**Rapport final**

**Projet Data Analysis - Jeux Vidéo**

*Victor Brass*

*Aline Dubucq*

*Issame Herkane*

*Florian Lys*

**Introduction au projet**

**Contexte**

En 2023, les ventes totales de jeux vidéo (toute plateforme et éditeur confondus) représentaient près de [188 milliards de $](https://www.lesechos.fr/tech-medias/hightech/lindustrie-du-jeu-video-repasse-en-marche-avant-en-2023-1973452#:~:text=Sur%20l'ensemble%20de%20l,les%20derni%C3%A8res%20projections%20de%20Newzoo.), soit un marché plus gros que celui de la musique et du cinéma réunis, notamment en France avec [5.5 milliards de €](https://www.leparisien.fr/economie/le-jeu-video-un-marche-plus-important-que-ceux-du-cinema-et-la-musique-reunis-22-10-2023-O544EA2EZ5AVTCHNL5IJOKY4FM.php) de revenus en 2023. L’industrie du jeu vidéo est donc une manne importante, riche en données à exploiter et où la concurrence est forte. Notre ambition à travers ce projet est de proposer une analyse des données du secteur, de reconnaître des corrélations et des disparités entre éditeurs, plateformes, distributeurs et de pouvoir élaborer un algorithme de machine-learning pouvant prédire le nombre de ventes d’un jeu vidéo.

**Objectifs**

Les principaux objectifs de ce projet sont:

1. Exploration, visualisation et pre-processing du jeu de données
2. Entraînement et évaluation des modèles de machine-learning pour la prédiction du nombre de ventes

Aucun des membres du groupe n’est un spécialiste de la data ni un expert pro du jeu vidéo ! Nous avons donc mis à profit toutes nos connaissances acquises au cours de la formation pour mener à bien ce projet. Nous sommes tous les 4 passionnés par le secteur que ça soit d’un point de vue économique ou par loisirs.

**Objectif 1 : Exploration, de data visualisation et de pre-processing des données**

**Compréhension et manipulation des données**

## **Cadre**

L’équipe de DataScientest nous a redirigé vers un jeu de données disponible sur Kaggle, issu d’un scrap du site [vgchartz.com](http://vgchartz.com). Le site répertorie les ventes totales de jeux-vidéos pour toutes les principales plateformes, allant des années 70 à aujourd’hui. Le site fournit également des variables additionnelles pour chaque jeu, tel que l’éditeur, le développeur ou bien encore le genre.

Le jeu de données en question était toutefois quelque peu limité. Le document ayant été créé en 2016, il ne pouvait pas inclure les ventes de jeux sortis plus récemment. La variable développeur manquait également au jeu de données. Nous avons donc fait un travail de recherche supplémentaire pour trouver un scap plus complet.

Au final, nous nous sommes tourné vers un autre jeu de données, toujours issu de vgchartz, allant jusqu’à 2019 et contenant les variables suivantes:

* Le nom du jeu
* Le genre
* Sa notation ESRB
* La plateforme
* L’éditeur
* Le développeur
* Le score de la critique
* Le score des utilisateurs
* Les ventes totales officielles
* Les ventes totales estimées
* Les ventes estimées par marché (Amérique du Nord, PAL, Japon, Autre)
* L’année de sortie

## **Pertinence**

Pas toutes les variables listées ci-dessus sont pertinentes dans le cadre de l’objectif que nous nous sommes fixé. La notation ESRB notamment est utilisée uniquement sur le marché nord-américain et montre un volume important de valeurs manquantes. Nous avons décidé de travailler sans.

Le score de la critique et des utilisateurs a également un intérêt limité. En examinant ces variables de près nous nous sommes rendu compte que la plupart des scores affichés pour chaque jeu sont en réalité issus de un à deux avis distincts. On peut donc difficilement partir du principe que ces données reflètent le sentiment global des joueurs et de la critique. De plus, de nombreux jeux vidéos n’ont pas récolté de score, résultant en un volume important de valeurs manquantes pour ces deux variables. Nous les avons donc également sorties du jeu de données.

Finalement, nous avons fait le choix de retirer les ventes par marché de notre analyse. Avec la popularisation de la dématérialisation sur le marché du jeu vidéo, il est devenu difficile pour vgchartz de systématiquement estimer l’ensemble des ventes réalisées. Dans un [billet publié sur le site](https://www.vgchartz.com/methodology.php), il est expliqué l’impact que ce changement majeur a sur la méthodologie de collecte de données. Depuis 2018, vgchartz n’estime plus les ventes et fournit les chiffres officiels communiqués par les éditeurs. Ces chiffres sont plus souvent à échelle globale que régionale, nous avons donc mis la priorité vers l’analyse et la prédiction des ventes globales.

