LETTRES SORBONNE UNIVERSITÉ

Loubes Alexis

Projet de méthodologie

L'objectif

Notre but ici est de tester différents modèles d'apprentissage pour de la classification sur un notebook python. Pour évaluer ces modèles on va utiliser : la précision, le rappel et enfin la moyenne harmonique de ces deux mesures c'est-à-dire la F mesure. Par soucis de clarté pour la comparaison et pour ne pas surchargé les graphique, seule la précision globale sera affichée

Nos outils méthodologiques pour la démarche seront les suivants :

- Quatre modèles testés simultanément : SVM, RandomForest, Naive Bayes et Perceptron avec une vectorisation simple ou avec TF-IDF.
- Des métriques d'évaluations pour chaque modèle.
- Utilisation progressive et cumulative de méthodes pour améliorer les résultats : si une méthode n'améliore pas significativement les résultats pour les deux modèles, on ne l'additionnera pas avec la prochaine amélioration.

➔ Améliorations utilisées : Suréchantillonnage, N-gram, enlever les mots vides, lemmatisation, modification des proportions entrainement/test du dataset.

- Augmentation du nombre d'instances du dataset (si possible).

Cette étude se veut rigoureuse dans la limite des moyens qu'elle emploiera qui seront insuffisants pour des résultats optimaux. Notre démarche ici n'est pas d'égaler ou de dépasser l'état de l'art, elle est plutôt : d'employer des outils d'apprentissage qui pour certains ont été appris en cours et de les comprendre et de réfléchir à certaines questions : Pourquoi ai-je ce résultat ? Pourquoi l'état de l'art a-t-il des meilleurs résultats ? Que pourrais-je faire pour obtenir des meilleurs résultats ?

Nos jeux de données

Les deux datasets ont été récupéré sur Kaggle et utilisé par plusieurs utilisateurs qui ont fait leur propre notebook d'apprentissage que nous verrons à la fin :

1. wiki_movie_plots_deduped.csv

- **Description :** Il s'agit d'un dataset élaboré via les pages de films sur wikipédia.
- **Mis en ligne par :** JustinR.
- **Licence : CC BY-SA 4.0.**
- **Lien : https://www.kaggle.com/datasets/jrobischon/wikipedia-movie-plots**
- **Taille : 34886 instances (films).**
- **Colonnes :** Release Year, Title,Origin/Ethnicity, Director(s), Plot, Main actor and actresses, Genre,Wiki Page URL.

On va vouloir prédire le genre du film à partir de son synopsis sur Wikipédia.

2. IEMOCAP_features.pkl

- **Description :** Il s'agit d'un dataset élaboré via des vidéos de discussions entre deux personnes où plusieurs données ont été récolté.
- **Mis en ligne par :** Ziqi Yuan **grâce à** declare-lab qui a produit ce fichier et bien sûr le site officiel IEMOCAP.
- **Licence :** inconnue.
- **Lien : https://www.kaggle.com/datasets/columbine/iemocap**
- **Taille : 7433 instances (phrases).**
- **Colonnes réalisées ici avec :** le label de l'émotion dans une colonne et la phrase dans l'autre.

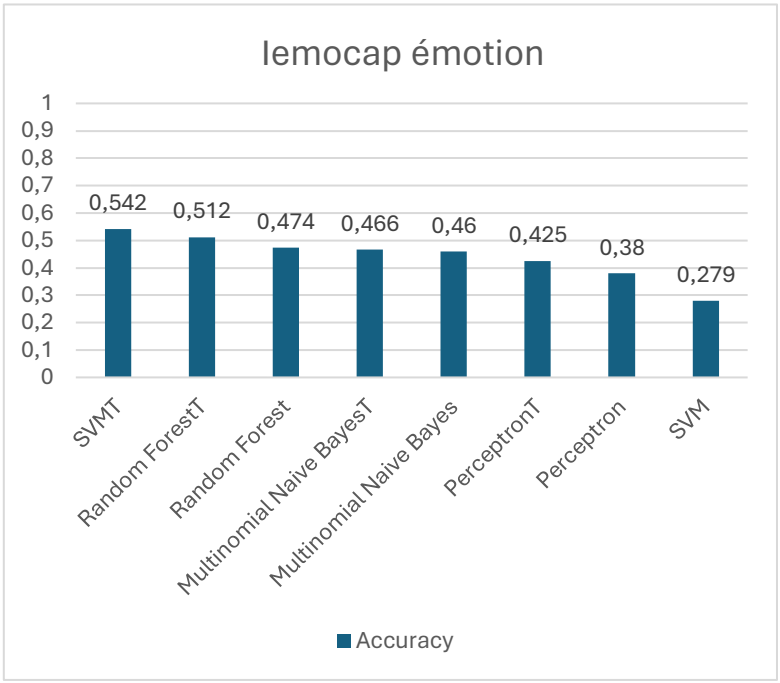
On va vouloir détecter l'émotion de la phrase selon son contenu.

