Le DataFrame est composé de 36 colonnes et 285k lignes.

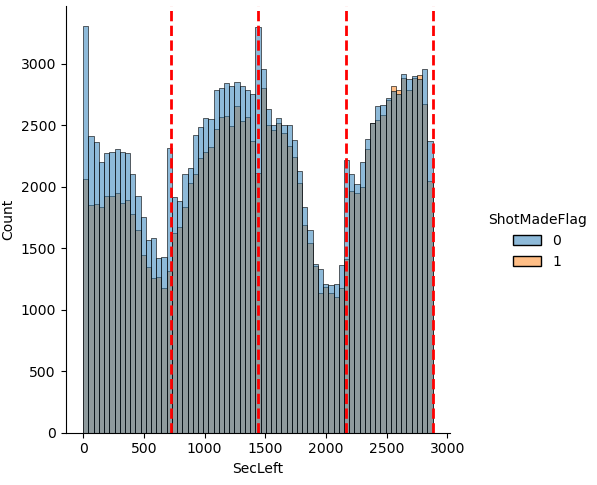


Figure  : Tirs en fonction de notre champ calculé SecLeft

Les pointillés en rouge correspondent à la fin de chaque quart-temps. Nous observons une certaine symétrie.

A la fin du premier et du troisième quart-temps, le nombre de tir chute, ce qui n’est pas le cas dans le deuxième et quatrième quart-temps où ce nombre reste important.

Autre enseignement, les rares fois où la barre orange (tir réussi) est supérieure à la barre bleu (tir raté) correspondent au premier quart-temps, potentiellement dans les instants où la pression est la moins forte. Nous voyons également que plus le match avance, plus l’écart entre les tirs réussis et les tirs manqués augmente. Autrement dit, plus la fin du match approche, plus la pression se fait ressentir. Enfin, on note que les barres sont très hautes à la fin du deuxième et du dernier quart-temps, signe que les joueurs attendent le dernier moment pour tenter le tir.

2.3.4. Prétraitement et constitution des Données

Notre base finale est obtenue après plusieurs opérations de nettoyage au niveau des variables et des données.

Elle résulte de la fusion entre la table 'NBA Shot Locations', où les 20 joueurs sélectionnés sont filtrés, et la table 'Player' issue du csv 'Players\_since\_1950', respectivement sur les variables 'Player Name' et 'Player'.

Ensuite, après la sélection des 20 joueurs dans la table 'Seasons\_Stats', nous avons fusionné cette première table de résultat sur les variables 'Year' et 'Player' (nom du joueur) afin d'avoir leurs stats par saisons.

Après avoir regroupé les différentes tables de 'Play by Play' par concaténation, nous les avons fusionnées avec la dernière table sur les clés 'Game ID' et 'GameEventID' afin d'enrichir notre jeu de données avec des variables nouvelles.

Enfin, nous avons nettoyé cette dernière table en ne conservant que les variables pertinentes et en supprimant les valeurs manquantes.

Une image contenant texte, capture d’écran, ligne, diagramme

Description générée automatiquement

Figure  : Constitution des données de la table finale

3. Modélisation

3.1. Adaptation Daframe

Lors de nos premiers tests de modélisation, nous nous sommes rapidement rendu compte que le dataframe constitué devait être retravaillé, certaines données étaient manquantes et d’autres avaient une corrélation directe avec la variable cible (target).

En effet un leak avait été introduit par inadvertance dans les features et nous permettait d’obtenir un très bonne accuracy (>0,8) pour ce type de problématique. Dans les fichiers playByplay, nous avions récupéré les informations ASSIST, qui permet de savoir si une passe décisive a été effectuée, et BLOCK, qui permet de savoir si un tir a été bloqué avant d’atteindre le panier. Le problème de ces features est que leur présence induit directement la réussite d’un tir ou non. Lorsque l’on peut rattacher une passe décisive au tir, celui-ci est forcément réussi. A contrario, lorsque le tir est contré, celui-ci est forcément raté. Aussi, nous sommes revenus en arrière dans la constitution de notre dataframe et avons retiré ces variables.

