Research Project at :

Ecole des Mines de Nancy

Aymane MOATAZ

Prédiction de la qualité d'un film avant sa sortie, en analysant investissements et engagement de spectateurs.

Tutor :

Dominique Benmouffek

Remerciements

Je tiens à exprimer ma profonde reconnaissance à tous ceux qui m'ont donné la possibilité de réaliser ce rapport. Je tiens à remercier tout particulièrement ma tutrice, Mme Dominique Benmouffek, dont les suggestions et les encouragements m'ont aidé à coordonner mon projet et à rédiger ce rapport.

Résumé

Data really powers everything that we do. — Jeff Weiner

1 Problématique

1.1 Modélisation de la qualité d'un film :

La modélisation de la qualité d'un produit par score numérique réel et bornée est une tâche significative dans le monde digital où plusieurs données entrant en jeu et disponibles sous différentes formes et dimensions sont utilisés pour refléter une information critique sur un service, ici il s'agit de prédire la qualité d'un film avant sa sortie qui est une information importante qui réduit l'insatisfaction après dépenses temporelles et matérielles de spéctateurs et a pourra donc prédir son futur investissement . Dans le monde du cinéma, la métrique la plus célèbre qui reflète la qualité en prenant en compte les retours d'utilisateurs est le score IMDB du site imdb.com.

1.2 Modélisation du score

Le score est calculée par la somme pondérée de score de critiques et d'utilisateurs:

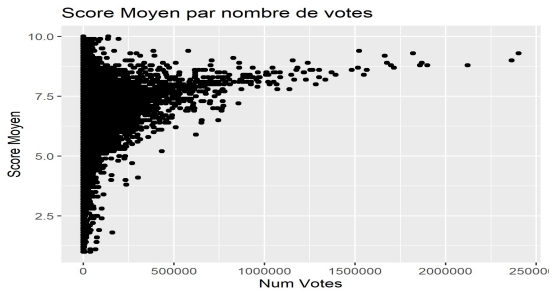
$$imdbscore = \sum_{\ell=0}^{nc-1} C^{\ell} imdbscore_c^{\ell} + \sum_{\ell=0}^{ns-1} U^{\ell} imdbscore_u^{\ell}$$

avec $C_j > U_i$ et $imdbscore_c$, $imdbscore_u$, nc et nu reflètent respectivement le score attribué par utilisateur, le score donné par spéctateur, le nombre de critiques et d'utilisateurs.

Avis de spéctateurs :

IMDb publie des moyennes de votes pondérées plutôt que des moyennes de données brutes. La façon la plus simple de l'expliquer est que, bien que le site accepte et prend en compte tous les votes reçus par les utilisateurs, tous les votes n'ont pas le même impact (ou "poids") sur la note finale. Lorsqu'une activité de vote inhabituelle est détectée, un autre calcul de pondération peut être appliqué afin de préserver la fiabilité de notre système.

Les utilisateurs enregistrés sur le site IMDb peuvent voter (de 1 à 10) pour chaque titre publié dans la base de données. Les votes individuels sont ensuite agrégés et résumés sous la forme d'une note IMDb unique, visible sur la page principale du titre. Par "titre sorti", nous entendons que le film (ou la série TV) doit avoir été projeté publiquement au moins une fois (y compris en festival). Les utilisateurs peuvent mettre à jour leurs votes aussi souvent qu'ils le souhaitent, mais tout nouveau vote sur le même titre écrasera le précédent, il s'agit donc d'un vote par titre et par utilisateur.



La plateforme étant orientée pûrement vers le cinéma il est envisageable qu'un large nombre de votes sur un film soit fortement corrélés à son succès mais également au score refletant sa qualité comme le montre la figure ci-dessus. Un nombre de vote non significatif cependant ne reflètent pas grand chose sur la qualité.