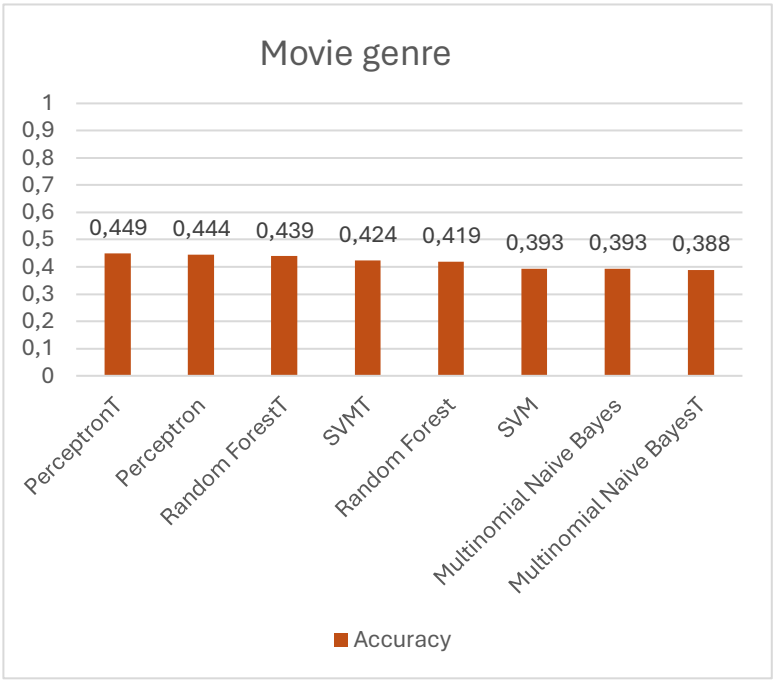
Les Résultats

1/ Premier essai avec 0,3 en proportion de test par rapport à l'apprentissage :

Voici les premiers résultats avec les deux Vectorizers et les quatre modèles sur une partie du dataset.

Lorsque qu'on a utilisé TF-IDF vectorizer on a mis un T après le modèle en légende, sinon on a utilisé Countvectorizer.





Observations :

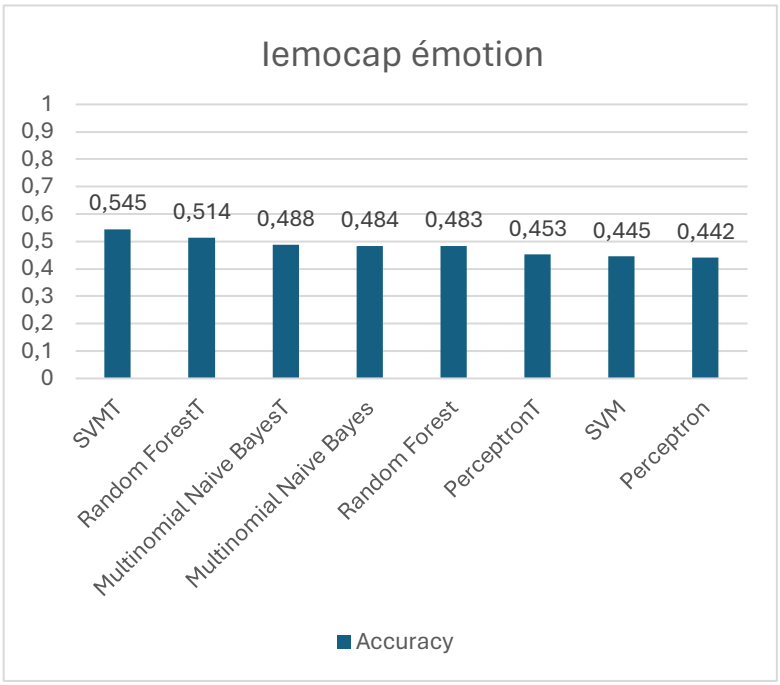
- TF-IDF améliore presque systématiquement les résultats comparés à Countvectorizer pour le même.
- Ce ne sont pas les mêmes modèles qui sont les plus performants d'un data set à l'autre
- Sans être excellent, les valeurs montrent bien que l'apprentissage augmente les chances de réussite qui sont supérieures à celle dans un cas de hasard puisqu'il y a 6 émotions à prédire et 5 genres à prédire.

