DataScientest • com

Rapport Final Projet

Customized Product Proposal

-

Recommender System

Promotion : Juin 2020 – Mars 2020 (formation Data Scientist)

Participants : Jérémy Lecourt

Contexte

Les systèmes de recommandation sont omniprésents sur la toile (YouTube, Amazon, Netflix, Google, Facebook…).

Du point de vue économique :

Ils ont un intérêt en particulier sur les sites marchands afin de maximiser les achats des clients, en ciblant les quelques produits qui vont les intéresser le plus lors de leur navigation, et pour lesquels donc la probabilité de convertir en achat est maximale.

Auparavant, il s'agissait simplement de proposer les « Best-Sellers » du site, et c'est par l'introduction des systèmes de recommandation que l'on a pu personnaliser l'approche en proposant des produits particulièrement adaptés au profil de chaque client, et de leur parcours de navigation. L'objectif étant donc à la fois d'augmenter la satisfaction client et le chiffre d'affaire réalisé.

Du point de vue métier :

J'ai travaillé chez un précédent client en tant que responsable de la transformation du Système d'Information « Vente », et la problématique de proposition de produit personnalisé a naturellement était étudiée, lors du processus de passage de commande client. Deux démarches ont été approfondies.
Celle du « Cross-Selling » qui consiste à proposer des produits d'une catégorie distincte et liés au produit en cours d'achat par le client (par exemple, quand le client achète de la pâte à pizza, lui proposer la sauce tomate qu'il pourrait acheter).
Et celle du « Up-Selling » qui consiste à proposer cette fois des produits de la même catégorie que le produit en cours d'achat par le client, mais d'une catégorie de prix supérieure afin de tenter de maximiser la marge, tout en gardant un client satisfait par son achat.

Du point de vue technique :

Afin de permettre un parcours fluide pour le client sur l'interface Web, et d'augmenter les chances d'achat, l'expérience utilisateur (UX) doit être particulièrement adaptée (proposer les produits pertinents au bon moment de la navigation sur le site, et au bon endroit sur l'écran).
De plus, l'architecture technique (UI + Requête en base sur les résultats élaborés) doit être relativement performante et optimisée afin de ne pas intégrer de temps de latence notable lors du parcours du client sur le site.

Du point de vue scientifique :

Les données nécessaires pour effectuer cette recommandation de produits doivent être définies et stockées au format approprié, et les techniques pour déterminer les bons produits à proposer doivent être élaborées, testées et mise en production au préalable. Il peut s'agir de simples requêtes pour l'approche « Best-Seller » classique, de Data Mining avancé (ML non supervisé) pour l'approche « Content-Based » liée aux caractéristiques / similitudes produits, ou de Machine Learning avancé pour l'approche « Recommender System » avec des propositions produits personnalisées au client.

Les principales limites des « Recommender System » auxquels on choisit de s'intéresser dans le cadre de ce projet peuvent être :

- Le manque de données utilisateurs/produits ou une dispersion trop importante de ces données (« sparsity »),
- Le manque de diversité engendré dans les propositions de produits (selon l'algorithme et de la base de données utilisés),
- Le manque de fraicheur des données ou le manque de prise en compte du contexte dans lequel a été effectué la notation
  (les goûts et les couleurs changent dans le temps, et selon le contexte du moment).

Objectifs

Ce projet sur les Systèmes de Recommandation s'inscrit dans le cadre d'une proposition personnalisée de Film à établir sur la base de données récoltées sur le site Imdb.com (notations utilisateurs/films entre 1995 et 2014 et commentaires associés).

- L'objectif principal de ce projet consiste à appliquer les différentes techniques vues pendant le cursus, pour élaborer un jeu de données représentatif et performant (web scraping, data collection and cleaning, data visualisation, feature engineering), puis à comparer différents algorithmes de systèmes de recommandation, pour sélectionner les approches les plus performantes et qui seront optimisées dans un dernier temps, afin d'obtenir les meilleures prédictions possibles pour les utilisateurs.

- Enfin, il s'agit de tenter d'améliorer à nouveau notre modélisation en tentant d'y intégrer le contexte ou le comportement perçu lors de la notation, et qui peut se caractériser par exemple par le commentaire laisser par l'utilisateur au même moment qu'il note un film.

Data

Cadre

Le jeu de données utilisé in fine dans notre modélisation est le jeu de données sur les films / notations utilisateurs mis à disposition par le site Imdb.com. Ce jeu de donnée concerne donc des films et des utilisateurs internationaux, plutôt en version anglais/américain et les données sont disponibles entre 1995 et 2014.

