MS Data Science - ENSAE ParisTech
Python pour le Data Scientist
Prédiction des notes données à des produits sur Amazon à partir des reviews correspondantes

Alexandre Combessie Thibaut Duguet

December 18, 2015

1 Introduction

Pour ce projet, nous avons utilisé des données de reviews de produits du site Amazon, l'objectif étant de prédire la note donnée au produit en fonction du texte et du résumé de la review. Les données ont été récupérées sur le site `http://jmcauley.ucsd.edu/data/amazon/links.html` avec l'accord de J. J. McAuley [1, 2].

Les données étant fournies par catégories de produit, nous avons testé 3 catégories de taille[1] croissante : "Musical Instruments" (118MB), "Baby" (185MB), et "Movies and TV" (1 320MB). Nous reportons ici les résultats pour "Musical Instruments". En conclusion, nous montrerons que cela fonctionne aussi sur les autres catégories.

2 Démarche

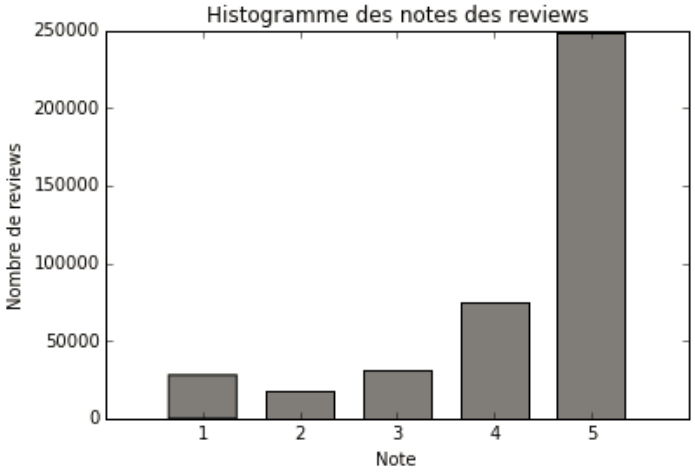
Nous avons essentiellement utilisé les packages sklearn et nltk afin d'extraire les features du texte des reviews. Nous avons ensuite implémenté deux modèles prédictifs :

- Naive Bayes avec une loi Multinomiale
- Support Vector Machine (SVM) avec Stochastic Gradient Descent

Nous avons testé d'autres modèles (Random Forests, Neural Networks, AdaBoost, k-Nearest Neighbors) mais le nombre élevé de features (>1000) nous a restreint à ces deux modèles pour des raisons de temps de calcul. Nous avons également ajouté des variables supplémentaires pour améliorer la performance de notre prédiction, à savoir le jour de la semaine, le mois de l'année et le nombre de mots dans la review et son résumé. L'utilisation de ces variables supplémentaires augmente la performance des modèles de façon relativement négligeable. En ne prenant en compte que ces variables supplémentaires, la prédiction devient triviale : seule la classe dominante est prédite.

[1] Taille compressée de la base de données des reviews en format json.gz

L'une des difficultés auxquelles nous avons été confrontés tient à distribution peu uniforme des notes :

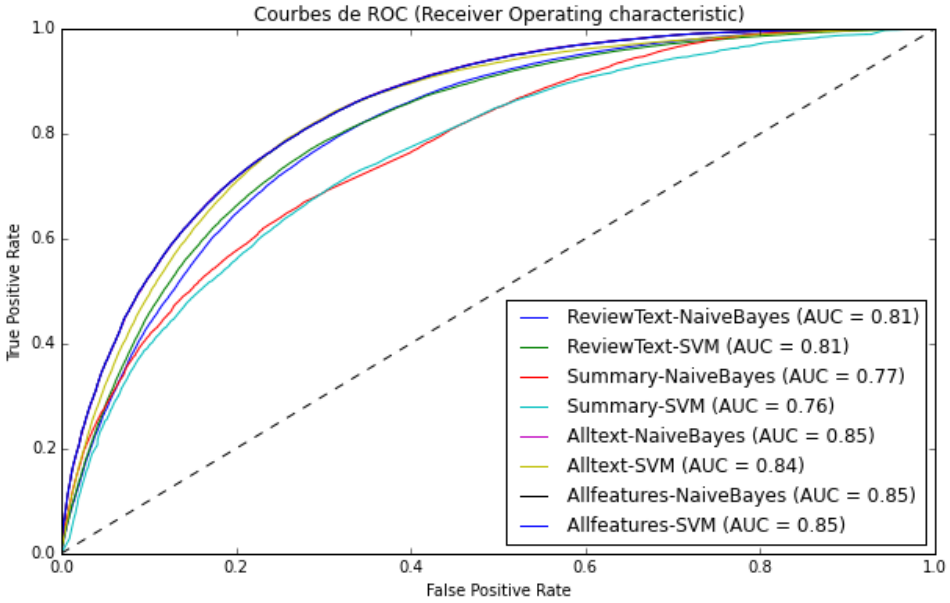


Nous avons donc décidé de modifier le problème pour ne s'intéresser qu'à une classification binaire, à savoir "1" pour les produits dont la note est 5 et "0" pour les autres.

3 Résultats clés

Le tableau suivant regroupe l'ensemble des résultats de performance des deux modèles sur le *test set*, en fonction des features utilisées en entrée dans le *training set*. On constate que les deux algorithmes sont très proches en performances, avec toutefois un léger avantage pour l'algorithme Naive Bayes, qui est par ailleurs bien plus rapide en temps de calcul (quelques secondes contre plusieurs minutes).