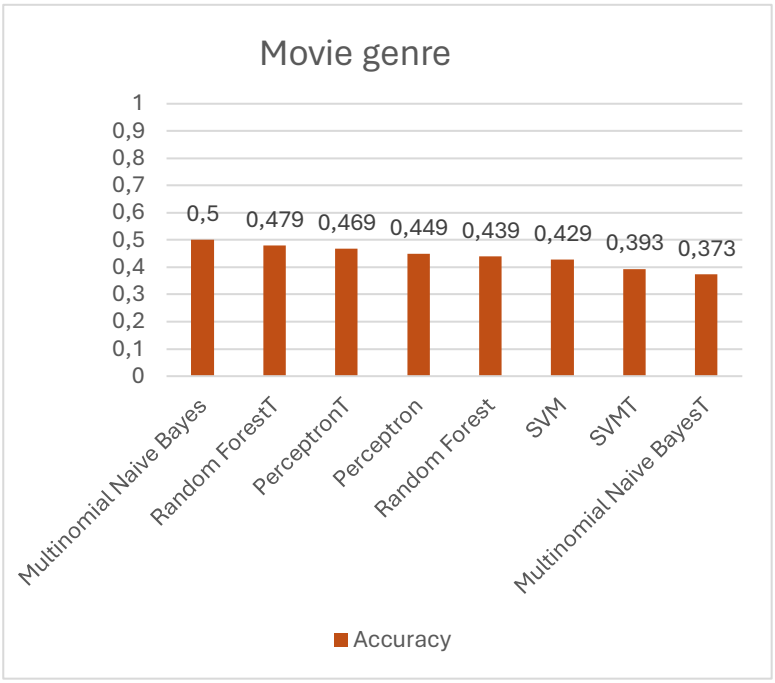
Pour un affichage moins lourd, seule la précision globale pour toute les étiquettes est affichée. Cependant, nous allons vérifier qu'on a des résultats équilibrés selon les classes avec la précision et le rappel par classe. On observe un grand problème dans les deux data sets : Ils sont déséquilibrés c'est-à-dire qu'il a des classes sous représentées.

```
Results for SVM:
Accuracy: 0.3939393939393939
Classification Report:
                precision    recall  f1-score   support

Action/Thriller       0.00      0.00      0.00        33
 Comedy/Romance       0.39      1.00      0.57        78
Drama/Melodrama       0.00      0.00      0.00        70
         Horror       0.00      0.00      0.00         7
 Sci-Fi/Fantasy       0.00      0.00      0.00        10

       accuracy                           0.39       198
      macro avg       0.08      0.20      0.11       198
   weighted avg       0.16      0.39      0.22       198
```

Figure 1 : Capture des résultats du notebook

2 / On suréchantillonne donc les étiquettes sous représentées :



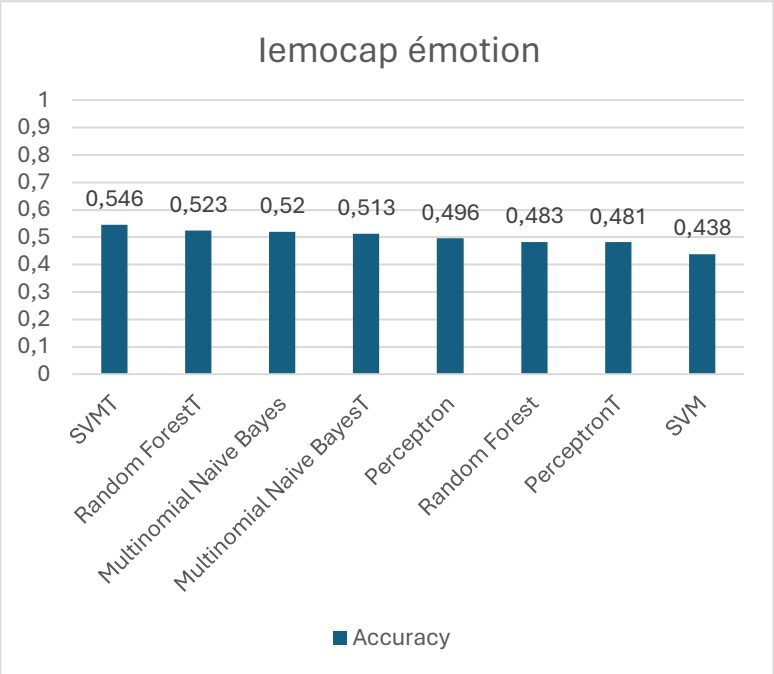


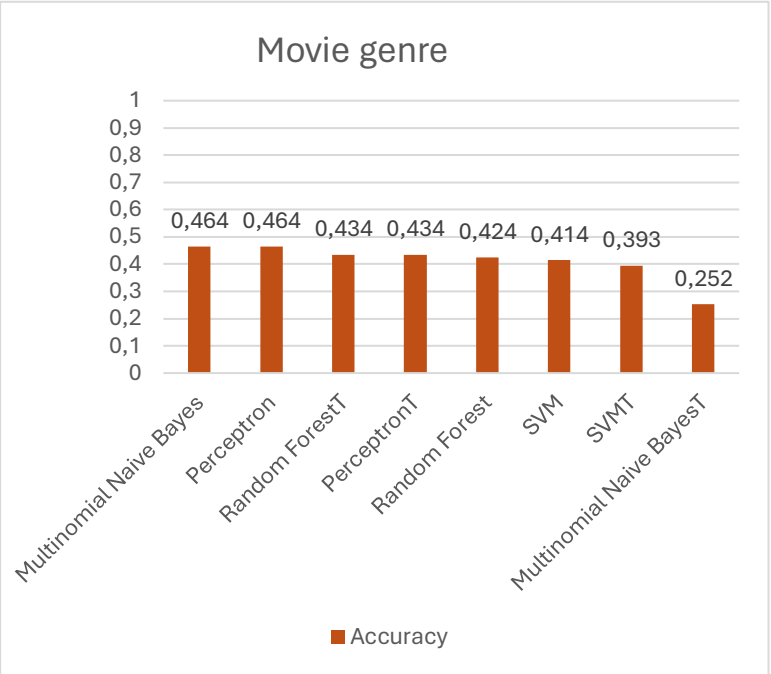
Observations :

- Amélioration globale surtout pour les modèles qui étaient moins performants.
- Pour le genre de film, Naive bayes rencontre une amélioration très élevée sans le TF-IDF qui lui, est en dernière position avec le même modèle ce qui prouve que ce vectorizer n'est pas forcément le meilleur tout le temps.
- Attention seule la partie d'apprentissage a subi un le suréchantillonnage.

