Analyse des avis clients en assurance

Groupe : DIA2
Noms : RUNAVOT Paul, LEVALLOIS Patrick

Barème

- **Nettoyage des données : 2 points (points négatifs si pas bien réalisés)**
  - Mise en évidence des mots (et n-grammes) fréquents
- **Résumé, traduction et génération : 2 points**
  - Sortir un fichier propre avec plusieurs colonnes nettoyées et textes corrigés et traduits
- **Correction d'orthographe : 2 points**
- **Détection de sentiments (multiclass, ou classification binaire) : 2 points (points négatifs possibles)**
- **Topic modelling et listes des topics (travail collectif) : 2 points (points négatifs possibles)**
- **Embedding pour identifier des mots similaires et enrichir liste des thèmes : 2 points (points négatifs possibles)**
  - Entraînement word2vec : 2 points, GloVe : 2 points
  - Visualisation des embedding avec matplotlib et Tensorboard : 2 points
  - Implémentation de la distance euclidienne ou cosine : 1 point
- **Apprentissage supervisé, chaque modèle bien fait et bien présenté : 2 points (points négatifs possibles)**
  - TF-IDF et ML classiques
  - Modèle basique avec une couche d'embedding (visualisation des embeddings avec Tensoboard : 1 point supplémentaire)
  - Modèle avec des embeddings pré-entraînés (visualisation des embeddings avec Tensoboard : 1 point supplémentaire)
  - **USE** (Universal Sentence Embedding) ou équivalents, RNN **LSTM**, **CNN**, **BERT** ou autres modèles sur Hugging Face
- **Interprétation des résultats (points négatifs possibles)**
  - Analyse des erreurs : 1 point
  - Détection de sentiments : 2 points
  - Modèles classiques avec les thèmes : 2 points
  - Modèles de deep learnig pour les mots : 2 points
- **Clarté de la présentation : 2 points (points négatifs possibles)**
- **Post sur Linkedin et obtention de 2000 likes/commentaire (présentation des résultats intéressants) → 20 (à normaliser éventuellement)**

Exemple de sommaire

Objectif et contexte

- **Contexte :**
  - Les clients de différentes assurances et différents produits peuvent laisser des avis ainsi que des notes de 1 à 5 sur différents sites tel que opinion assurance ou trustpilot.

- **Objectifs :**
  - Identifier les thématiques dans les commentaires
  - Comprendre l'impact des avis sur les notes données

- **Techniques utilisées :**

  - Réaliser une analyse exploratoire
  - Embeddings
  - identifier les thématiques
  - Entrainement de modèles Tensorflow
  - Analyse des erreurs

1 | Présentation de la problématique et des données du projet

- Structure générale des fichiers
- Présentation des variables
- Analyse/présentation des variables

Présentation de la problématique

- **La base de données est un datasets d'avis clients sur leurs assurances.**

- **Chaque ligne contient le nom de l'assurance, le produit , la date, un avis et une note associé.**

- **On cherche à analyser l'impact des avis sur les notes des usagers et performer des analyse de sentiments sur les avis**

1.2 Présentation des données

Objectifs

- **Comprendre les produits et les assureurs ainsi que leurs notes moyennes**
- **Visualisation l’évolution temporelle des notes**

Présentation des données

- **Variables**
  - **Auteur**
  - **Date de publication**
  - **Type de produit**
  - **Assureur**
  - **Avis client**
  - **Traduction de l’avis en anglais**
  - **Note associée**

Exemples d’observations

- **Fautes d'orthographe ou problèmes de format, ex (Bonjour,\n\nAprès des années)**
- **Absences de données dans certaines colonnes**

```
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   note             24104 non-null  float64
 1   auteur           32999 non-null  object 
 2   avis             33000 non-null  object 
 3   assureur         33000 non-null  object 
 4   produit          33000 non-null  object 
 5   type             33000 non-null  object 
 6   date_publication  33000 non-null  object 
 7   date_exp         33000 non-null  object 
 8   avis_en          32998 non-null  object 
 9   avis_cor         0 non-null      float64
 10  avis_cor_en      0 non-null      float64
dtypes: float64(3), object(8)
```

2 | Exploration/Transformation/ feature engineering

- Nettoyage des données
- Exploration simple et nettoyage
- Word embedding (self-supervisé)

2.1 Nettoyage des données

Objectifs

- **Rendre le dataset clair et utilisable**
- **Corriger les fautes d'orthographe**
- **Supprimer les lignes incomplètes**

Nettoyages simples

- **Corrections des fautes d'ortographes en français et en anglais**
- **Suppression de la ponctuation et des stopwords**
- **Tokenisation**
- **Lemmatisation**
- **Analyse des n-grammes**