1.3 Remarques:

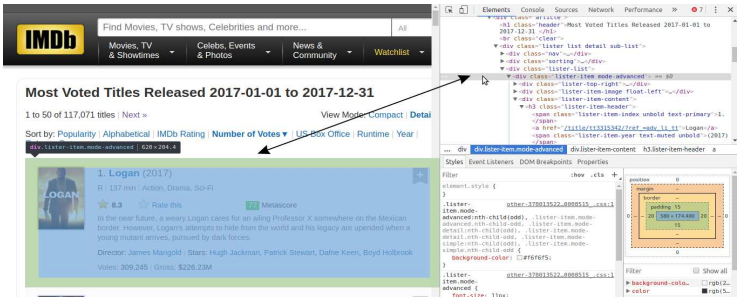
Par cette première modélisation du score, nous avons pu comprendre que le score proposé par IMDB reflète la qualité d'un film, nous avons également saisi la significativité des retours d'utilisateurs, le score attribué nécessite des enregistrements dans le site qui ont suffisament nombreux au contraire de l'engagement, d'autres statistiques potentiellement significatives ne sont pas disponibles sur le site de l'étude. C'est deux constats ont motivé la recherche de données dans d'autres sites.
Cette collecte de données va être détaillée dans la partie suivante de l'étude.

2 Collecte et Préparation des données :

Pour trouver des données dans un projet de machine learning, nous pouvons s'appuyer sur des bases de données SQL et NoSQL, des API ou des ensembles de données CSV prêts à l'emploi. Cependant, il n'est pas toujours possible de trouver un ensemble de données pour répondre à la problématique posée, ce qui est notre cas, les bases de données ne sont pas tenues à jour et les API sont coûteuses.

Les données recherchées se trouvent dans des sites Web. La solution est donc le web scraping, nous avons donc gratter plusieurs pages Web avec Python en utilisant BeautifulSoup et des requêtes pour récupérer toutes les données potentiellement nécessaires, ces dernières ont été stockés dans des fichiers json par la suite afin de les préparer au nettoyage et à l'analyse exploratoire.

2.1 Collecte des données :



La première partie vise à collecter les données des sites web imdb et numbers : informations sur le casting, réalisateurs, sociétés de production, récompenses, genres, budget, brut, description, imdbrating, etc. Pour avoir une idée des retours des utilisateurs et des critiques, nous avons collecté les variables num_critic_reviews et num_user_reviews qui calculent respectivement le nombre de retours écris sur un film par critique et par les utilisateur. Nous avons également collecté les intercations facebook. Les variables director_fb_likes (réalisateur) et movie_fb_likes l, ainsi que celles des 3 acteurs les plus célèbres de chaque film.

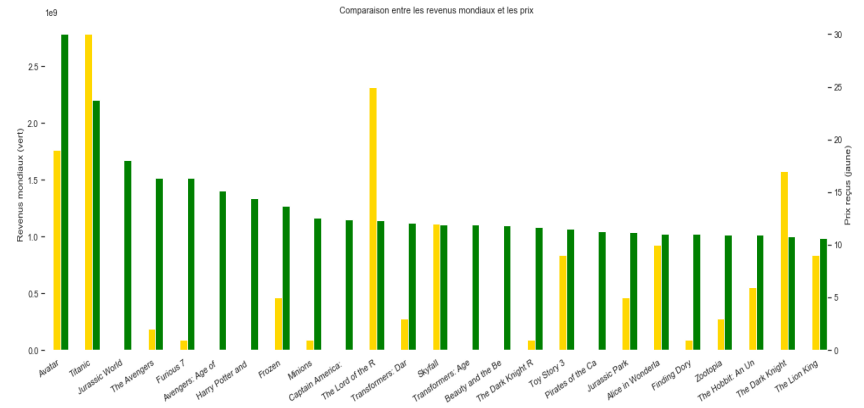
2.2 Préparation des données

Après organisation et préparation des données collectés par scraping, nous avons construit une dataset de dimension $(5112, 108)$