	Accuracy	**Precision**	**Recall**	**F1-Score**
Texte de la review - Naive Bayes	69.4%	67.5%	98.4%	80.1%
Texte de la review - SVM	69.4%	67.5%	98.4%	80.1%
Résumé de la review - Naive Bayes	72.0%	71.6%	91.6%	80.3%
Résumé de la review - SVM	69.1%	68.0%	95.6%	79.4%
Texte & résumé de la review - Naive Bayes	78.7%	78.7%	90.3%	84.1%
Texte & résumé de la review - SVM	76.2%	74.1%	95.1%	83.3%
Variables supplémentaires - Naive Bayes	62.4%	62.4%	100.0%	76.9%
Variables supplémentaires - SVM	62.4%	62.4%	100.0%	76.9%
Toutes les features - Naive Bayes	78.6%	78.7%	90.2%	84.0%
Toutes les features - SVM	78.6%	78.7%	90.2%	84.0%



La performance des modèles est bien meilleure en classification binaire, et nous avons alors cherché à optimiser les modèles en jouant sur les différents hyperparamètres mis à disposition par le package sklearn (paramètre α du modèle Naive Bayes, et paramètres α, *loss*, et *penalty* du modèle SVM). Ces optimisations nous ont permis d'atteindre une accuracy de plus de 80% avec le modèle SVM, sur toutes les features.

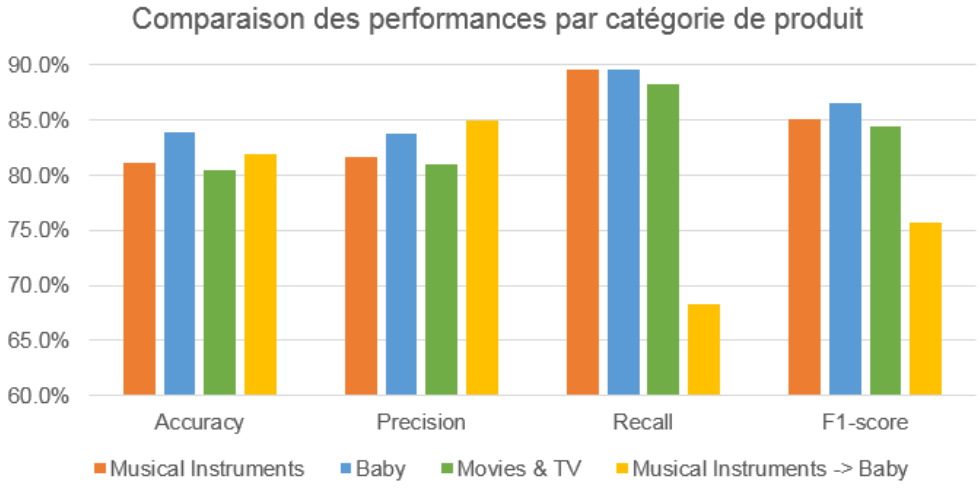
	Accuracy	**Precision**	**Recall**	**F1-score**
Naive Bayes - Sans optimisation	78.6%	78.5%	90.2%	83.5%
Naive Bayes - Avec optimisation	78.6%	78.5%	90.2%	83.5%
SVM - Sans optimisation	78.6%	78.5%	90.2%	83.5%
SVM - Avec optimisation	81.1%	81.6%	89.6%	85.1%

Enfin, nous nous sommes intéressés à la performance de l'algorithme Naive Bayes en fonction du nombre de mots dans la review, et nous avons constaté que la performance diminuait avec l'augmentation du nombre de mots.

Nombre maximum de mots dans la review	**Accuracy**
17	86.3%
30	80.5%
60	79.3%
100	78.1%
200	77.2%
300	73.5%

4 Conclusion

Pour évaluer la robustesse de notre travail, nous avons appliqué notre approche à nos trois jeux de données de catégorie de produit, en appliquant notre meilleur modèle (SVM optimisé). Cela a confirmé la robustesse de notre approche. Nous avons également testé la robustesse en appliquant le classifieur construit sur "Musical Instruments" à la catégorie "Baby". La performance baisse, notamment en recall, mais reste honorable.



Ces bonnes performances se paient en temps de calcul : la performance se dégrade rapidement avec la taille des bases de données : 15 minutes pour "Musical Instruments", 1 heure pour "Baby" et 14 heures pour "Movies and TV".

Enfin, nous nous sommes intéressé aux 20 features les plus importantes (les mots des reviews) de nos modèles pour chaque catégorie de produit. On observe quelques différences dans le vocabulaire utilisé par les utilisateurs pour chaque catégorie de produit.

Musical Instruments

superbly waltz
joseph
emotionally strauss
collaboration subdued
bluesy playful
poem atmospheric
haunting choruses
concerti 1963
excellence strengths
stravinsky expressive
arguably

Baby

soothes cheerful
entertains
maneuvers smiling
adores ivid calms
chilly vipu superb
parks mesmerized
invaluable smiles
snuggled downsides
justice playtime
memories

Movies and TV

determination explores
understated
masterfully
simplicity standout
exquisite addictive
impeccable
relaxed downside
strengths mesmerizing
dedication showcases
splendid heartbreaking
engrossing superbly

References

[1] J. J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *KDD*, 2015.

[2] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR*, 2015.