

Table des matières

1	Introduction :	2
2	Exploration des données :	3
3	Pré-traitement des données :	3
4	Entraînement du modèle :	5
4.1	Entraînement du model SVM :	5
4.2	Optimisation des hyperparamètres :	6
5	Visualisation des résultats :	7
5.1	Matrice de confusion :	7
5.2	La courbe ROC :	7
5.3	World Cloud :	8
6	Validation des résultats :	9
6.1	Technique de validation croisée :	9
6.2	Ensemble de test :	9
7	Conclusion :	11

1 Introduction :

Le présent projet vise à développer un modèle de détection de sentiments dans les critiques de films. L'analyse des sentiments est une tâche essentielle dans le domaine du traitement automatique du langage naturel (NLP) qui permet de comprendre les opinions et les émotions exprimées dans un texte. Dans ce contexte, notre objectif est de construire un modèle capable de prédire si une critique de film est positive ou négative.

Dans ce rapport, nous présenterons les différentes étapes du projet, notamment la collecte et la préparation des données, l'exploration et l'analyse des données, la conception et l'entraînement des modèles, ainsi que l'évaluation des performances. Nous discuterons également des choix méthodologiques, des algorithmes et des techniques utilisées, en mettant l'accent sur la justification de nos décisions.

Nous utiliserons une variété de modèles d'apprentissage supervisé tels que la régression logistique, le SVM (Support Vector Machine), Random Forest et Naive Bayes pour construire notre modèle de détection de sentiments. Nous comparerons les performances de ces modèles en utilisant des mesures d'évaluation telles que l'exactitude (accuracy), la précision, le rappel et le score F1. Enfin, nous validerons nos résultats en utilisant des techniques de validation croisée ou des ensembles de données de test.

L'aboutissement de ce projet sera un modèle de détection de sentiments performant et robuste, capable de classer les critiques de films en fonction de leur polarité (positive ou négative). Les résultats obtenus pourront être utilisés pour une prise de décision éclairée dans le domaine cinématographique et pourraient être étendus à d'autres domaines où l'analyse des sentiments est pertinente.

2 Exploration des données :




On commence par l'exploration de notre base de données. Elle est composée de deux dossiers, l'un contient les données d'entraînements et l'autre les données de test. Chacun de ces deux dossiers contient un dossier pour les avis positifs et un autre pour les avis négatifs, 12 500 avis chacun. Les avis sont étiquetés, un avis négatif a un score ≤ 4 sur 10, et un avis positif a un score ≥ 7 sur 10. Chaque avis est stocké dans un fichier .txt et les noms des fichiers suivent la structure suivante `[[id]_[rating].txt]` où `[id]` est un identifiant unique et `[rating]` est le score pour cet avis sur une échelle de 1 à 10. Par exemple, le dossier `[test/pos/200_8.txt]` est le texte d'un ensemble de tests étiquetés positifs avec un identifiant unique : 200, et une note de 8/10. Voici un aperçu des fichiers :

0_2	12/04/2011 10:48	Document texte	1 Ko
1_3	12/04/2011 10:48	Document texte	1 Ko
2_3	12/04/2011 10:48	Document texte	2 Ko
3_4	12/04/2011 10:48	Document texte	1 Ko
4_4	12/04/2011 10:48	Document texte	1 Ko
5_4	12/04/2011 10:48	Document texte	3 Ko
6_3	12/04/2011 10:48	Document texte	2 Ko
7_1	12/04/2011 10:48	Document texte	2 Ko
8_2	12/04/2011 10:48	Document texte	2 Ko
9_4	12/04/2011 10:48	Document texte	3 Ko
10_3	12/04/2011 10:48	Document texte	6 Ko
11_3	12/04/2011 10:48	Document texte	2 Ko
12_4	12/04/2011 10:48	Document texte	2 Ko
13_1	12/04/2011 10:48	Document texte	1 Ko
14_1	12/04/2011 10:48	Document texte	1 Ko
15_2	12/04/2011 10:48	Document texte	3 Ko
16_1	12/04/2011 10:48	Document texte	1 Ko
17_3	12/04/2011 10:48	Document texte	1 Ko
18_1	12/04/2011 10:48	Document texte	4 Ko