Par la suite nous avons décidé d’introduire l’équipe adverse dans le dataframe. En effet certaines équipes étant par nature meilleures en défense, cela peut réduire les performances au tir d’un joueur (notamment grâce à des contres). Pour cela nous avons calculé le champs vs\_team de sorte que si le joueur joue à domicile, alors vs\_team sera égale à AwayTeam, sinon il sera égal à HomeTeam.

Enfin les performances d’un joueur peuvent être différentes d’un match à un autre, selon sa conditions physique (blessure, fatigue, etc.), selon sa condition morale, ou encore selon l’équipe adverse pouvant avoir une tactique défensive contre un joueur précis. Pour cela nous avons également décidé de garder le champs GameID qui nous permet d’identifier un match précis.

Après la reconstitution de ce dataframe nous avons pu passer dans un premier temps à une vision statistique des données, puis à la modélisation.

3.2. Rapport de modélisation

Le dataframe étant désormais finalisé, nous avons pu nous lancer dans la phase de modélisation.

Modéliser ce type de problème est intrinsèquement compliqué. La réussite d’un tir étant par nature difficile à modéliser. Certaines composantes semblent intuitives dans sa réussite, par exemple, la distance par rapport au panier. D’autres seront tout de suite plus abstraites (fatigue du joueur, confiance au moment du tir, …).

Par ailleurs, notre sélection du top 20 des joueurs, ainsi que la création d’un certain nombre de features a abouti à une volumétrie non négligeable, d’environ 300 000 lignes pour une trentaine de colonnes. Aussi, le choix du modèle a dû intégrer ces critères pour donner de bons résultats tout en maitrisant la durée du traitement.

Après plusieurs tentatives pour analyser lesquels donnaient des résultats probants, trois modèles ont été retenus avant optimisation des hyperparamètres :

* **LightGBM**
* **XGBoost**
* **Réseaux de neurones**

En lieu et place du train\_test\_split habituel, nous avons utilisé une approche de validation croisée appelée StratifiedKfold. Celle-ci prend en compte la répartition des classes dans les ensembles de données. L'objectif principal est de garantir que chaque fold (ou pli) de l'ensemble de données utilisé pour l'entraînement et la validation contient une proportion similaire de chaque classe par rapport à l'ensemble de données global.

Dans une validation croisée K-Fold classique, les données sont simplement divisées en K sous-ensembles de manière aléatoire, ce qui peut entraîner une distribution déséquilibrée des classes dans certains sous-ensembles. Cela peut affecter les performances des modèles, notamment sur des jeux de données déséquilibrés, où certaines classes sont sous-représentées.

Enfin, pour l’optimisation des hyperparamètres, nous avons utilisé une librairie intitulée Optuna.

Optuna est une bibliothèque d'optimisation d'hyperparamètres open-source, conçue pour être flexible, efficace et facile à utiliser. Elle permet d'automatiser le processus de recherche des meilleurs hyperparamètres pour les modèles de Machine Learning, en maximisant (ou minimisant) une métrique donnée, comme la précision, la perte ou toute autre métrique d'évaluation.

Optuna repose sur un concept de "study" (étude), qui est un ensemble d'essais (trials). Chaque essai correspond à une exécution de modèle avec un ensemble spécifique d'hyperparamètres. L'utilisateur définit un objectif (une fonction à optimiser) et Optuna se charge de tester différents ensembles d'hyperparamètres pour trouver ceux qui optimisent au mieux cet objectif.

3.2.1. Modèle LightGBM

* **Présentation :**

LightGBM est une implémentation de gradient boosting développée par Microsoft, optimisée pour l'efficacité et la rapidité. Il utilise une approche basée sur les histogrammes pour réduire la complexité temporelle du processus de croissance des arbres. LightGBM est particulièrement adapté pour les grands ensembles de données avec de nombreuses caractéristiques.

Parmi les trois modèles retenus, c’est celui qui offre le meilleur couple Accuracy / rapidité.

* **Caractéristiques principales :**

**Efficacité en mémoire et rapidité d'entraînement :**

Grâce à l'approche par histogramme, LightGBM peut former des modèles plus rapidement et avec une consommation de mémoire moindre par rapport à d'autres méthodes de boosting.