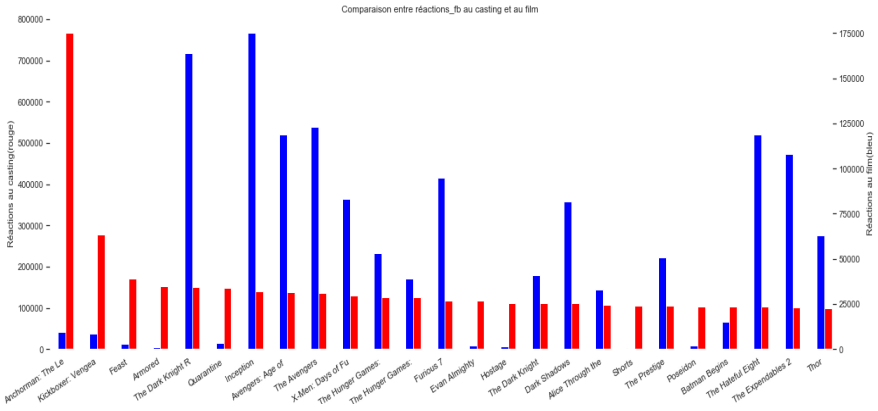
3 Analyse exploratoire de données - EDA

Le données collectées des sites ont été stockées dans des fichiers json. La bibliothèque Pandas a permis de préparer la data frame pour l'utilisation. Le jeu de données contient de nombreuses informations : une première collection des retours; nombre de commentaires par utilisateur, par critique, les réactions au réalisateur, au film et aux 3 premiers acteurs ainsi que les titres, descriptions des films, les réactions totales au casting, le budget de production, les prix reçus et les revenus nationaux et mondiaux génerés.
Nous constaterons clairement que certains paramètres sont correlés et donc tous les paramètres ne sont pas forcément utiles et peuvent être correlés. Nous développerons aussi une idée de l'importance des retours des utilisateurs qui motivera la deuxième section de l'étude.

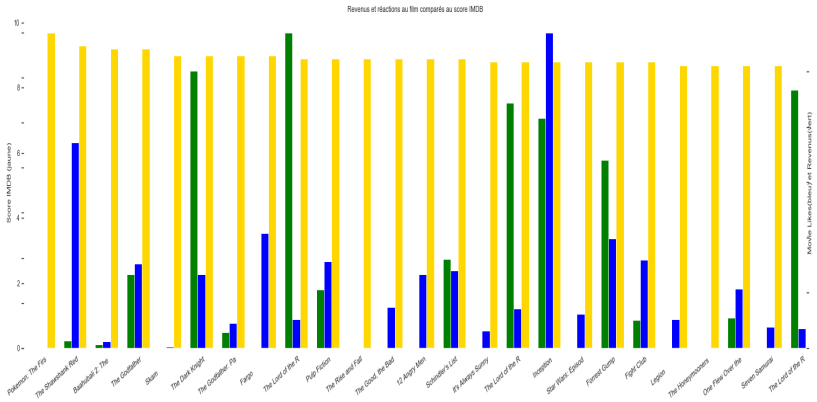
3.0.1 Comparaison des revenus mondiaux générés et les prix reçus



3.0.2 Comparaison des réactions utilisateurs au casting et au film



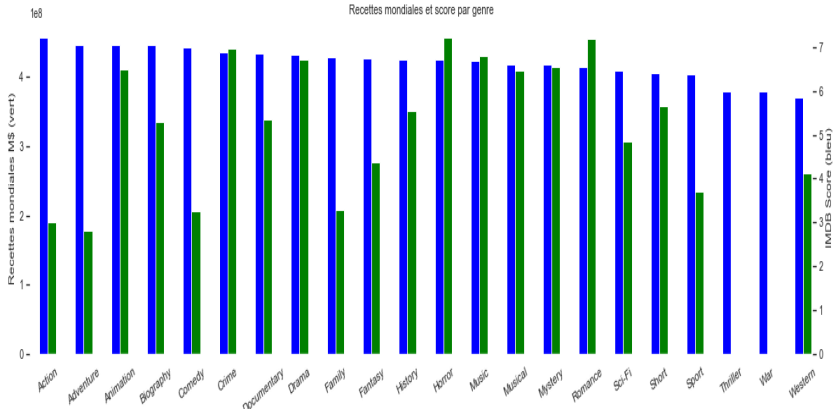
3.0.3 Influence de la réaction aux films et des revenues générés sur le score