I wish I knew what to make of a movie like this. It seems to be divided into two parts -- action sequences and personal dramas ashore. It follows Ashton Kutsher through survival swimmer school, guided by Master Chief Kevin Costner, then to Alaska where a couple of spectacular rescues take place, the last resulting in death. I must say that the scenes on the beach struck me as so stereotypical in so many ways that they should be barnacle encrusted. A typical bar room fight between Navy guys and Coast Guardsmen ("puddle pirates"). The experienced old timer Costner who is, as an elderly bar tender tells him, "married to the Coast Guard." The older chief who "keeps trying to prove to himself that he's still nineteen." The neglected ex wife ashore to whom Costner pays a farewell visit. The seemingly sadistic demands placed on the swimmers by the instructors, all in pursuit of a loftier goal. The gifted young man hobbled by a troubled past. The problem is that we've seen it all before. If it's Kevin Costner here, it's Clint Eastwood or John Wayne or Lou Gosset Jr. or Vigo Mortenson or Robert DeNiro elsewhere. And the climactic scene has elements drawn shamelessly from "The Perfect Storm" and "Dead Calm." None of it is fresh and none of the old stereotypical characters and situations are handled with any originality. It works best as a kind of documentary of what goes on in the swimmer's school and what could happen afterward and even that's a little weak because we don't get much in the way of instruction. It's mostly personal conflict, romance, and tension about washing out. It's a shame because the U. S. Coast Guard is rather a noble outfit, its official mission being "the safety of lives and property at sea." In war time it is transferred to the Navy Department and serves in combat roles. In World War II, the Coast Guard even managed to have a Medal of Honor winner in its ranks. But, again, we don't learn much about that. We don't really learn much about anything. The film devolves into a succession of visual displays and not too much else. A disappointment.

3 Pré-traitement des données :

On va transformer ces fichiers .txt en une base de données classique qui contiendra deux colonnes, le texte du commentaire (en anglais) et le rating de ce commentaire pour savoir si c'est un commentaire négatif ou positif.

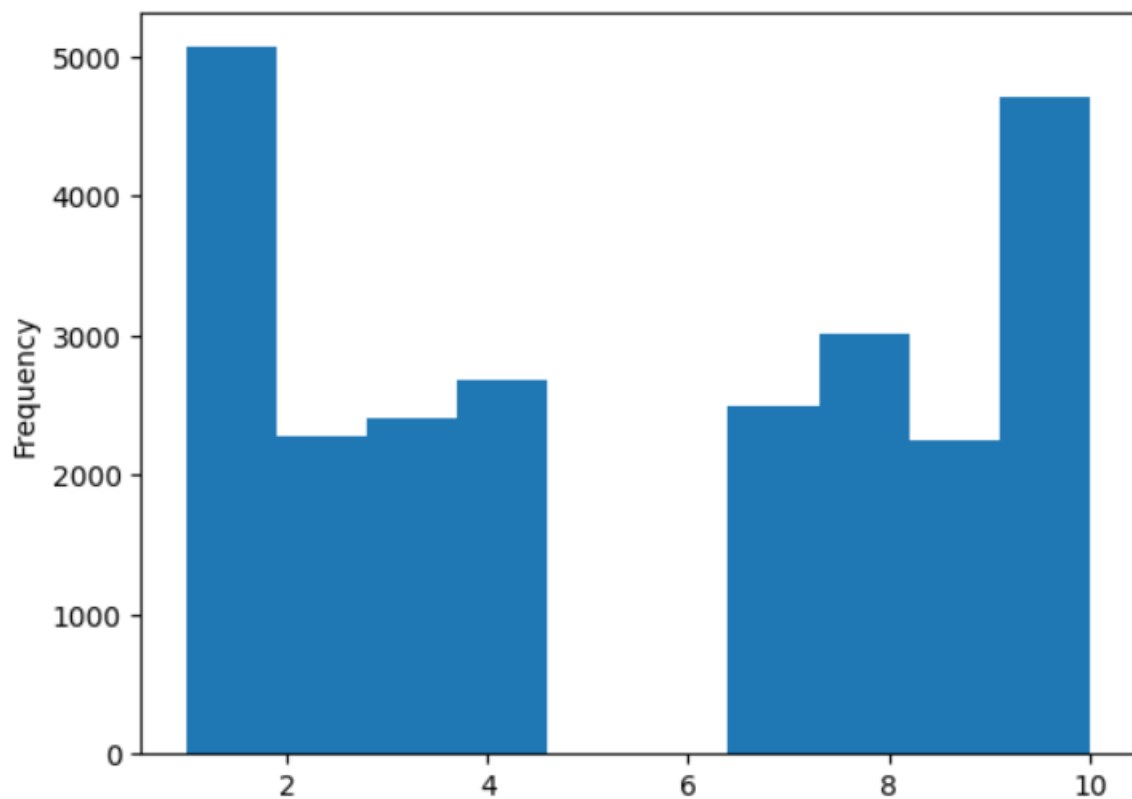
 test	14/04/2023 02:35	Data Base File	22 656 Ko
 train	14/04/2023 02:42	Data Base File	23 336 Ko
 unsup	14/04/2023 02:47	Data Base File	46 844 Ko