**Support de données dispersées (sparse data) :**

LightGBM gère efficacement les données ayant beaucoup de valeurs manquantes ou des données catégorielles.

**Apprentissage en profondeur (Leaf-wise growth) :** Contrairement aux autres implémentations de boosting qui utilisent une approche de niveau (level-wise), LightGBM construit les arbres de manière à maximiser la réduction de perte (leaf-wise growth). Cela permet une meilleure performance, bien qu'il faille faire attention à éviter l'overfitting.

* **Applications :**

LightGBM est souvent utilisé pour des tâches de classification (binaire, multiclass), de régression, de rang (ranking) et d'autres problèmes en compétition de Machine Learning.

Nous avons lancé 60 études du modèle afin de définir les hyperparamètres, la sortie est la suivante :

[I 2024-07-29 14:52:56,767] Trial 59 finished with value: 0.6601845080168085 and parameters: {'num\_leaves': 23, 'max\_depth': 12, 'learning\_rate': 0.08041752736856972, 'n\_estimators': 302, 'subsample': 0.9411201509994473, 'colsample\_bytree': 0.5932063689592868}. Best is trial 26 with value: 0.6618458163680879.

Best hyperparameters: {'num\_leaves': 113, 'max\_depth': 19, 'learning\_rate': 0.03295825041204072, 'n\_estimators': 389, 'subsample': 0.8162306085513036, 'colsample\_bytree': 0.7835908415381942}

En lançant notre modèle final, les résultats sont les suivants :

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Figure  : Modèle LightGBM - Matrice de Confusion

L’accuracy, que nous souhaitons optimiser est de 0,66. Le modèle donne des résultats satisfaisants sur le ShotMadeFlag = 0 avec un f1-score de 0.71 et légèrement inférieur sur la variable ShotMadeFlag =1 avec un score de 0.6.

Les features importants sont les suivantes :

Une image contenant texte, capture d’écran, Tracé, diagramme

Description générée automatiquement

Figure  : Modèle LightGBM - Les features importants

Une image contenant texte, ligne, capture d’écran, Tracé

Description générée automatiquement

Figure 20 : La Courbe ROC

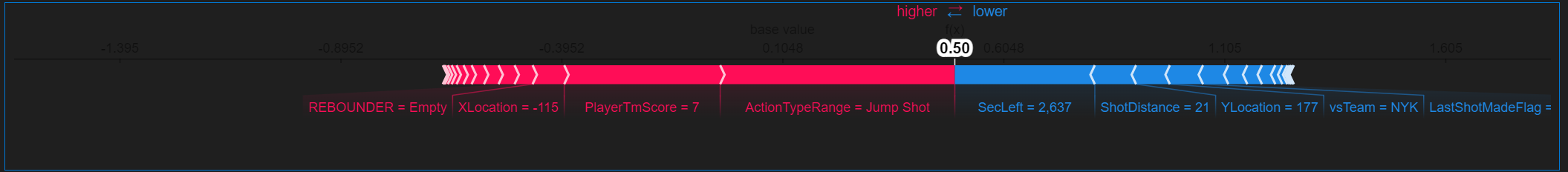


Figure 21 : Les caractéristiques pour une instance individuelle

**SHAP values :**

Nous avons ensuite analysé les shap values.

Les **SHAP values** (SHapley Additive exPlanations) sont une méthode d'interprétabilité des modèles de Machine Learning. Elles permettent d'expliquer les prédictions de modèles complexes en quantifiant l'importance de chaque caractéristique (feature) sur une prédiction donnée. SHAP est basé sur la théorie des jeux, et plus spécifiquement sur la valeur de Shapley, un concept qui provient de l'analyse des contributions équitables des joueurs dans un jeu collaboratif.

Une image contenant texte, capture d’écran, ligne

Description générée automatiquement

Figure 22 : Les Shap values pour toutes les instances

L’analyse des Shap values corrobore par exemple la figure 2.15.

Pour la variable SecLeft, plus celle-ci est faible (la fin du match est proche), plus la feature value est faible (probabilité de rater le tir augmente). Cela est dû à la fatigue du joueur en fin de match, la pression lors du tir si le match est serré, etc.