Nettoyages réalisés

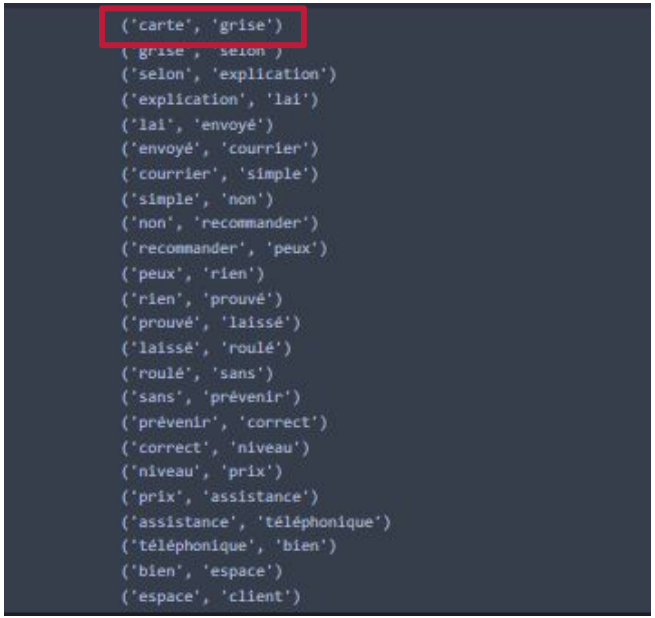
Ajout manuel de nouveaux stopwords dans la liste des stopwords a retirer

Avant

Après

Nettoyages réalisés

Analyse des bigrames pour trouver des mots composés



Remplacement des deux mots “carte” et “grise” cote à cote par “carte_grise”

Synthèse des nettoyages réalisés

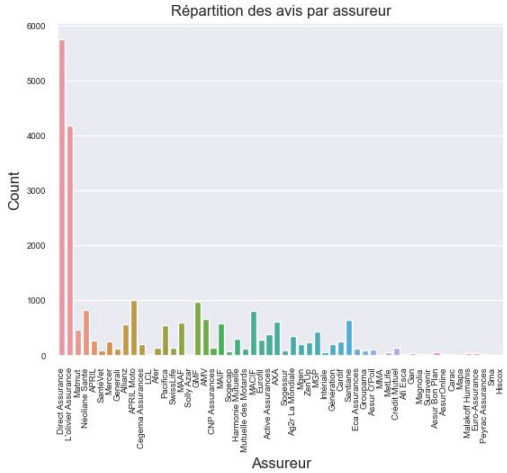
- **Après nettoyage on passe d'environ 33000 ligne a environ 25000 et on ajoute 3 colonnes à notre fichiers.**

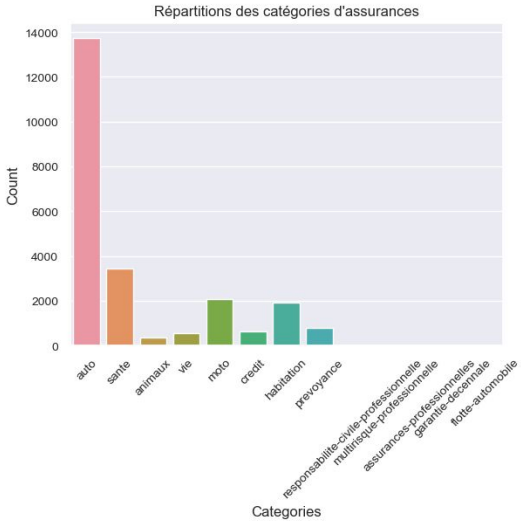
2.2 Exploration des données

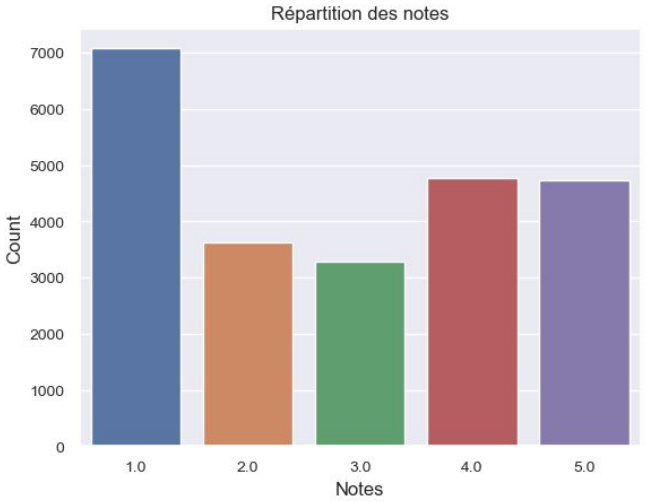
Objectifs

- **Comprendre les produits et les assureurs ainsi que leurs notes moyennes**
- **Visualisation l’évolution temporelle des notes**