## **Traitement de la donnée**

Nettoyage de la donnée

Un travail de nettoyage et de pré-traitement des données était nécessaire avant de pouvoir débuter notre analyse.

La première étape était d’unifier les variables “Total Shipped” et “Global Sales”. Comme précisé dans la partie précédente, la popularisation des ventes en dématérialisé à eu un impact sur la méthodologie de collecte de données de vgchartz. Le site a cessé d’estimer systématiquement les ventes de tous les jeux, valeur que l’on retrouve dans la variable “Global Sales”, pour communiquer les ventes officielles à la place, dans la variable “Total Shipped”.

Nous avons regroupé ces deux variables dans une nouvelle variable nommée “Estimated Sales”. Pour se faire, nous avons créé une fonction qui prend la valeur “Global Sales” si elle est renseignée, la valeur “Total Shipped” sinon. Si aucune des deux ne sont renseignées, la variable prend simplement une valeur manquante.

Avec cette nouvelle variable créée, les variables “Total Shipped”, “Global Sales” et chacune des variables représentant les ventes par région ont pu être retirées du jeu de données. La variable “Estimated Sales” représentant notre cible, nous avons supprimé du jeu toutes les lignes où il y figurait une valeur manquante.

La deuxième étape était de gérer les valeurs manquantes dans les variables “Publisher” (éditeur) et “Developer” (développeur). La stratégie de nettoyage ici était de prendre pour chaque jeu ayant une valeur manquante dans une de ces deux colonnes la modale sur l’ensemble des jeux ayant le même nom mais sortis sur des plateformes différentes. Cette approche permet d’éviter la surreprésentation d’un développeur ou d’un éditeur résultant de la prise de la modale de l’ensemble de la colonne. Avec cette méthode, plus de 1000 valeurs manquantes ont pu être complétées pour chacune des deux variables.

Finalement, nous avons supprimé du jeu de données toutes les lignes affichant entre 2 valeurs manquantes ou plus sur l’ensemble des variables.

Avec l’ensemble du travail de nettoyage effectué, nous  sommes rendus à un jeu de données relativement propre. Le nombre de valeurs manquantes par variable est inférieur à 700, pour une table comportant plus de 21’000 entrées.

Ajout de données

Nous avons jugé pertinent d’identifier la franchise à laquelle appartiennent les jeux de notre dataset. La franchise en elle-même est un argument marketing de poids et contribue grandement à la communication autour d’un jeu par les éditeurs. L’ajout d’une telle colonne nous permettra de tester l’impact de la license sur les ventes.

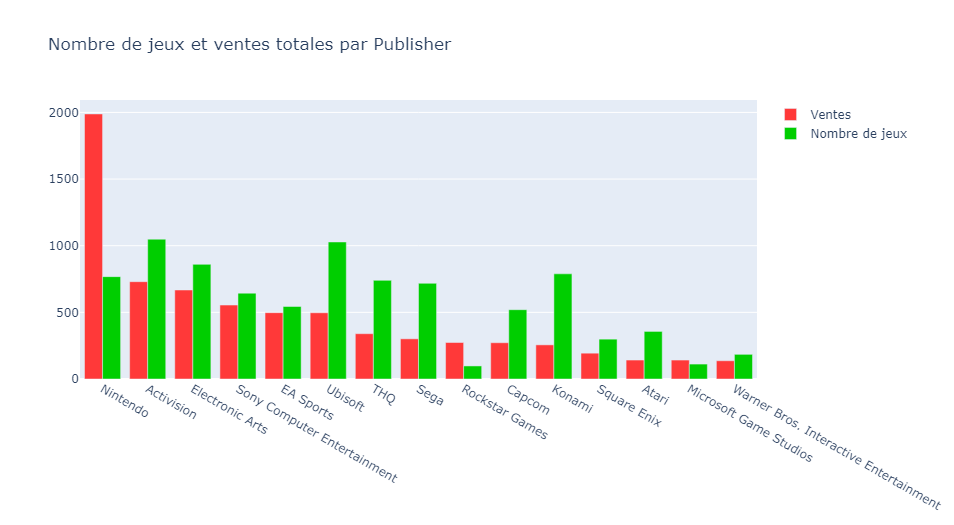
Pour se faire, nous avons récupéré une liste de franchises populaires de l’industrie du jeu vidéo sur WikiPedia, comportant un peu plus d’une centaine d’entrées. Nous avons par la suite développé une fonction basée sur du text mining pour reconnaître la présence de ladite franchise dans le titre de chaque jeu.