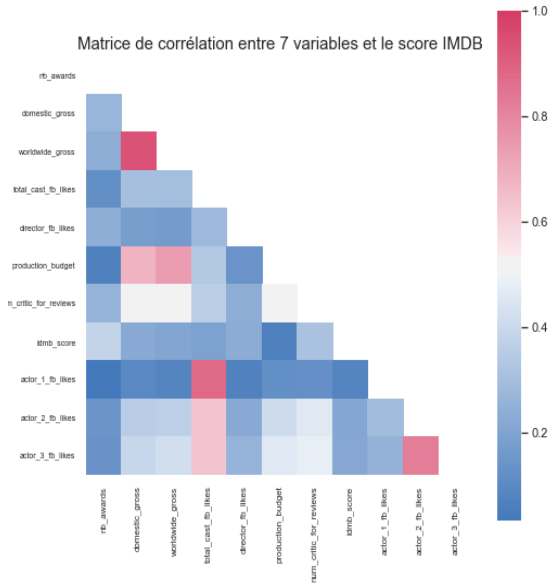
3.0.4 Score et Revenus par genres cinématographiques

Recettes par genre cinématographique	
Genre	Recettes (M)
Aventure	460
Animation	451
Sci-Fi	440

Score par genre cinématographique	
Genre	IMDB score
Biographie	7.3
Guerre	7.1
Drame	7
Mystère	7
Histoire	7



3.0.5 Matrice de corrélation



3.0.6 Conclusion et choix des paramètres influents

Comme on peut le voir dans les images ci-dessus, le score imdb est corrélé au nombre de récompenses et aux recettes brutes mais pas vraiment au budget de production et au nombre de likes facebook du casting. Évidemment, les recettes nationales et mondiales sont fortement corrélées. Cependant, plus

le budget de production est important, plus les recettes brutes le sont aussi. Comme le montre le carnet de notes, le budget n'est pas vraiment corrélé au nombre de récompenses. Ce qui est amusant, c'est que la popularité du troisième acteur le plus célèbre est plus importante pour le score IMDB que la popularité du plus célèbre (corrélation 0,2 contre 0,08).

4 Modèle prédictif

4.1 Théorie

4.1.1 Définitions

Avant de formaliser le modèle, quelques définitions s'imposent. Dans tout le document, nous supposons que l'échantillon d'apprentissage $D_n = (X_1, Y_1), ...(X_n, Y_n)$ de variables aléatoires indépendentes et identiquement distribuées $\mathbb{R} \times [0,1]^d$ $(d \geqslant 2)$ avec la même distribution qu'une paire générique indépendante (X, Y) satisfaisant $\mathbb{E}Y^2 < \infty$. L'espace $[0,1]^d$ est doté de la métrique euclidienne standard. Pour un x fixe $\in [0,1]^d$ notre objectif est d'estimer la fonction de régression $r(x) = \mathbb{E}[Y|X = x]$ en utilisant les données D_n. L'estimateur de la fonction de régression r_n est consistent si $\lim_{n\to\infty} \mathbb{E}[r_n(X) - r(X)] = 0$

