Etape 1 : Récupération des données

* Webscrapping du site trustpilot
  + Jeu pour création du modèle : Scrapping de 120 000 commentaires cdiscount avec leur note
  + Jeu de test : Scrapping de 100 commentaires amazon avec leur note
* Mise en forme des données et enregistrement du jeu pour création du modèle en local dans un fichier cdiscount.csv
* Mise en forme des données et enregistrement du jeu de test en local dans un fichier amazon\_test1.csv

Etape 2 : Tokenization et nettoyage des données

* Utilisation des bibliothèque NLTK et REGEX pour tokenizer et nettoyer
* Chargement des stop Word
* Création d’une fonction qui :
  + Tokenize les commentaires en mots
  + Ne conserve que les mots contenant des lettres
  + Supprime les stopwords dans 3 langues (anglais, français, espagnol)
* Application de la fonction et enregistrement dans un fichier cdiscount\_retraite.csv

Etape 3 : Wordcloud

* Affichage des 100 mots les plus représentés dans les commentaires sur un fond au format d’une étoile
* Affichage des 100 mots les plus représentés dans les titres sur un fond au format d’une étoile
* Séparation des commentaires en 2 catégories :
  + Commentaires positifs avec notes de 4 et 5
    - Affichage des 100 mots les plus représentés dans les commentaires positifs sur un fond au format d’une étoile
  + Commentaires négatifs avec notes de 1 et 2
    - Affichage des 100 mots les plus représentés dans les commentaires négatifs sur un fond au format d’une étoile
* Séparation des titres en 2 catégories :
  + Titres positifs avec notes de 4 et 5
    - Affichage des 100 mots les plus représentés dans les titres positifs sur un fond au format d’une étoile
  + Titres négatifs avec notes de 1 et 2
    - Affichage des 100 mots les plus représentés dans les titres positifs sur un fond au format d’une étoile

Etape 4 : Création des modèles de machine learning

* Vectorisation des mots avec CountVectorizer de Sklearn avec les paramètres suivants pour l’ensembles des modèles Sklearn testés :
  + CountVectorizer(min\_df=3, max\_features=12000)
* Création d’un modèle de machine learning BalanceRandonForestClassifier testé avec des paramètres différents
  + n\_estimators=10, max\_depth=10 accuracy = 0.39
  + n\_estimators=10, max\_depth=100 accuracy = 0.50
* Création d’un modèle de machine learning RandonForestClassifier testé avec des paramètres différents
  + n\_estimators=10, max\_depth=10 accuracy = 0.65
  + n\_estimators=10, max\_depth=100 accuracy = 0.69
  + Paramètres par défaut accuracy = 0.69
* Création d’un modèle de machine learning DecisionTreeClassifier testé avec des paramètres différents
  + max\_depth=1 accuracy = 0.64
  + max\_depth=10 accuracy = 0.67
* Création d’un modèle de machine learning GradentBoostingClassifier testé avec des paramètres différents
  + n\_estimators=2, learning\_rate=1, random\_state=0 accuracy = 0.65
  + n\_estimators=5, learning\_rate=0.4 accuracy = 0.67
* Création d’un modèle de machine learning SVC testé avec des paramètres différents
  + C=10 accuracy = 1
  + C=100 accuracy = 1
  + C=200 accuracy = 1
* Vectorisation des mots avec TfidfVectorizer de Sklearn avec les paramètres suivants pour l’ensembles des modèles Sklearn testés :
  + TfidfVectorizer (min\_df=3)
* Création d’un modèle de machine learning BalanceRandonForestClassifier testé avec des paramètres différents
  + n\_estimators=10, max\_depth=10 accuracy = 0.43
* Création d’un modèle de machine learning RandonForestClassifier testé avec des paramètres différents
  + n\_estimators=10, max\_depth=10 accuracy = 0.65
  + (n\_estimators=50 accuracy = 0.698