Plus de 10 millions de notations sont présentes, mais on choisit de sélectionner les notations les plus récentes pour éviter la dérive temporelle, tout en gardant un volume de donnée suffisant pour cette problématique (environ 100K à 200K notes) et éviter l'overfitting.

Par ailleurs, une collecte des commentaires associées aux notes utilisateur est envisagée (environ 100K notes/commentaires pour la dernière partie) car ces données ne sont pas directement mises à disposition par le site Imdb.

Pertinence

Les données de notations chargées avec le Dataset Imdb (https://datasets.imdbws.com) ont été croisées avec les principales données de contenu concernant les Films (origine via https://grouplens.org/datasets/movielens/20m/) afin d'évaluer les potentielles corrélations entre les variables et la cible.

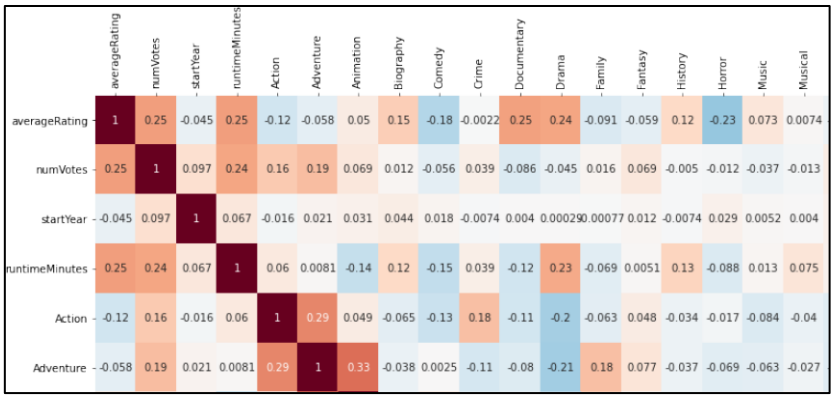
Pour faire ce « merge », la clé identifiant les films a été transformée sur l'un des Datasets pour obtenir un format commun (« imdbId »). Des nettoyages et reformatages ont été effectués (par exemple, « runtimeMinutes » est passé au format « integer » en élimant les lignes sans contenu : « /N », « date_rating » a été transformé depuis le « Timestamp » exprimé en seconde depuis 1970 vers un format date classique AAAA-MM-DD HH:MM:SS pour faciliter l'analyse temporelle, les 3 principaux genres rattachés à chaque film ont été décomposés dans des variables catégorielles pour l'analyse et la data visualisation).
In fine, un filtre temporel a été réalisé sur le contenu résultant afin de ne retenir que les données de notations les plus récentes du Dataset d'origine, tout en garantissant qu'il y ait un ordre de grandeur satisfaisant pour traiter ce type de problème (au moins 100K notes). Ainsi l'année 2014 entière a été sélectionnée (environ 290K notes).

Les variables de contenu sur les films qui ont été retenues pour l'analyse de corrélation avec la variable « rating » cible (notation client/film) sont les suivantes :

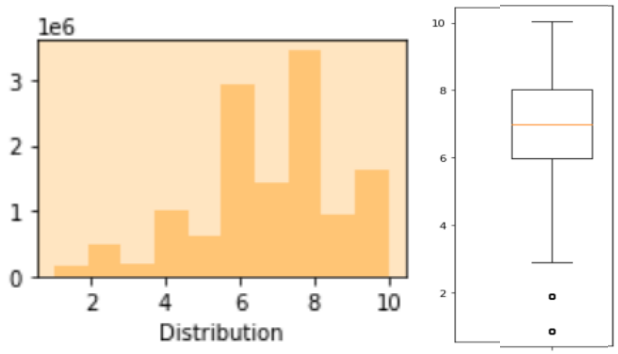
- Notation moyenne film
- Nombre de votes film
- Année de parution film
- Durée en minutes film
- Genres film (3 principaux)

Les principales corrélations identifiées sont malgré tout faibles (< 0,25, à part naturellement le lien entre « rating » client/film et « averageRating » de chaque film). Ces liens faibles sont présentés dans les 2 premiers notebooks (collecte et data visualisation), mais leurs faibles

intérêts ont poussé à s'orienter par la suite vers une approche Collaborative Filtering, plutôt que Content-Based.