Dans l’autre sens, si nous regardons la variable ShotDistance, le phénomène inverse est observé. Plus cette variable est faible (le tir est proche du panier), plus la probabilité de réussir le tir augmente (feature value tend vers le rouge).

3.2.2. XGBOOST

* **Présentation :**

XGBoost est un autre modèle de gradient boosting très populaire, souvent utilisé dans des compétitions de Machine Learning comme celles de Kaggle. Il est connu pour ses performances élevées, sa robustesse et sa flexibilité. XGBoost améliore les implémentations classiques du boosting grâce à des optimisations telles que le calcul distribué, la régularisation et la gestion efficace de la mémoire.

* **Caractéristiques principales :**

**Régularisation :** XGBoost introduit une régularisation explicite (L1 et L2) pour éviter le surapprentissage (overfitting), ce qui lui permet d'être plus robuste sur des jeux de données bruyants.

**Gestion des valeurs manquantes :** XGBoost a un mécanisme intégré pour gérer les valeurs manquantes, en adaptant automatiquement les splits d'arbres.

**Parallélisation et calcul distribué :** L'algorithme supporte l'entraînement en parallèle et la distribution du calcul sur plusieurs machines, ce qui le rend idéal pour les grands ensembles de données.

**Optimisation des performances :** XGBoost utilise des techniques d'optimisation basées sur des approximations par gradient pour accélérer le processus de construction des arbres.

* **Applications :**

XGBoost est utilisé dans des problèmes de classification, régression, rang et survie, et est réputé pour ses performances sur les jeux de données structurés/tabulaires.

Les résultats en sorti d’Optuna sont les suivants :

[I 2024-07-29 16:06:43,167] Trial 19 finished with value: 0.6478010664695634 and parameters: {'booster': 'dart', 'lambda': 0.545250915868964, 'alpha': 0.1862682433889052, 'max\_depth': 14, 'eta': 0.0732125737589444, 'gamma': 0.34258422729834104, 'subsample': 0.8260454036152651, 'colsample\_bytree': 0.7269577376450733}. Best is trial 14 with value: 0.6507224187388909.

Best hyperparameters: {'booster': 'dart', 'lambda': 0.7611599117364898, 'alpha': 0.2694478668585406, 'max\_depth': 12, 'eta': 0.08760207662576841, 'gamma': 0.24105373659758264, 'subsample': 0.7806768513525376, 'colsample\_bytree': 0.9515248736781333}

Les features importants sont les suivantes :

Une image contenant texte, capture d’écran, diagramme, Tracé

Description générée automatiquement

Figure 23 : Modèle XGBOOST – Les features importants

Une image contenant texte, ligne, capture d’écran, Tracé

Description générée automatiquement

Figure 24 : La Courbe ROC

Les résultats sont assez proches de ceux obtenus grâce à LightGBM :

Une image contenant texte, capture d’écran, Police, logiciel

Description générée automatiquement

Figure 25 : Modèle XGBOOST - Matrice de Confusion

3.2.3. Réseaux de neurones

* **Présentation :**

Les réseaux de neurones sont des modèles d'apprentissage automatique inspirés du fonctionnement du cerveau humain. Ils sont composés de couches de neurones artificiels (ou unités) organisées en couches d'entrée, couches cachées et couches de sortie. Chaque neurone est une unité de calcul qui reçoit des entrées pondérées, applique une fonction d'activation, et produit une sortie transmise aux neurones des couches suivantes.

* **Caractéristiques principales :**
* **Flexibilité et expressivité :** Les réseaux de neurones peuvent modéliser des relations complexes et non linéaires entre les variables. Ils sont particulièrement puissants pour des tâches telles que la reconnaissance d'images, le traitement du langage naturel (NLP) et les séries temporelles.
* **Capacité de généralisation :** Avec suffisamment de données et une architecture appropriée, les réseaux de neurones peuvent apprendre des représentations générales des données, ce qui leur permet de bien généraliser sur des exemples non vus.
* **Applications :**

Les réseaux de neurones sont utilisés dans une variété de domaines : vision par ordinateur, reconnaissance vocale, traduction automatique, robotique, médecine, finance, et bien d'autres. Ils sont devenus un standard pour les tâches nécessitant une compréhension fine et complexe des données, comme l'identification d'objets dans des images ou la prévision de séquences.