Seul les tokens subissent une vectorisation, il n'y a donc pas de contexte. Essayons maintenant de faire la vectorisation en ajoutant les groupes de deux mots.

3/ Bigramme et unigramme :



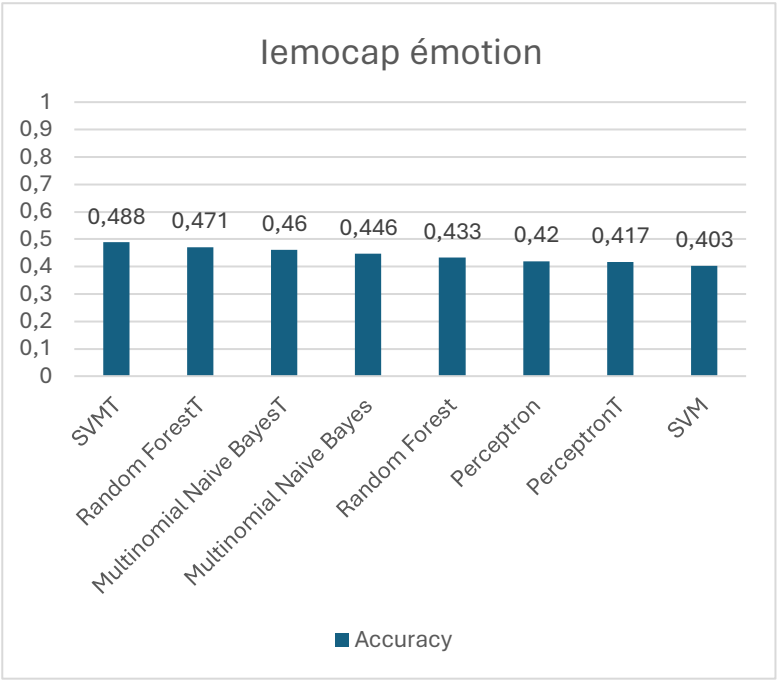


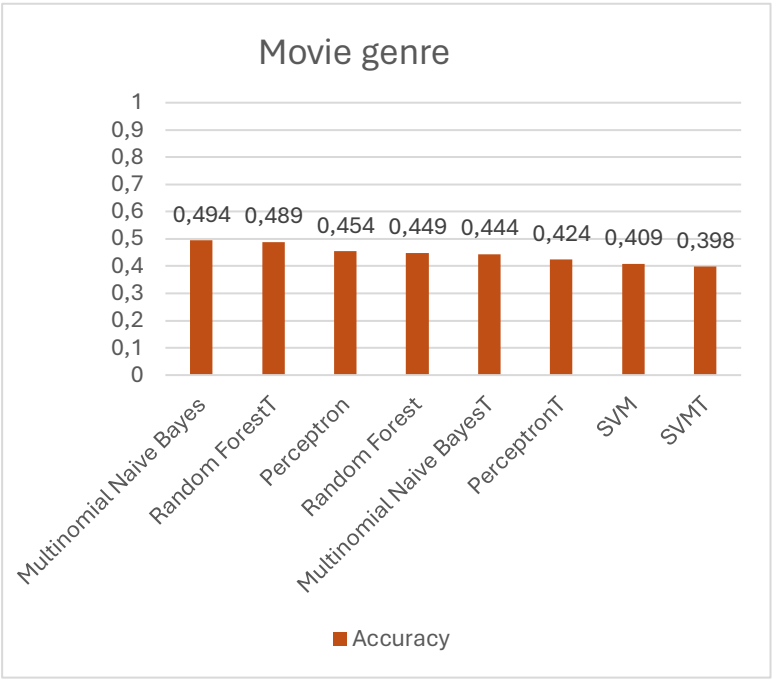
Observations :

- Amélioration globale des résultats seulement pour Iemocap émotion.
- Baisse des performances pour Movie genre.

Après avoir agi sur la vectorisation, nous allons essayer de traitement nos données textuelles en amont. Comme en retirant les mots vides.

4/ Suppressions des « stopwords » :





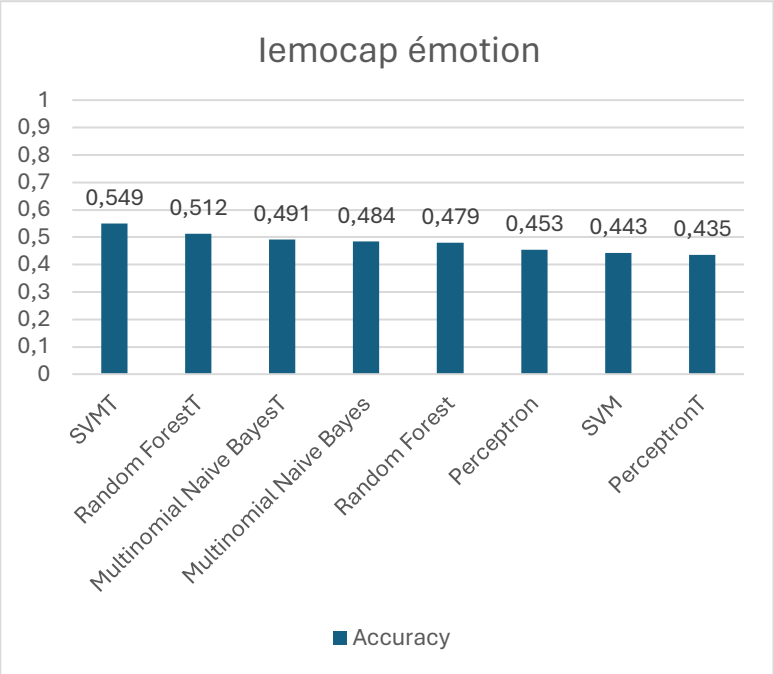
Observations :