Grâce à cette méthode, nous avons ajouté la variable “Franchise” à notre jeu de données qui prend le nom de la franchise, si reconnue.

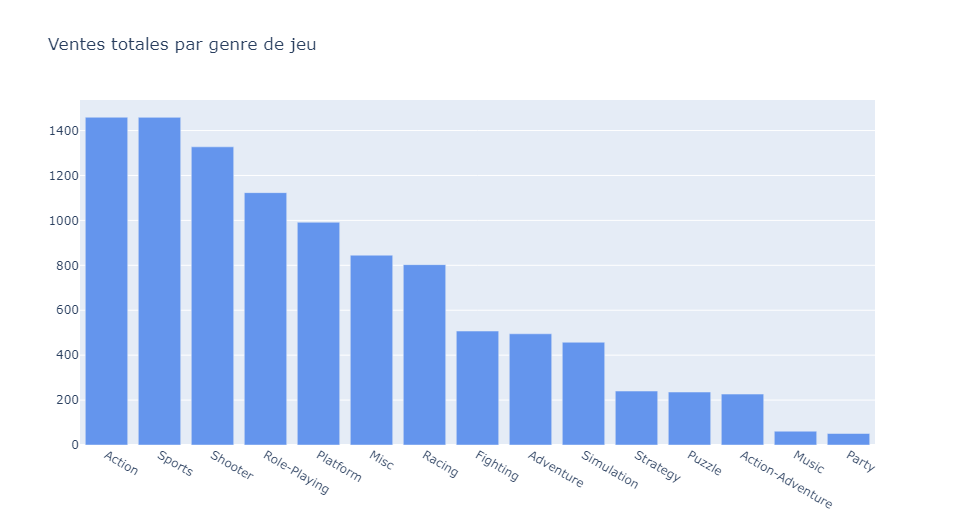
Finalement, afin de mieux comprendre les variables importantes de notre dataset, nous avons ajouté une colonne aléatoire contenant des chiffres random.

## **Visualisations et Statistiques**

Pour la partie visualisation et statistique, nous nous sommes penchés sur les relations entre les variables, notamment le genre du jeu, les publishers, les plateformes et les années. La visualisation était particulièrement centrée sur la compréhension entre le nombre de jeux, les ventes totales par publisher mais aussi quels genres de jeux étaient le plus vendus afin d’avoir un aperçu des forces en présence. Si par déduction et connaissance on peut dire que Nintendo est probablement le plus gros publisher au monde, il faut pouvoir le placer parmi les autres de par le nombre de ventes et le nombre de jeux sortis.