La répartition de la variable cible « rating » est assez concentrée autour de la moyenne des notes (environ 7/10, et 50% des notes entre 6/10 et 8/10, avec au-delà des 2 quartiles centraux une répartition assez hétérogène entre les notes 1/10 et 10/10) :



Projet

Classification du problème

Les algorithmes de type Collaborative Filtering sont des algorithmes particuliers de Machine Learning Avancé en vue de traiter les besoins en « Recommender System » (intégration de propositions de produits personnalisés dans les applications web).

L'approche peut être assimilée dans un premier temps à une approche de type « clustering » car il faut tout d'abord étudier les similitudes entre les produits (démarche Item-Based) ou entre les utilisateurs (démarche User-Based), en définissant la mesure de similitude appropriée et le nombre de plus proches voisins à considérer.
En définitive, la prédiction qui consiste à évaluer au plus juste la note qui serait donnée pour un utilisateur/film à partir des notes données par les plus proches voisins, peut s'apparenter à un problème de régression particulier. En effet, il s'agit de prédire le plus précisément possible une note sur échelle particulière continue (entre 0 et 5, ou entre 0 et 10 par exemple).
Enfin, l'approche de type « Matrix Factorisation » consiste à réduire la dimension du jeu de donnée [Utilisateurs x Produits], en introduisant un certain nombre de facteurs latents qui vont permettre de caractériser au mieux d'une part les types de produits, et d'autre part les types d'utilisateurs, toujours pour effectuer in fine la meilleure prédiction possible sur une échelle donnée. Ces facteurs latents, contrairement à l'approche Content-Based, sont implicites et calculés par l'algorithme selon le nombre de facteurs souhaités et pour minimiser la fonction perte afin d'effectuer au mieux les prédictions souhaitées.

Même si ces approches sont assez particulières aux Systèmes de Recommandation par rapport aux autres approches plus classiques en Machine Learning, on pourrait les apparenter à des approches de type « apprentissage semi-supervisé » dans le sens où cela nécessite d'apprendre des caractéristiques ou des regroupements propres au jeu de données dans un premier temps, puis d'effectuer une prédiction de type régression sur une variable cible continue (note utilisateur / produit).

Enfin, la mesure de performance retenue pour évaluer ces différents algorithmes est la RMSE (écart quadratique moyen entre les notes réelles et les notes prédites, sur un jeu de test distinct du jeu d'entrainement). Cette mesure est très classique pour des problèmes de type régression et définie par la formule suivante :

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Choix du modèle & Optimisation

Les algorithmes suivants ont été entrainés et évalués à titre de comparaison. Ils sont issus de la librairie « surprise » que j'ai étudiée de fond en comble pour traiter les problématiques souhaitées sur les Recommender System :

- Normal Predictor :

Il s'agit d'un algorithme basic dont le concept consiste à effectuer la meilleure prédiction possible en garantissant une distribution normale centrée des notes, avec une moyenne et un écart-type estimée via la méthode MLE (Maximum Likelihood Estimation).

- KNN Basic

Il s'agit de l'approche classique Collaborative Filtering, dont les principaux paramètres sont l'approche Item-Based ou User-Based, la mesure de similarité, le nombre de plus proches voisins utilisés).

- KNN With Means

Il s'agit de la même approche Collaborative Filtering que le précédent KNN Basic, mais en prenant en compte la notation moyenne de chaque utilisateur, afin d'améliorer le résultat. Les principaux paramètres possibles pour l'optimisation sont identiques.

- SVD

Il s'agit de l'approche Matrix Factorisation classique, effectuée avec une descente de gradient stochastique (SGD), dont les principaux paramètres pour l'optimisation sont le nombre de facteurs latents, le nombre d'époques calculées, les paramètres d'initialisation des vecteurs utilisateurs et produits, les taux d'apprentissage et les termes de régularisation.

- SVDpp

Il s'agit de la même approche Matrix Factorisation, utilisant une méthode SGD avec les mêmes paramètres d'optimisation, mais qui intègre une démarche plus complexe prenant en compte en plus de la notation explicite une notation implicite (=> c'est-à-dire le fait qu'un utilisateur ait donné une note à un produit, indépendamment de cette note).

Algorithme retenu :

J'ai retenu pour l'optimisation l'algorithme SVD classique car c'est celui qui, avec les paramètres par défaut, a donné la meilleure performance sur le jeu de données sélectionné, et qui par ailleurs a un temps de traitement relativement correct (contrairement au SVDpp qui atteint une performance similaire mais avec des temps de traitement bien plus long).