	rowid	id	rating	text
0	1	0	3	story man unnatural feeling pig start opening ...
1	2	0	9	bromwell high cartoon comedy ran time program ...
2	3	10000	4	airport start brand new luxury plane loaded va...
3	4	10000	8	homelessness houselessness george carlin state...
4	5	10001	10	brilliant lesley ann warren best dramatic hobo...
...
24995	24996	999	3	saw last night stockholm film festival one hug...
24996	24997	99	1	film pick pound turn rather good century film ...
24997	24998	99	8	christmas together actually came time raised j...
24998	24999	9	1	one dumbest film ever seen rip nearly ever typ...
24999	25000	9	7	romantic drama director martin ritt unbelievab...

On va retirer les valeurs dupliquées et on a aucune valeur manquante.

```
#Valeurs dupliquées
df_train.duplicated().sum()
```

98



Ce graphe nous montre la répartition des scores dans la base de données d'entraînement.

4 Entraînement du modèle :

Avant d'entraîner notre modèle on va d'abord vectoriser les textes en vecteurs numériques avec la vectorisation TF-IDF pour mieux entraîner notre modèle. Puis on va diviser nos données en deux parties une pour l'entraînement 80% et l'autre pour le test 20%.

```
# Charger les données d'entraînement depuis la base de données
x_train = df_train['text']
y_train = df_train['rating']

# x_train est une liste de textes et y_train est une liste d'étiquettes du score
# Convertir les textes en vecteurs numériques avec la vectorisation TF-IDF
vectorizer = TfidfVectorizer()
x_train_vectorized = vectorizer.fit_transform(x_train)

# On divise les données en ensemble d'entraînement et ensemble de test
x_train_vec, x_test_vec, y_train, y_test = train_test_split(x_train_vectorized, y_train, test_size=0.2, random_state=42)
```

On va également encoder la variable rating pour la transformer en une variable catégorielle binaire :

```
# Encodage de la colonne rating
df['rating'] = df['rating'].apply(lambda x: 0 if x < 5 else 1)
```

4.1 Entraînement du modèle SVM :

Après avoir testé plusieurs modèles. On va entraîner un modèle SVM et faire un test de prédiction sur le dataframe de test. Pour finir on va calculer l'accuracy, le recall, la précision et le F1-score pour pouvoir comparer aux autres modèles.

```
# Entraînement du modèle
svm = SVC(kernel='rbf')
svm.fit(x_train_vec, y_train)

# Faire des prédictions sur l'ensemble de test
y_pred_svm = svm.predict(x_test_vec)

# Calculer l'exactitude des prédictions
accuracy_svm = accuracy_score(y_test, y_pred_svm)
precision_svm = precision_score(y_test, y_pred_svm, average='macro')
recall_svm = recall_score(y_test, y_pred_svm, average='macro')
f1_score_svm = f1_score(y_test, y_pred_svm, average='macro')

print("Exactitude de SVM: ", accuracy_svm)
print("Précision de SVM: ", precision_svm)
print("Rappel de SVM: ", recall_svm)
print("F1 score de SVM: ", f1_score_svm)

Exactitude de SVM: 0.43330650839955737
Précision de SVM: 0.3514561189677158
Rappel de SVM: 0.3302485947978263
F1 score de SVM: 0.2925203011885414
```