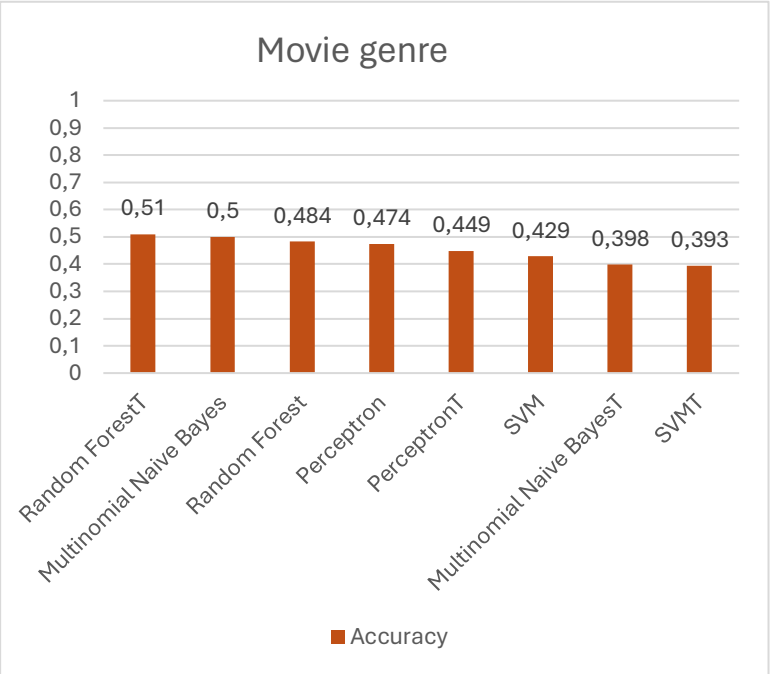
- Baisse générale des résultats par rapport au test de suréchantillonnage

➔ Stopwords non pertinents ou peut être à enrichir ?

La suppression des mots vides n'a pas l'air de fonctionner on va donc essayer de lemmatiser à la place.

5/ Lemmatisation des données (avec wordnet) :





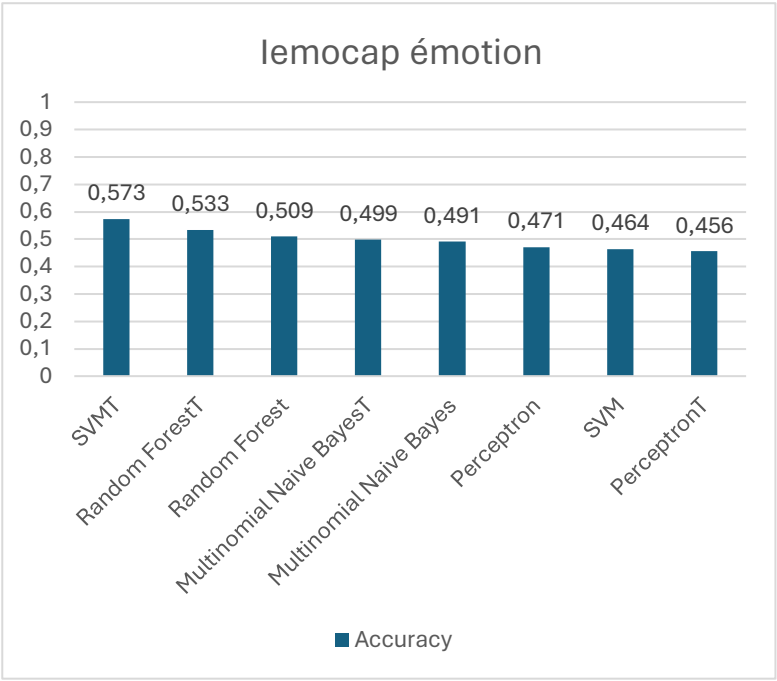
Observations :