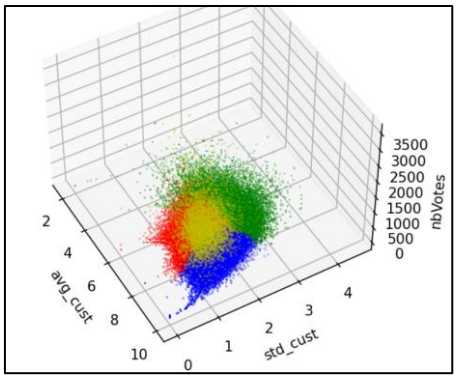
Analyse des erreurs :

- Une première analyse a consisté à étudier l'amélioration des performances selon la dispersion du jeu de données. En effet, le jeu de données brut sélectionné a une dispersion de 99% (sparsity) ce qui est très élevé et dénote des clients / produits atypiques dont les prédictions sont très difficiles à réaliser correctement (rareté de leurs notations). En utilisant des filtres raisonnables (au moins 15 votes par utilisateur, et au moins 100 votes par film), on obtient un jeu de données un peu réduit (en passant de 290K notes à 210K notes) mais dont la dispersion a été sensiblement améliorée à 92,3 %.

  ⇨ Cela a contribué par la suite à obtenir des meilleures performances quel que soit l'algorithme utilisé.

- Une seconde analyse a été faites sur la distribution des prédictions évaluées sur le jeu de test, pour remarquer que ces prédictions sont très centrées sur la moyenne de notation (3,5 / 5). Aussi par exemple, on a pu remarquer que les clients dont les notations ont tendance à s'écarter le plus de la moyenne ont des notations plus complexes à prédire (RMSE plus élevée).
  ⇨ Pour tenter d'améliorer les prédictions, j'ai repris le travail de clustering client effectué au préalable dans le 3ème notebook, et j'ai réentraîné le modèle spécifiquement sur le groupe de client « 2 – Vert », qui correspond à ces clients aux notations éparpillées. Toutefois, je n'ai pas observé d'amélioration des

prédictions mais au contraire elles se sont dégradées par rapport à la performance calculée sur ces mêmes clients suite à l'entraînement sur le jeu complet.



- ⇨ On dénote donc la spécificité de ces algorithmes de Collaborative Filtering, où la performance est tirée du principe « collaboratif » avant tout, c'est-à-dire que le cluster « 2 – Vert » tire profit de ses plus proches voisins, et notamment de tous les autres clusters 1/3/4 plus classiques, pour prédire au mieux ses notations dans notre jeu de données.
- ⇨ Intuitivement, il y a malgré tout des démarches de clustering qui peuvent être pertinentes à réaliser avant d'appliquer un Recommender System de type « collaboratif », mais il faut que ce clustering traduise réellement des comportements divergents parmi les clients, et des comportements qui améliorent ensuite les prédictions sur les notations en spécifiant le bon cluster.
- ⇨ Ce n'était manifestement pas le cas du clustering réalisé dans le Notebook 3, dont les variables utilisées pour séparer les comportements restaient très simplistes, faute de données disponibles sur les clients dans notre Dataset.

Conclusions sur les performances des différentes modélisations :

Suite aux différents travaux effectués, c'est bien l'algorithme SVD classique entraîné et évalué sur le jeu de données dont la dispersion a été améliorée (92,3%), et dont les principaux paramètres ont été optimisés ensuite via une grille de recherche, qui obtient la meilleure performance, c'est-à-dire une RMSE à 0,78 sur une échelle de notation pour rappel qui est entre 0 et 5.

Plus précisément, voici par ordre d'importance les améliorations qui ont été permises lors des différentes modélisations :

1. Via le choix de l'algorithme :

La RMSE est passée de 1,37 à 0,80, depuis le NormalPredictor jusq'au SVD classique

NormalPredictor	KNNBasic	KNNWithMeans	SVD	SVDpp
1,37	0,86	0,84	0,81	0,80

2. Via la dispersion améliorée du dataset :

La RMSE est passée de 0,85 à 0,80, avec une dispersion qui a évoluée de 99% à 92,3%

3. Via les paramètres optimaux :

La RMSE est passée de 0,80 à 0,78, au bout de 10h d'entraînement/évaluation par cross-validation sur une combinaison de 1800 paramètres différents. On en déduit que les paramètres par défaut de la librairie « surprise » sont déjà particulièrement adaptés à notre jeu de données pour cet algorithme SVD.