Voici les résultats de la prédiction du model sur les données de test :

	SVM
Accuracy	0,8859
Precision	0,8863
Recall	0,8857
F1_score	0,8858

4.2 Optimisation des hyperparamètres :

On va se baser sur les données d'entraînement pour optimiser les hyperparamètres de notre model SVM.

```
from sklearn.model_selection import GridSearchCV

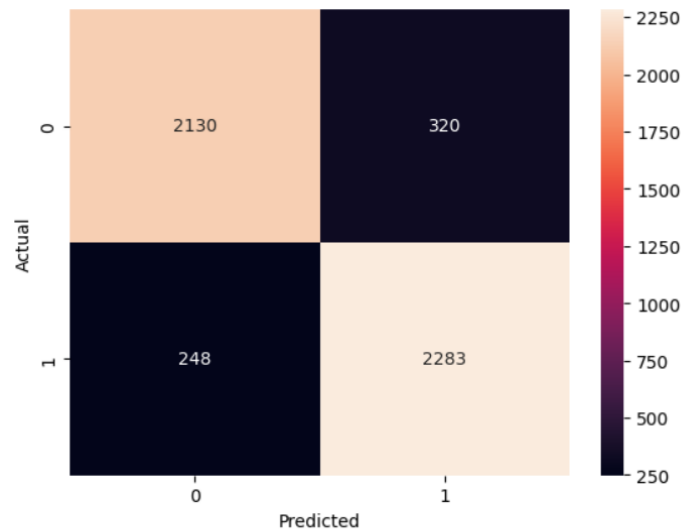
param_grid = {'C': [0.1, 1, 10], 'gamma': [0.1, 1, 10], 'kernel': ['linear']}
svm_model = SVC()
grid_search = GridSearchCV(svm_model, param_grid)
grid_search.fit(x_train_vec, y_train)
print(grid_search.best_params_)
```

Voici les résultats du grid_search :

```
{'C': 1, 'gamma': 0.1, 'kernel': 'linear'}
```

5 Visualisation des résultats :

5.1 Matrice de confusion :



La matrice de confusion nous montre que les FN sont supérieurs au FP et également que les VP sont supérieurs au VN donc le modèle a un peu plus de mal pour détecter les avis négatifs que les avis positifs.

5.2 La courbe ROC :

Le modèle qu'on a adopté est un bon modèle de classification car la courbe est très proche du coin supérieur gauche. De plus le score AUC (0.89) est très proche de 1 donc notre modèle est capable de bien distinguer entre les classes positives et négatives.

ne sont pas très utiles pour notre analyse. Mais il y a pas mal de mots qui représentent les sentiments des personnes qui ont commentées comme : "Funny", "Good"...

6 Validation des résultats :

6.1 Technique de validation croisée :

On va d'abord utiliser la technique de validation croisée. On remarque que notre modèle a une très bonne précision de 95% et que l'écart type est très faible ce qui indique que notre modèle n'est pas très sensible à la distribution des données.

```
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
from sklearn.datasets import make_classification

# Création de données fictives pour l'exemple
X, y = make_classification(n_samples=1000, random_state=0)

# Création du modèle SVM
clf = SVC(kernel='linear', C=1, random_state=0)

# Validation croisée avec 5 partitions
scores = cross_val_score(clf, X, y, cv=5)

# Affichage des scores moyens et de l'écart-type
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

Accuracy: 0.95 (+/- 0.02)

6.2 Ensemble de test :

Pour s'assurer d'avantage que le modèle est adéquat on va utiliser une base de données de test différente de notre base de données originale. On va bien sûr traiter la data de la base de test de la même manière avec laquelle nous avons traité la base de données d'entraînement. (Encodage de la variable rating, vectorisation des textes et suppression des valeurs dupliquées)