Random Forests

Formellement, une forêt aléatoire est un prédicteur constitué d'une collection d'arbres de régression de base randomisés $\{r_n(x, \Phi_m, D_n), m \geqslant 1\}$, où $\Phi_1, \Phi_2, ...$ sont des sorties indépendentes et identiquement distribuées d'une variable de randomisation Φ. Ces arbres aléatoires sont combinés pour former l'estimation de régression agrégée $\overline{r_n}(X, D_n) = \mathbb{E}_\Phi[r_n(X, \Phi, D_n)]$, où $\mathbb{E}_\Phi$ désigne l'espérance par rapport au paramètre aléatoire, conditionnellement à X et à l'ensemble de données D_n. Dans la suite, pour alléger un peu la notation notation, nous omettrons la dépendance des estimations dans l'échantillon, et nous écrirons par exemple $\overline{r_n}(X)$ au lieu de $\overline{r_n}(X, Dn)$. Dans la pratique, l'espérance ci-

dessus est évaluée par Monte Carlo, c'est-à-dire en générant M arbres aléatoires (généralement grands) et en prenant la moyenne des résultats individuels (cette procédure est justifiée par la loi des grands nombres.

4.1.2 Architècture du modèle

Nous supposerons que chaque arbre aléatoire individuel est construit de la manière suivante. Tous les nœuds de l'arbre sont associés à des cellules rectangulaires de telle sorte qu'à chaque étape de la construction de l'arbre, la collection de cellules associées aux feuilles de l'arbre (c'est-à-dire les nœuds externes) forme une partition de $[0,1]^d$. La racine de l'arbre est $[0,1]^d$. La procédure suivante est ensuite répétée $[log_2 k_n]$ fois, $kn \geqslant 2$ un paramètre déterministe, fixé au préalable par l'utilisateur, et dépendant éventuellement de n.

- À chaque nœud, une coordonnée de $X = (X^{(1)}, ..., X^{(d)})$ est sélectionnée, avec la j-ième caractéristique ayant une probabilité $p_{nj} \in (0,1)$ d'être sélectionnée.
- À Chaque nœud, une fois la coordonnée sélectionnée, le partage se fait au milieu du côté choisi.

Chaque arbre aléatoire $r_n(X, \Phi)$ produit la moyenne sur tous les Y_i pour lesquels les vecteurs X_i correspondants tombent dans la même cellule de la partition aléatoire que X.
Nous annalysons la décomposition biais–variance :
$\mathbb{E}[\overline{r_n}(X) - r(X)]^2 = \mathbb{E}[\overline{r_n}(X) - \tilde{r_n}(X)]^2 + \mathbb{E}[\tilde{r_n}(X) - r(X)]^2$

Nous supposerons dans notre cadre que la fonction de régression cible $r(X) = \mathbb{E}[Y|X]$, qui est initialement une fonction de $X = (X^{(1)}, ..., X^{(d)})$ ne dépend en fait que d'un sous-ensemble non vide S des d caractéristiques. En d'autres termes, en laissant $X_S = (X_j : j \in S)$ et $S = Card(S)$, on a $r(X) = \mathbb{E}[Y|XS]$.

4.1.3 Effet d'aggrégation

La variance de l'estimation des forêts, après décomposition du compromis biais variance est $\mathcal{O}(k_n/n(logk_n)^{S/2d})$. Ce résultat est intéressant en soi puisqu'il montre l'effet de l'agrégation sur la variance de la forêt. Pour comprendre cette remarque, rappelons que l'on prouve la cohérence des arbres individuels (aléatoires ou non) en laissant le nombre de cas dans chaque nœud terminal devenir grand avec une variance typique de l'ordre k_n/n.
Ainsi, pour de tels arbres, le choix $k_n = n$ (c'est-à-dire environ une observation en moyenne dans chaque nœud terminal) n'est manifestement pas adapté et conduit à un overfitting et à une explosion de la variance. D'autre part, la variance de la forêt est de l'ordre de ordre $k_n/n(logk_n)^{S/2d}$. Par conséquent, en considérant $k_n = n$, la variance est de l'ordre de $1/(logn)^{S/2d}$ une quantité qui tend toujours vers 0 lorsque n augmente !