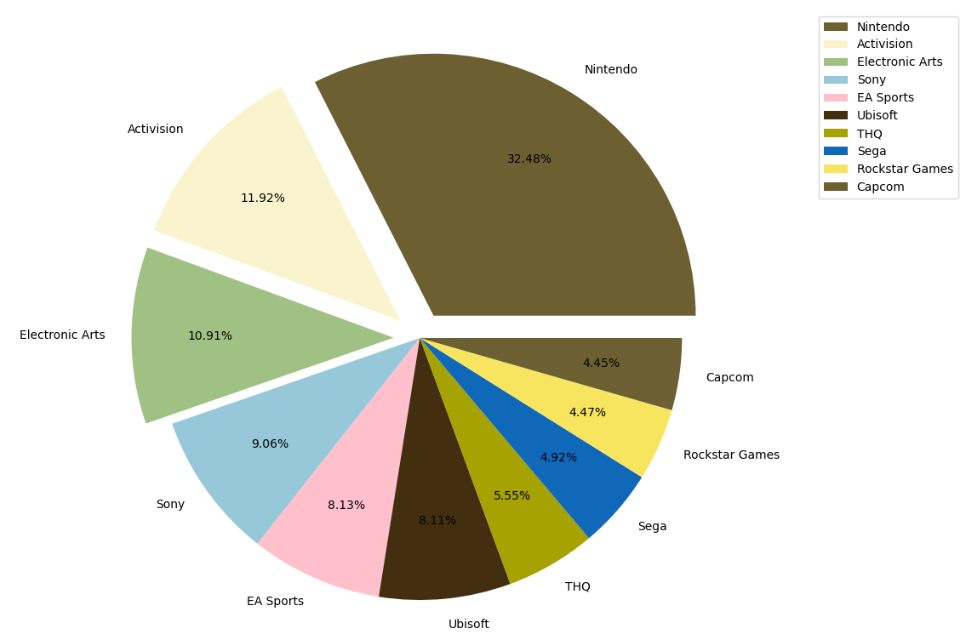


Nintendo est effectivement le publisher qui a les ventes totales les plus importantes mais pas le nombre de jeux sortis. Pour cela, c’est Activision et Ubisoft qui ont produit le plus de jeux.



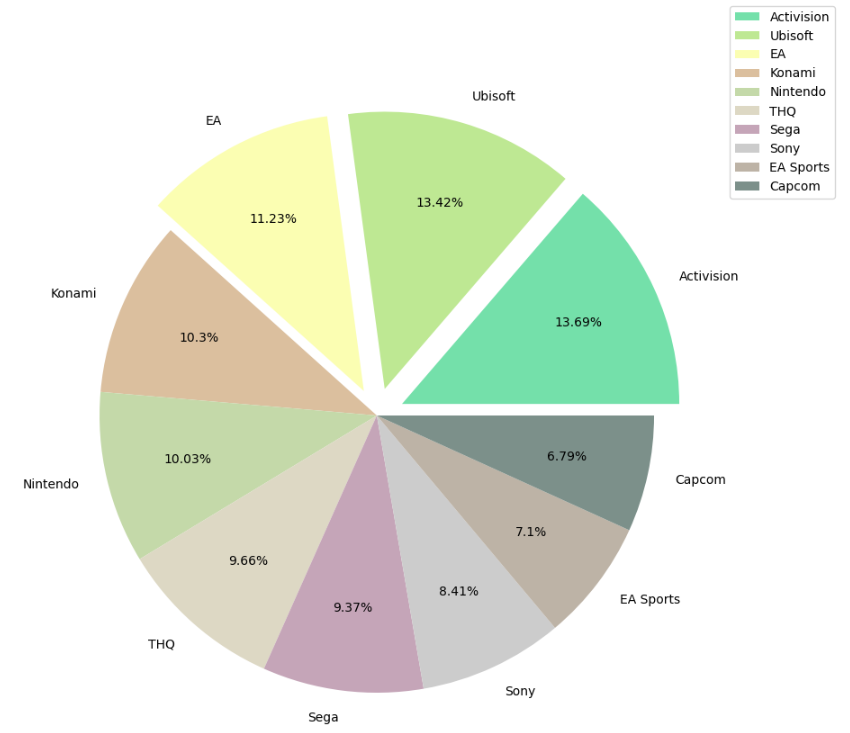
Les jeux d’action et de sport sont les 2 genres qui ont le plus de ventes totales, suivis de près par les jeux de tirs. A l’opposé, les jeux de fêtes et de musique sont les genres qui se vendent le moins.

Si l’on prend la répartition des ventes globales pour les 10 plus grands publishers mondiaux, on obtient ceci:



Nintendo, EA et Activision représentent à eux trois près de 55% des ventes globales de jeux vidéo, sur les 10 premiers éditeurs de jeux en termes de ventes. Nintendo représentant à lui seul 32.5% de parts de marché.

Si on représente le nombre de jeux sortis par publisher, on obtient:

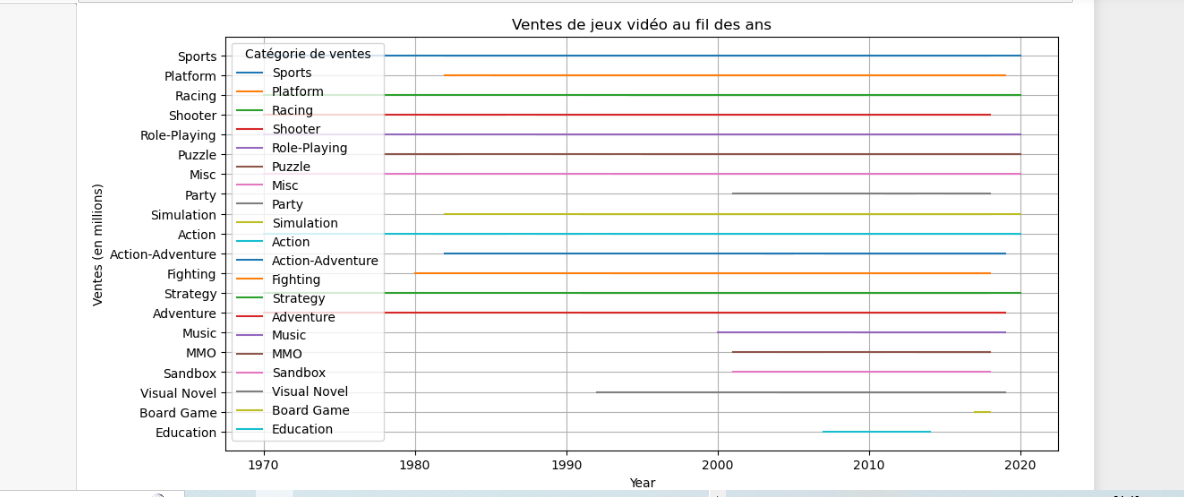
**

Ce qui est intéressant ici, c'est qu'en termes de nombre de jeux sortis, il y a moins de positions monopolistiques. Les 3 éditeurs ayant sorti le plus de titres représentent un peu moins de 40% de la totalité des 10 premiers éditeurs par nombre de jeux. Activision étant le plus gros avec environ 14% des jeux.

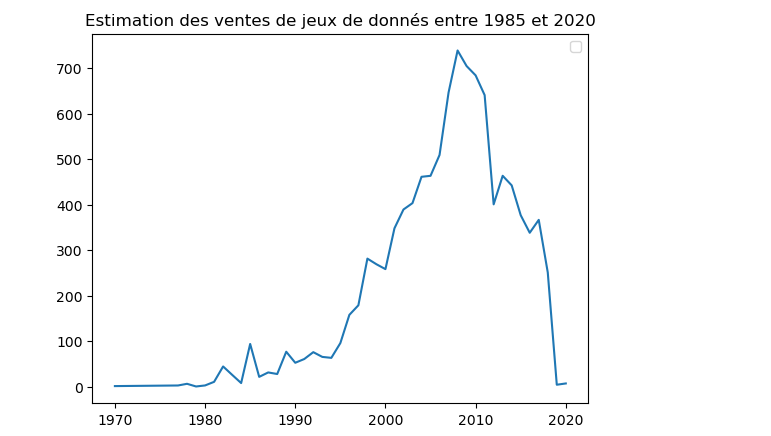
Ainsi, la comparaison avec le camembert précédent permet de montrer que ce n'est pas parce qu'un éditeur sort beaucoup de jeux qu'il a les plus grosses ventes. Nintendo a sorti 3% de titres en moins par rapport à Activision mais a réalisé 20% de ventes globales en plus.

Après ces observations, nous avons effectué des tests statistiques afin de comprendre si ces variables visualisées pouvaient avoir une influence sur notre variable cible, l’estimated sales.

Si on représente les ventes de jeux vidéo au fil des ans , on obtient :

**

*Un graphique qui montre les ventes de jeux vidéo au fil des ans, en distinguant les différentes catégories de ventes par couleur.*

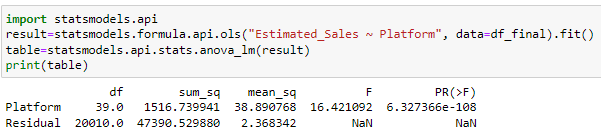
**

*À partir de 1992, il y a eu une augmentation de la vente globale des jeux de données, qui a atteint un pic en 2008.*

Test Anova entre l'estimated\_sales et le type de plateforme

H0 : il n'y a pas d'effet significatif du type de plateforme sur les ventes

H1 : il y a un effet significatif du type de plateforme sur les ventes

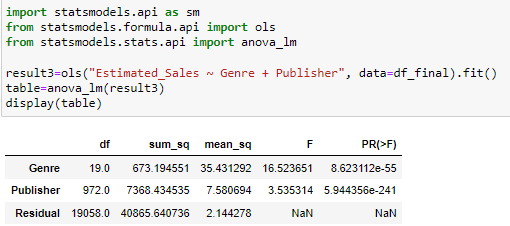


La p-value étant inférieure à 5%, alors on rejette H0 et on conclut H1 : la plateforme a un effet significatif sur les ventes