A défaut d’utiliser Optuna, nous avons essayé différents modèles afin d’analyser les résultats.

Dans un premier temps, nous avons implémenté un modèle relativement simple :

Une image contenant texte, capture d’écran, Police, logiciel

Description générée automatiquement

Figure 26 : Couches du modèle

Après 20 Epochs, les résultats semblent relativement satisfaisant :

Epoch 20/20

5683/5683 [==============================] - 8s 1ms/step - loss: 0.6338 - accuracy: 0.6468 - val\_loss: 0.6240 - val\_accuracy: 0.6544

Ensuite, nous avons densifié notre modèle en ajoutant des neurones dans les premières couches ainsi qu’une couche dense en fin de modèle :

Une image contenant texte, capture d’écran, logiciel, Police

Description générée automatiquement

Figure 27 :Couches du modèle

Après 50 Epochs, les résultats sont les suivants :

Epoch 50/50

5683/5683 [==============================] - 11s 2ms/step - loss: 0.6214 - accuracy: 0.6562 - val\_loss: 0.6188 - val\_accuracy: 0.6604

Ceux-ci sont légèrement meilleurs, mais la différence n’est pas très importante.

Nous avons ensuite lancer un troisième modèle :

Une image contenant texte, capture d’écran, logiciel, Police

Description générée automatiquement

Figure 28 : Couches du modèle

Nous y avons notamment inclus des couches **LeakyReLU** (Leaky Rectified Linear Unit) qui sont un type de fonction d'activation non linéaire. Elles sont utilisées pour introduire de la non-linéarité dans les réseaux de neurones, ce qui permet aux modèles d'apprendre des relations plus complexes dans les données et éviter le surapprentissage.

Après 50 Epochs, les résultats sont quasi identiques au modèle précédent :

Epoch 50/50

5683/5683 [==============================] - 11s 2ms/step - loss: 0.6217 - accuracy: 0.6556 - val\_loss: 0.6196 - val\_accuracy: 0.6620

Enfin, nous avons essayé un dernier modèle plus lourd, avec plus d’Epochs :

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

Figure 29 : Couches du modèle

Aussi, nous avons implémenté un callback early\_stopping qui va surveiller la val\_loss et arrêter le modèle si celui-ci ne s’améliore pas après 10 epochs :



Nous avons ensuite lance 100 Epochs, afin d’essayer d’améliorer encore les résultats de notre modèle. Les résultats sont les suivants :

Epoch 53/100

5683/5683 [==============================] - 27s 5ms/step - loss: 0.6239 - accuracy: 0.6610 - val\_loss: 0.6214 - val\_accuracy: 0.6653

Une image contenant texte, capture d’écran, Tracé, diagramme

Description générée automatiquement

Figure 30 : Courbe d’apprentissage

L’accuracy s’améliore légèrement mais semble stagner autour de 0,66.

4. Perspectives d’Amélioration

Au cours de ce projet, un travail conséquent a été mené pour explorer et évaluer divers modèles de Machine Learning. Chaque modèle a été ajusté en optimisant minutieusement les hyperparamètres afin d'obtenir des performances optimales avec le jeu de données disponible. Cette démarche nous a permis d’atteindre un niveau d’accuracy satisfaisant, tout en identifiant les limites inhérentes aux données actuelles et aux modèles testés.

Néanmoins, dans l’optique de poursuivre cette amélioration, plusieurs pistes d’enrichissement peuvent être envisagées. Tout d'abord, l'ajout de nouvelles variables explicatives serait une direction prometteuse. Actuellement, les modèles se basent uniquement sur les données quantifiables, mais de nombreux facteurs qui influencent la réussite d’un tir au basket sont complexes à transcrire numériquement. Par exemple, la fatigue du joueur au moment de l’action, la pression exercée par le défenseur, ou encore l’ambiance générale dans la salle sont des éléments déterminants, mais difficiles à capturer avec les variables traditionnelles. Le défi consisterait donc à développer des indicateurs plus avancés ou à collecter des données supplémentaires qui permettent de mieux modéliser ces aspects contextuels.