Description des travaux réalisés

Répartition de l'effort sur la durée et dans l'équipe

Le diagramme de Gantt reste assez synthétique pour la bonne raison que j'ai été le principal acteur durant tout le projet (pas besoin de répartition particulière des tâches, à part la première analyse de cadrage, car le groupe initial de 3 a été réduit suite à 2 abandons en début de projet). Voir Diagramme de Gantt joint en annexe.

Bibliographie

- Approche Collaborative Filtering
https://realpython.com/build-recommendation-engine-collaborative-filtering/

- Lien vers la documentation de la Librairie « surprise »
https://surprise.readthedocs.io/en/stable/prediction_algorithms_package.html

- Collaborative Filtering et Clustering client
https://hal.archives-ouvertes.fr/hal-00661187/document

- Recommender System et Sentiment Analysis
https://towardsdatascience.com/a-recommender-system-based-on-customer-preferences-and-product-reviews-3575992bb61

Bilan & Suite du projet

Ce projet pourrait contribuer à un accroissement scientifique, à travers la démarche atypique d'incorporer du Sentiment Analysis dans l'approche Recommender System. Cette démarche est intéressante car elle intègre des Réseaux de Neurones de type LSTM / GRU, ou de type Encoder/Decoder.

Par ailleurs, j'ai écarté les approches « Recommender System Hybrides » un peu plus classiques car elles nécessitent pour être performantes des données plus précises sur les clients et sur leurs comportements de navigation (non disponibles dans les Datasets fournis en ligne, pour respecter la protection des données individuelles).

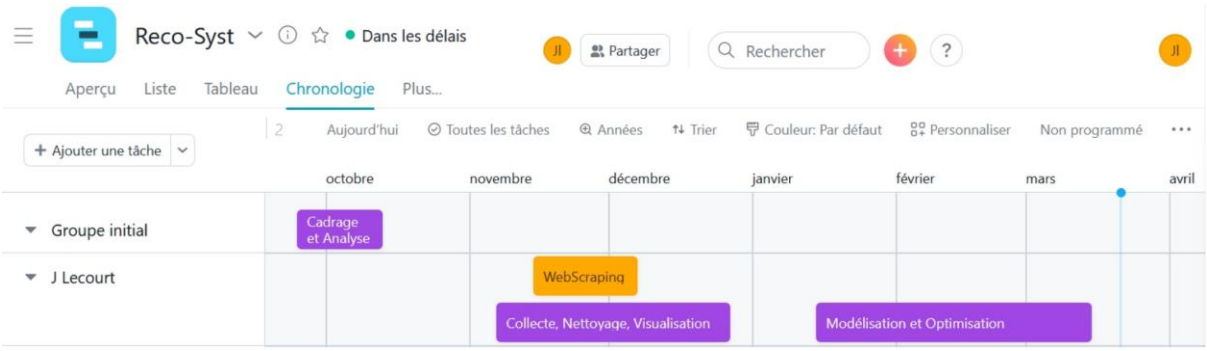
Finalement, j'ai pu entreprendre une démarche de Clustering et de Collaborative Filtering, en associant différentes approches, ce qui m'a mené à obtenir une modélisation relativement performante après optimisation.

Ce projet imaginé sur les Recommender System et que l'équipe « Datascientest » m'a laissé présenter lors de la formation des groupes, a été l'occasion de mettre en pratique les principaux modules issues du cursus Data Scientist, et d'approfondir une librairie spécifique (« Surprise ») ainsi que le Web Scraping qui était hors cursus.

C'était aussi une occasion de faire le lien avec une certaine partie de mon expérience professionnelle comme présenté dans la première partie de ce dossier (vente omnicanale / proposition de produit personnalisée), et je remercie l'équipe de m'avoir laissé entreprendre ce projet personnel.

Annexes

Diagramme de gant



Description des principaux notebooks (étapes du projet)

https://github.com/jl-datascientist/Customized.Product.Proposal.-.Recommender.System

1-LoadDataset :

Notebook qui présente la collecte, le nettoyage / formatage des données et les dépendances entre les différentes variables.

2-DataVisualisation :

Notebook qui présente les 5 visualisations approfondies du jeu de données.

3- First modelisation :

Notebook qui présente la première modélisation sous forme de clustering client, avec la visualisation 3D associée.

4- Advanced modelisation and optimisation :

Notebook qui présente la démarche d'amélioration du jeu de données, la comparaison des différents modèles « Recommender System » envisagés et l'optimisation du modèle retenu.