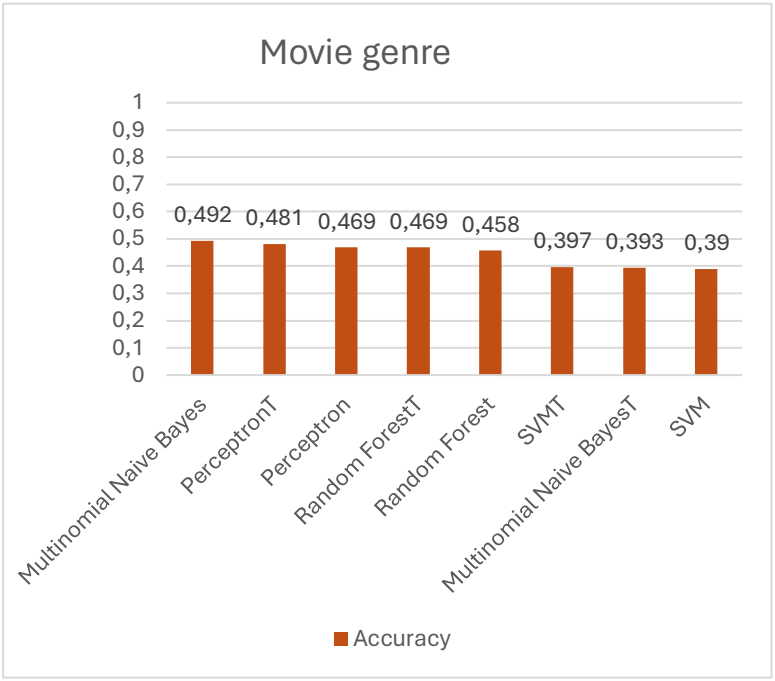
- Améliorations pour certains modèles dans iemocap émotion et Movie genre

Les améliorations influencent les résultats sur une échelle très réduite de l'ordre du centième ou du millième.

➔ On va essayer de faire varier la taille du jeu de données de test par rapport à l'apprentissage.

6/ Proportion du test 0,3 -> 0,2 :





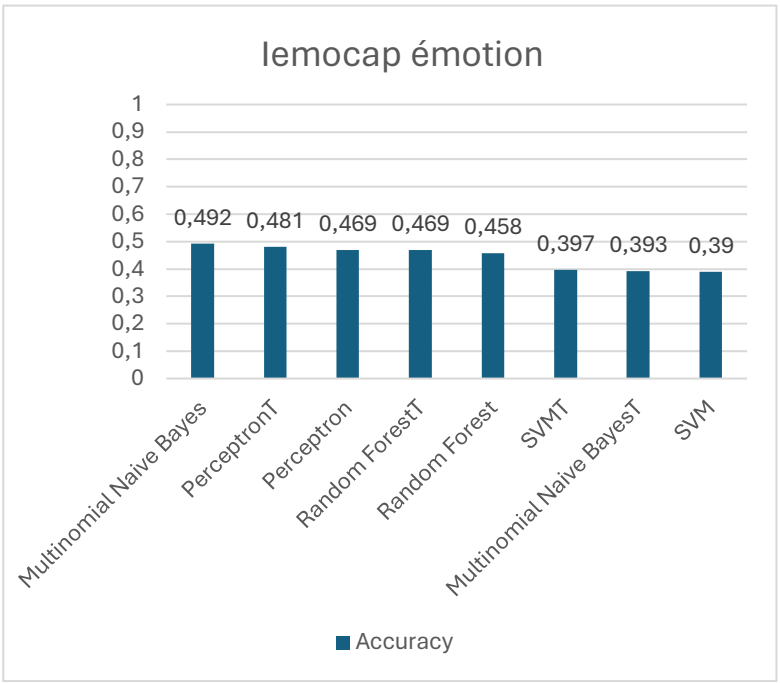
Observations :

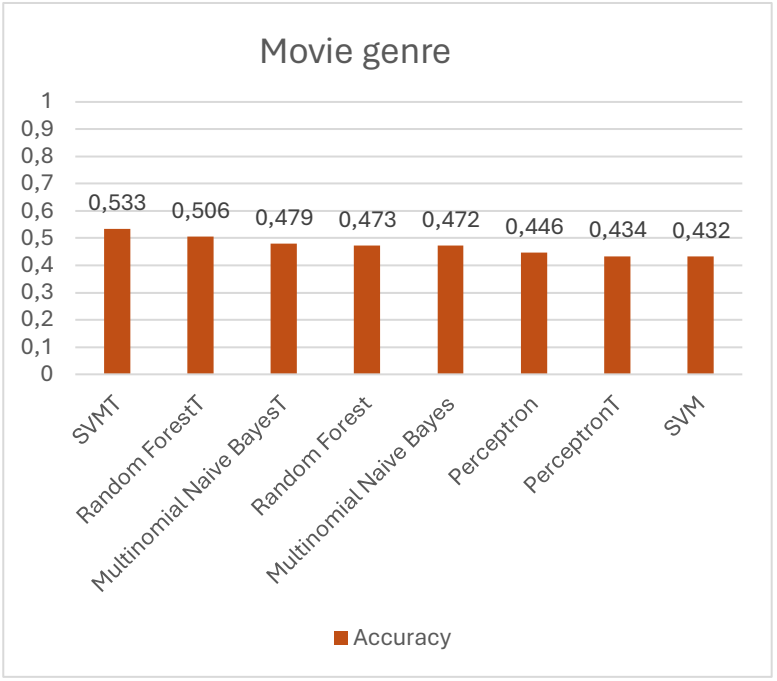
- Iemocap émotion améliore ses performances tandis que Movie genre régresse un peu.
- On a testé sur moins de données et appris sur plus de données.

➔ Est-ce que les performances se sont améliorées pour iemocap parce qu'on a plus appris ou parce qu'on a testé sur moins de données ?

On va faire l'inverse pour voir si cela change quelque chose.