4.2 Implémentation du modèle

Nous possédons des données labellisées, nous utiliserons donc Random Forest pour essayer de prédire le score de qualité défini. A l'aide de la bibliothèque scikit learn nous pourrons tester d'autres modèles adaptés à la problématique pour validé l'intuition fondée sur une base théorique. Premièrement, nous divisons aléatoirement notre jeu de données en une partie d'entraînement et une partie de test. Cette division permettra a fortioti de mesurer la performance de notre modèle.

Random Forest moins formellement est une méthode de bagging (nous génèrons des données additionnelles depuis les données d'entraînement en utilisant des combinaisons à répétitions), qui consiste à considérer un ensemble d'arbres appris chacun sur un échantillonage aléatoire de la base d'exemples; la prédiction se fait par vote majoritaire. L'algorithme effectue un apprentissage sur de multiples arbres de décision entrâinés sur des sous-ensembles de données légèrement différent. Les valeurs obtenues pour ce modèle pour un nombre d'estimateurs de 300:

Score Random Forest	
Données	Score
Entrainement	0.934
OOB Score	0.514

4.2.1 Tuning

Nous pouvons constater un écart entre le score sur les données d'entrainement et les données test, qui est assez important mais la valeur OOB score reste néanmoins acceptable. Nous avons changé le paramétrage de ce modèle et utilisé l'estimation Ou-of-Bag. Cette erreur est aussi précise que l'erreur sur un test set de la taille du training set. Le nombres d'estimateurs va être décider ultérieurement par étude de validation croisée.

4.3 Vérification du choix théorique

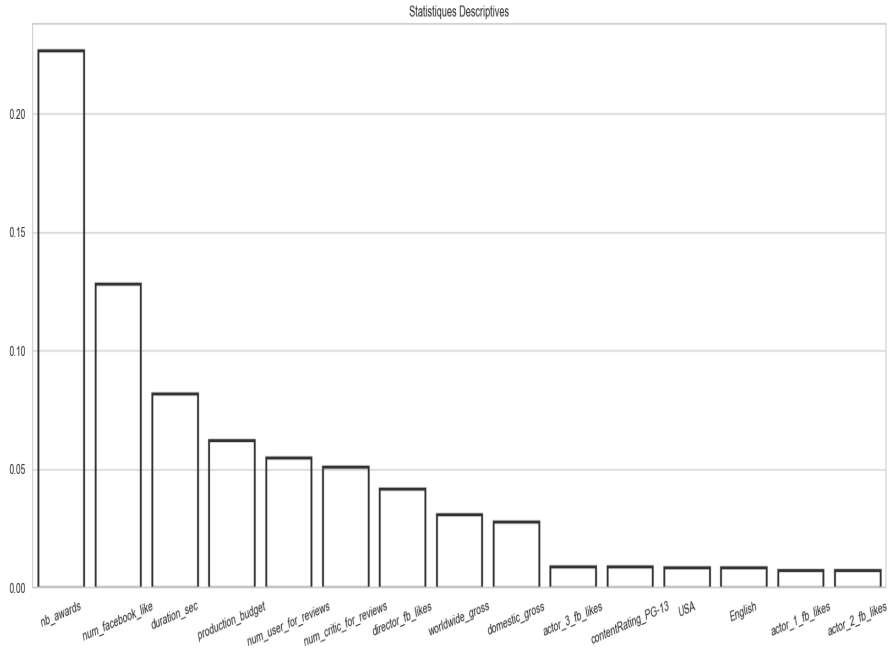
4.3.1 Méthodes de Boosting

Gradient Boosting est une méthode de boosting où l'on utilise plusieurs modèles predictifs (arbres) crées iterativement qui sont ensuite pondérés pour obtenir la prédiction finale. Nous allons tracer des learning curves pour vérifier s'il y a overfitting ou non et pour calculer le score.

Score Méthodes de Boosting		
Données	Score Ada Boosting	Score Gradient Boosting
Entrainement	0.49	0.96
Test	0.45	0.44

5 Résultats

5.1 Statistiques Descriptives



5.2 Système de Recommendation