Test Anova entre l'estimated\_sales et le genre des jeux + le publisher

H0 : il n'y a pas d'effet significatif du genre et du publisher sur les ventes

H1 : il y a un effet significatif du genre et du publisher sur les ventes

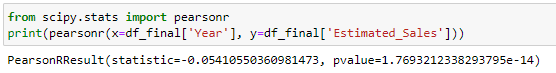


La p-value étant inférieure à 5%, alors on rejette H0 et on conlut H1 : le genre du jeu et le publisher ont un effet significatif sur les ventes

Voyons si l'année de sortie a un effet sur les ventes avec un test de Pearson

H0 : l'année n'a pas de corrélation sur les ventes

H1 : l'année a une corrélation sur les ventes

****

# La p-value étant inférieure à 5%, alors on rejette H0 et on conclut H1 : l'année a un effet sur les ventes

En conclusion, il est clair que notre modèle de machine-learning devra prendre en compte très fortement le genre, la plateforme et le publisher pour déterminer le nombre de ventes possibles.

**Modélisation et évaluation des résultats**

# **Étapes de réalisation du projet**

# **Classification du problème**

Notre problème de machine-learning s’apparente à une régression : prédire un chiffre de ventes à partir de variables catégorielles.

Pour y répondre, nous avons entraîné et testé les modèles ci-dessous, en comparant la performance de chacun sur la base du score de train et du score de test :

* Régression Linéaire
* Arbre de régression
* Random Forest

Chacun de ces modèles ayant des besoins spécifiques de préprocessing, nous allons détailler cette étape séparément, ainsi que leur phase de modélisation et d’optimisation.

## **Pre-processing**

Régression Linéaire

Tout d’abord, nous avons choisis de retirer les variables « Name », « Basename », « Developer » afin de réduire la dimension de notre dataset et le nombre de valeurs uniques. Ainsi, nous avons notre DataFrame X et la variable cible est l’Estimated\_Sales. Les NaNs ont été également retirés (méthode dropna), beaucoup étaient liés à des jeux quasi-inconnus.

La séparation du jeu de données a été faite avec la méthode train\_test\_split pour obtenir un jeu d’entraînement contenant 80% des données et un jeu de test avec 20%.

Nous avons fait un One Hot Encoder sur les variables catégorielles et avons ainsi obtenu un DataFrame avec 1001 colonnes, ce qui est énorme. Mais comme vu avec les tests statistiques, ces colonnes sont importantes, donc souhaitions donc les garder. Ensuite, nous avons procédé à un StandardScaler pour les variables numériques.

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

Description générée automatiquement**

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

Description générée automatiquement

En seconde phase de pré-processing pour la régression linéaire, nous avons entraîné le modèle uniquement sur le ranking des jeux de 1 à 100, le dataset se composait de 66 colonnes, se concentrant ainsi sur les jeux les plus célèbres et les plus vendus.

Enfin pour une dernière modélisation de régression linéaire, nous avons repris le dataset complet en retirant la colonne Plateforme (ce qui a enlevé une centaine de colonnes), réduisant ainsi la taille de notre premier DataFrame.

Nous avons ainsi testé 3 régressions linéaires, sur des DataFrames retravaillés afin qu’ils puissent à la fois être pertinents et exhaustifs mais aussi un peu plus réduits afin que le modèle puisse prédire de la façon la plus pertinente possible.

Arbre de régression et Random Forest

Comme durant le pré-processing pour la régression linéaire, nous avons commencé par retirer les colonnes « Name » et « Basename ». Nous avons toutefois conservé la colonne « Genre » mais avons supprimé la colonne « Rank ». L’objectif était non seulement d’entraîner différents types de modèles, mais également avec différentes variables.

Le jeu de donnée a également été séparé entre un jeu d’entraînement et de test, le jeu de test représentant 20% du jeu total.

Concernant l’encodage, nous avons tout d’abord exécuté un OneHotEncoder sur les colonnes « Franchise », « Genre », « Plateforme », « Publisher » et « Developer ». Cette simple opération a transformé 5 variables qualitatives en 4'116 variables binaires. Cette considérable augmentation de la taille jeu de donnée est principalement dû à la variable « Developer », contenant plusieurs milliers de valeur uniques.

Cette décuplation ne sera pas un problème pour l’entraînement de notre modèle pour autant, l’arbre de régression et la Random Forest sont adaptés pour traiter un tel volume.