7/ Proportion du test 0,3 -> 0,4 :





Observations :

- Résultats totalement inversés : Iemocap émotion régresse et Movie genre s'améliore grandement
- Cela peut s'expliquer par la différence entre ces deux datasets

➔ Iemocap émotion possède 7433 instances très pauvres en contenu, en revanche Movie genre n'a que 659 instances qui sont cependant très riches en contenu comme on peut le voir si dessous.

Une instance Iemocap émotion :

➔ `["You're", 'not', 'sorry', 'you', 'came?']`

5 token

VS

Une instance Movie genre :

➔ `['After', 'leaving', 'the', 'army', 'and', 'returning', 'to', 'university', 'newly', 'graduated', 'upper', 'class', 'Stanley'……]`

448token

Presque 100 fois plus que Iemocap emotion

8/ Augmentation de la taille du dataset :

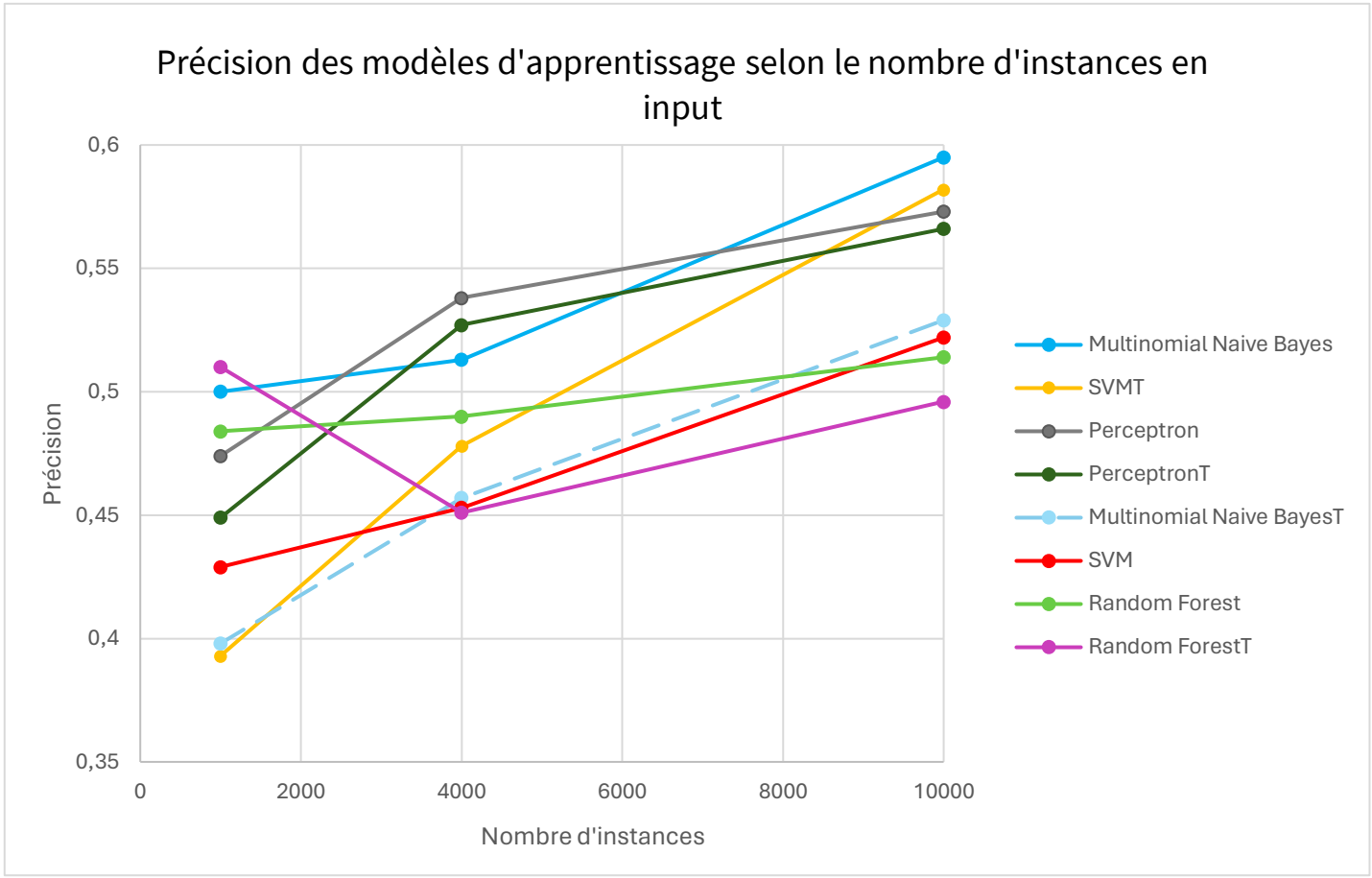
N'ayant seulement assez de données que pour Movie genre, nous allons augmenter le dataset seulement pour lui.

De plus, il doit être précisé que Movie genre subit un prétraitement préalable dans ses données. Nous nous sommes débarrassés de catégories pour éviter d'avoir des dizaines de classes à prédire comme la catégorie « Family ». Nous avons aussi fusionné des genres comme « Action » et « Thriller » ce qui pourrait d'ailleurs former un biais dans les résultats.