Par ailleurs, l’adoption de modèles plus lourds et sophistiqués ou des modèles basés sur des architectures plus complexes, pourrait aussi constituer une piste d'amélioration.

Ces modèles pourraient être en mesure de capturer des relations plus subtiles dans les données et d’améliorer ainsi l’accuracy, bien qu’ils requièrent une puissance de calcul plus importante et un travail supplémentaire pour éviter le surapprentissage.

En somme, bien que le travail accompli ait abouti à des résultats significatifs, les perspectives d'amélioration restent nombreuses et variées, qu'il s'agisse d’enrichir les données disponibles ou de recourir à des modèles plus avancés.

5. Problèmes rencontrés

- Lors de la réalisation de ce projet nous avons pu rencontrer quelques difficultés et doutes.

- Lors de nos tests de modélisation, nous avons constitué différents dataframe, jusqu’à arriver à celui qui a été sélectionné. L’un des dataframe (non sélectionné) nous a permis d’avoir une accuracy « un peu trop bonne » et a mis en doute nos calculs.

Les features étant :

Une image contenant texte, menu, capture d’écran

Description générée automatiquement

Figure 31 : Les features du DataFrame

Dans ce dataframe les champs YLocation et XLocation ont été supprimés, en effet l’angle ayant été calculé et la distance connue, ces informations peuvent être considéré comme redondantes.

L’angle est aussi calculé en valeur absolu, et le distinguo entre la droite et la gauche, ou le centre est connu par la variable Zone.

La distance exacte n’est pas non plus connue, la feature ShotZoneRange nous permet d’avoir les différentes tranches. Et les pourcentages de blocages et de passes décisives pour un joueur au cours d’une saison ont été introduit.

Avec ce dataframe et le model LightGBM nous obtenons une accuracy de 0,88 qui nous parait un peu trop bonne pour être valable.

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Figure 32 : Accuracy du modèle

Une image contenant texte, capture d’écran, Rectangle, Police

Description générée automatiquement

Figure 33 : Matrice de Confusion du modèle

Avec les mean features :

Une image contenant texte, capture d’écran, Tracé, ligne

Description générée automatiquement

Figure 34 : Les features

- De plus parmi les difficultés rencontrées, un test mathématique a été effectué sur les données. En effet nous avons les champs XLocation et YLocation constituant l’abscisse et l’ordonné par rapport au panier, ainsi que le champs ShotDistance correspondant à la distance avec le panier.

Grace à un calcul simple suivant le théorème de Pythagore, Xlocation^2 + YLocation^2 devrait être égale à shotDistance^2. Cependant cette règle n’est pas respectée par quelques 3000 lignes, ce qui met en doutes la cohérence de certaines données.

- Par ailleurs, les contraintes personnelles/professionnelles de chacun ont contribué à des difficultés organisationnelles et logistiques. Certains membres ont pu avoir des problèmes personnels assez important, et le projet a dû s’organiser autrement.

6. Bibliographie

1. <https://www.kaggle.com/jonathangmwl/nba-shot-locations>
2. <https://sports-statistics.com/sports-data/nba-basketball-datasets-csv-files/>
3. <https://www.kaggle.com/nathanlauga/nba-games?select=ranking.csv>
4. <https://www.kaggle.com/drgilermo/nba-players-stats?select=Players.csv>
5. NUMPY. Fundamental package for scientific computing with python. https://numpy.org/
6. PANDAS. Python data analysis library. https://pandas.pydata.org/
7. Seaborn. https://seaborn.pydata.org/
8. Matplotlib. https://matplotlib.org/
9. SKLEARN. <https://scikit-learn.org/stable/>
10. https://optuna.readthedocs.io/en/stable/
11. <https://ieeexplore.ieee.org/document/8845529>
12. <https://lightgbm.readthedocs.io/en/stable/>
13. https://shap.readthedocs.io/en/latest/
14. [Documentation XGBoost](https://xgboost.readthedocs.io/)
15. https://xgboost.readthedocs.io/en/stable/
16. <https://keras.io/>