Finalement, la variable « Year » a été encodée à l’aide d’un Standard Scaler. Notons que chaque variable a bien été encodée d’abord sur le jeu d’entraînement avant de transformer le jeu de test afin d’éviter une fuite de données entre les deux jeux.

Contrairement à l’approche utilisée pour la régression linéaire, l’entièreté des variables et des lignes ont été utilisées pour l’entraînement des modèles d’arbre de régression et Random Forest.

## **Modélisation et optimisation**

La régression linéaire :

Comme expliqué dans la partie pre-processing, nous avons exploré 3 modèles de régression linéaire en modifiant certains aspects de notre DataFrame afin de tester la pertinence du modèle.

La première régression linéaire, sur le DataFrame entier a eu un score de :

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

Description générée automatiquement

La seconde régression linéaire sur le top 100 des jeux uniquement a eu un score de :

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

Description générée automatiquement

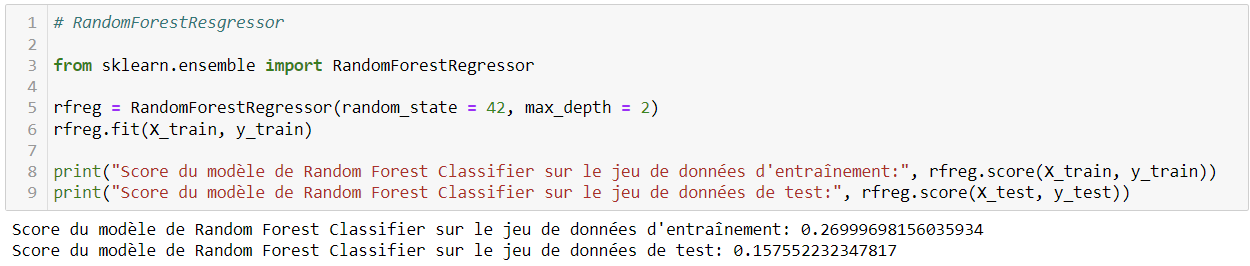
La troisième régression linéaire sur le DataFrame entier moins la variable Plateforme a eu un score de :

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

Description générée automatiquement

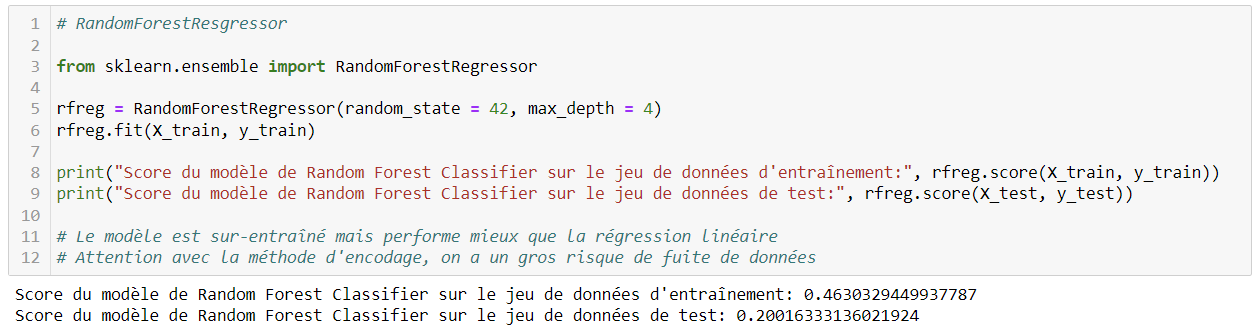
Random Forest :

Le modèle Random Forest a produit des prédictions plus proches du jeu de test comparé à la régression linéaire dès la première tentative.