En effet pour 659 instances, 1000 avaient été mises en Input du prétraitement, il y a de la perte. Pour la mise en forme nous mettrons les instances mises en Input dans les résultats et pas leurs nombres réels.

Voici donc un graphique des résultats de la précision générale des modèles selon le nombre d'instances en input.

Les paramètres actifs sont : 0,2 en test, suréchantillonnage et lemmatisation ;



Observation :

- Pour tous les modèles, l'augmentation du nombre d'instances améliore la précision et ne s'est pas encore stabilisée à **10000.**
- La deuxième croissance **(4000->10000**) bien moins grande que la première pour la moitié des modèles ce qui laisse déjà supposer une future stabilisation de la précision au fur et à mesure que l'on rajoute des instances.

- On distingue deux groupes :

MNB, SVMT, Perceptron et PerceptronT (plus précis).

VS

MNBT, SVM, RandomForest et RandomforestT (moins précis).

Discussion des résultats et Conclusion

On peut déjà établir des considérations sur les modèles :

- **SVM** avec TF-IDF présente les meilleures performances la plupart du temps mais est sans doute gourmand en données, il a eu besoin de plus de données pour Movie genre qui avait des instances assez pauvres. Multinomial naive Bayes est également assez bon.

- **Random forest** avec TF-IDF est régulier en général il est toujours dans les meilleures précisions mais est le modèle qui a le moins profité de l'ajout d'instances sur le graphique plus haut. En effet il passe de premier a dernier. Peut être présente-t-il un avantage sur les datasets avec moins de données ?

Nous ne jugeons ici les modèles que pour nos datasets et non sur leurs efficacités générales.

Pour ce qui est de la Vectorisation :

L'efficacité de la vectorisation semble dépendre à la fois du dataset mais également du modèle d'apprentissage. Par conséquent ici nous ne pouvons dire si l'un est meilleur que l'autre. On a vu que des modèles avaient des meilleurs résultats pour le même modèle sans TF-IDF et inversement.

Que retenir de ces résultats ?

Les résultats obtenus ont permis de mettre en lumière certains paramètres qui influencent sur les modèles d'apprentissages : Le nombre d'instances, La longueur des instances, la pertinence de l'étiquetage des données, la taille du jeu de donnée d'apprentissage/test, les prétraitements.

Ce qu'on aurait pu faire en plus :

- Faire une **matrice de confusion** pour observer les classes qui posent un problème et s'adapter en conséquence.
- Essayer toutes les combinaisons d'amélioration qu'on a utilisé jusque-là.
- Améliorer la tokenisation
- Mieux paramétrer perceptron et d'autres modèles

Les Notebook des autres utilisateurs de kaggle

IEMOCAP :

L'utilisateur ayant publié un dataset a utilisé un réseau de neurones profond CNN avec Pytorch avec en features le son et le texte, alors Ce n'est pas très pertinent de le comparer.

Je vais plutôt comparer mes résultats avec un notebook trouvable ici : https://www.kaggle.com/code/shtrausslearning/twitter-emotion-classification fait par Andrey Shtrauss

Le but du notebook est de reconnaître 5 émotions parmi des tweet.

Voici ses résultats :

```
« 'test_accuracy': 0.933,
 'test_f1': 0.9331965147051929 »
```

Ils sont nettement supérieurs à nos résultats. La grande différence est le modèle utilisé. Voici un extrait d'un glossaire expliquant le modèle :

« BERT (Bidirectional Encoder Representations from Transformers) est un article publié par des cherc heurs de Google AI Language. Il a fait sensation dans la communauté de l'apprentissage automatique en présentant des résultats de pointe dans une grande variété de tâches de NLP, notamment la réponse aux questions (SQuAD v1.1), l'inférence en langage naturel (MNLI), etc.
La principale innovation technique de BERT consiste à appliquer l'entraînement bidirectionnel de Tra nsformer, un modèle d'attention populaire, à la modélisation du langage. »

Movie genre :

ADITIKUMARI est un des premiers utilisateurs qui m'est proposé quand je cherche des notebooks sur ce dataset https://www.kaggle.com/code/aditikumari1710/wikipedia-movie-genre-prediction . Il a utilisé Multinomial Naive Bayes et a obtenu une précision générale de 0.6 ce qui n'est pas si loin de nos meilleurs résultats. La différence avec mon notebook est un meilleur nettoyage du texte que je n'ai pas vraiment fait et la réduction des classes : Seulement 3 dans ce notebook (comedie, drame et horreur)

Bibliographie :

- *Wikipedia Movie plots*. (2018, 15 octobre). Kaggle. https://www.kaggle.com/datasets/jrobischon/wikipedia-movie-plots
- *IEMOCAP*. (2020, 17 décembre). Kaggle. https://www.kaggle.com/datasets/columbine/iemocap
- *Sklearn.linear_model.Perceptron*. (s. d.). scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html
- *ChatGPT*. (s. d.). https://chat.openai.com/
- Tremblay, C. (2023, 4 février). *Comprendre le SMOTE et éviter ses pièges*. Kobia. https://kobia.fr/imbalanced-data-smote/
- *Définition et présentation de BERT, un nouveau modèle de NLP*. (2021, 19 mai). Quantmetry. https://www.quantmetry.com/glossaire/bert-nlp/