```

# Encodage de la colonne rating
df_test['rating'] = df_test['rating'].apply(lambda x: 0 if x < 5 else 1)

# Charger les données de test depuis la base de données
x_testb = df_test['text']
y_testb = df_test['rating']

# x_test est une liste de textes et y_test est une liste d'étiquettes du score
# Convertir les textes en vecteurs numériques avec la vectorisation TF-IDF

x_test_vectorized = vectorizer.transform(df_test['text'])

```

Puis on va utiliser notre modèle pour prédire la classe des critiques de la base de données de test :

```

y_predtest_svm_b = svm_model_best.predict(x_test_vectorized)

# Calculer l'exactitude des prédictions
accuracy_svm_b = accuracy_score(y_testb, y_predtest_svm_b)
precision_svm_b = precision_score(y_testb, y_predtest_svm_b, average='macro')
recall_svm_b = recall_score(y_testb, y_predtest_svm_b, average='macro')
f1_score_svm_b = f1_score(y_testb, y_predtest_svm_b, average='macro')

print("Exactitude de SVM: ", accuracy_svm_b)
print("Précision de SVM: ", precision_svm_b)
print("Rappel de SVM: ", recall_svm_b)
print("F1 score de SVM: ", f1_score_svm_b)

```

Les résultats sont les suivants :

```

Exactitude de SVM:  0.8706555922909442
Précision de SVM:  0.8707259659209727
Rappel de SVM:  0.8706784364567883
F1 score de SVM:  0.8706532296707405

```

Ils sont presque similaires aux résultats obtenus avec le test sur la base d'entraînement.

7 Conclusion :

Dans le cadre de ce projet, nous avons développé un modèle de détection de sentiments dans les critiques de films en utilisant différentes techniques d'apprentissage supervisé. Notre objectif était de construire un modèle capable de prédire si une critique de film était positive ou négative.

Nous avons commencé par collecter et préparer les données en effectuant des opérations telles que le nettoyage des textes, la vectorisation des données textuelles et la division de l'ensemble de données en ensembles d'entraînement et de test. Ensuite, nous avons exploré les données pour mieux comprendre leurs caractéristiques et leurs distributions.

L'évaluation de nos modèles a été réalisée en utilisant différentes mesures telles que l'exactitude (accuracy), la précision, le rappel et le score F1. Nous avons également utilisé des techniques de validation croisée ou des ensembles de données de test pour valider nos résultats.

Les résultats obtenus ont démontré que notre modèle était capable de prédire avec une certaine précision les sentiments exprimés dans les critiques de films. Cependant, il est important de noter qu'il existe des limitations et des défis dans la détection des sentiments, notamment la subjectivité des opinions et les subtilités linguistiques qui peuvent influencer les résultats.

En conclusion, ce projet nous a permis de développer un modèle de détection de sentiments dans les critiques de films en utilisant différentes techniques d'apprentissage supervisé. Les résultats obtenus fournissent des informations précieuses sur les sentiments exprimés dans les critiques et peuvent être utilisés pour prendre des décisions éclairées dans l'industrie cinématographique. Cependant, il convient de continuer à explorer et à améliorer les modèles en tenant compte des spécificités du domaine et des défis inhérents à l'analyse des sentiments.

Ce projet ouvre la voie à des développements futurs, tels que l'exploration de techniques plus avancées d'apprentissage automatique, l'utilisation de modèles de langage pré-entraînés tels que BERT, l'extension à d'autres langues et domaines, ainsi que l'intégration de techniques de traitement du langage naturel pour une analyse plus fine des sentiments exprimés dans les critiques de films.

En somme, ce projet a été une expérience enrichissante qui nous a permis d'explorer l'application des techniques d'apprentissage supervisé à l'analyse des sentiments dans les critiques de films, et d'appréhender les défis et les opportunités de ce domaine passionnant du traitement automatique du langage naturel.