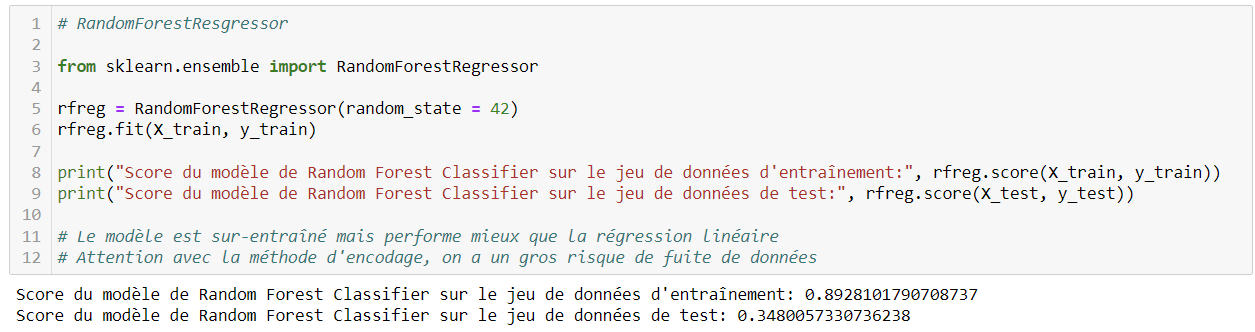
Nous avons observé un score sur le jeu d’entraînement s’élevant à 26%, 15% sur le jeu de test. Ce premier test a été réalisé avec une profondeur maximale de 2 pour éviter un temps de calcul trop long.

Ce résultat reste malheureusement trop bas, nous avons donc fait une deuxième tentative, cette fois-ci avec un profondeur maximale de 4.



Le score sur le jeu de test est sensiblement meilleur, s’élevant à 20%. Le score sur le jeu d’entraînement augmente par contre considérablement, montant à 46%.

Une dernière tentative a été faite sans indiquer au modèle la profondeur maximal. Dans ce cas de figure, le modèle continue de s’entraîner et de faire des embranchements jusqu’à ce que chaque feuille soit pur. On considère comme pur une feuille qui contient une valeur unique, qui ne peut pas s’embrancher d’avantage.



Cette augmentation du nombre du nombre d’embranchements a eu pour effet d’améliorer considérablement le score sur le jeu de test, s’élevant maintenant à 34%. Le score sur le jeu d’entraînement a lui par contre explosé, affichant près de 90%.

Augmenter le nombre d’embranchement dans le modèle de Random Forest a pour effet de sur-entraîner le modèle. Certes, le score sur le jeu de test augmente lui aussi, mais pas suffisament. Passé un certain stade, ce sur-entraînement devient contre productif.

Nous estimons que garder la profondeur maximale à 4 noeuds rend le résultat le plus optimal en utilisant ce modèle.

Arbre de régression

* Décrivez celui / ceux que vous avez retenu et pourquoi ?
* Avez-vous utilisé des techniques d’optimisation de paramètres de type Grid Search et Validation Croisée ?
* Avez-vous testé des modèles avancés ? Bagging, Boosting, Deep Learning… Pourquoi ?

## **Interprétation des résultats**

* Avez-vous analysé les erreurs de votre modèle ?
* Cela a-t-il contribué à son amélioration ? Si oui, décrivez.
* Avez-vous utilisé des techniques d’interprétabilité de type SHAP, LIME, Skater… (Grad-CAM pour le Deep Learning..)
* Qu’est ce qui a (ou non) engendré une amélioration significative de vos performances ?

**Conclusion et bilan**

# 

# **Difficultés rencontrées lors du projet**

* Quel a été le principal verrou scientifique rencontré lors de ce projet ?
* Pour chacun des points suivants, si vous avez rencontré des difficultés, détaillez en quoi elles vous ont ralenti dans la mise en place de votre projet.
* Prévisionnel : tâches qui ont pris plus de temps que prévu, etc.
* Jeux de données : acquisition, volumétrie, traitement, agrégation, etc.
* Compétences techniques / théoriques : timing d’acquisition des compétences, compétence non proposée en formation, etc..
* Pertinence : de l’approche, du modèle, des données, etc..
* IT : puissance de stockage, puissance computationnelle, etc.
* Autres

# **Bilan**

* Détaillez quelle a été votre contribution principale dans l’atteinte des objectifs du projet.
* Avez-vous modifié le modèle depuis la dernière itération ? Si oui, détaillez.
* Présentez les résultats obtenus et comparez-les au benchmark
* Pour chacun des objectifs du projet, détaillez en quoi ils ont été atteints ou non.
* S’ils ont été atteints, dans quel(s) process(es) métier(s) votre modèle peut-il s’inscrire ? Détaillez.

**Suite du projet**

* Quelles pistes d’amélioration suggérez-vous pour augmenter les performances de votre modèle ?
* En quoi votre projet a-t-il contribué à un accroissement de la connaissance scientifique ?

## **Bibliographie**

* Sur quels éléments bibliographiques (articles de recherches, blog, livres, etc) vous êtes vous appuyé pour réaliser votre projet ?

# **Annexes**

Lien Git :