Exploration globale des données







Répartitions très inégales pour l'assureur et les catégories d'assurance

Exploration globale des données

- **Nombre d'avis et moyenne de note par produit, par assureur, par date, etc.**
- **Nombre de mots dans un avis, en fonction de la note**
- **Créer des modèles par produit éventuellement suite à l'exploration**

2.3 Word embedding

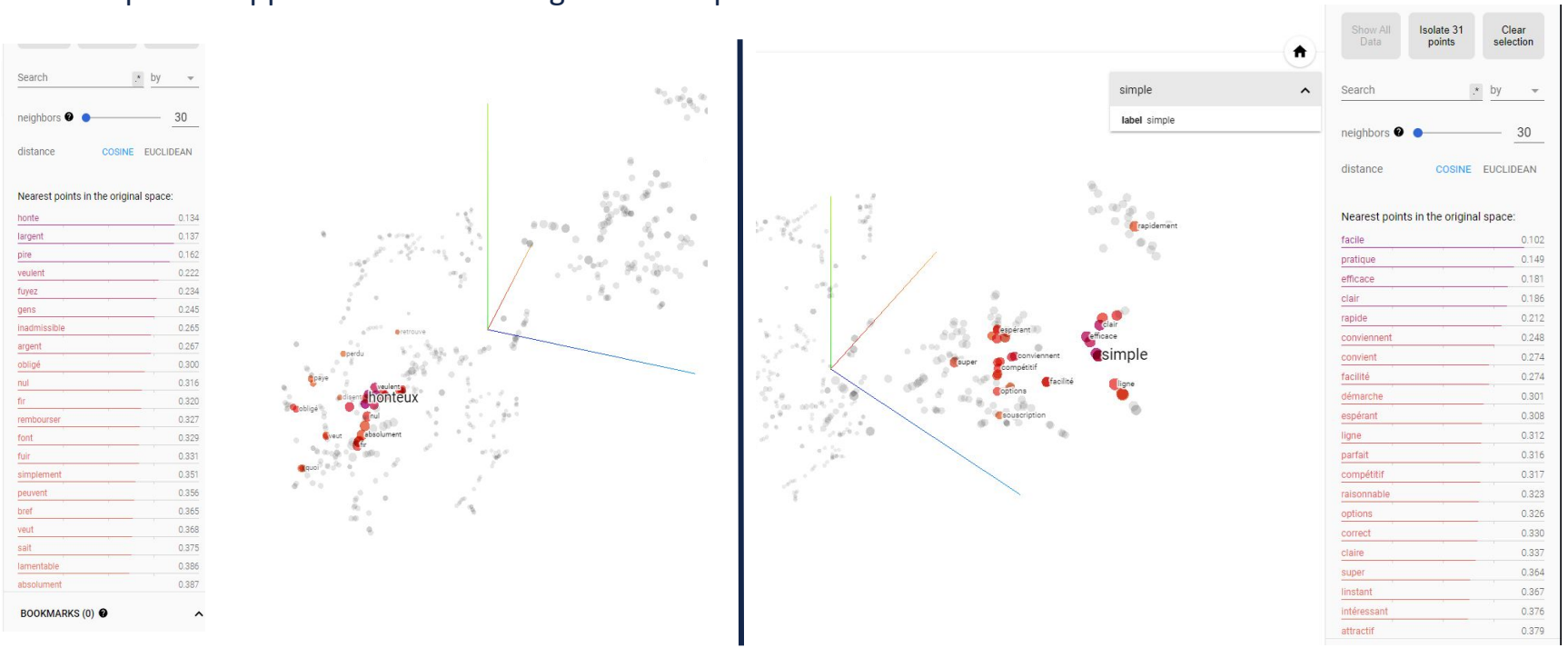
Objectifs

- **Création de word embeddings**
- **Visualisation**
- **Applications**

Entrainement d’un modèle Word2vec

Entrainement d’un model d’embedding sur nos données puis visualisation des embeddings avec Tensorboard

Termes positifs opposés aux termes négatifs sur le plan



Création d’un outils de comparaison sémantique

Cette fonction retourne les x mots les plus proche sémantiquement du mots choisi

```
most_similar_words("voiture", model_fr)
```

```
[('vehicule', 0.8452456593513489),
 ('garage', 0.7874598503112793),
 ('lassistance', 0.7833312749862671),
 ('risque', 0.7790943384170532),
 ('panne', 0.7755792140960693)]
```

Celle ci retourne la similarité cosinus entre 2 mots.

```
cosine_similarity('assurance','voiture')
```

```
0.36168456
```

3 | Modèles de machine learning/ Apprentissage non-supervisé Apprentissage supervisé

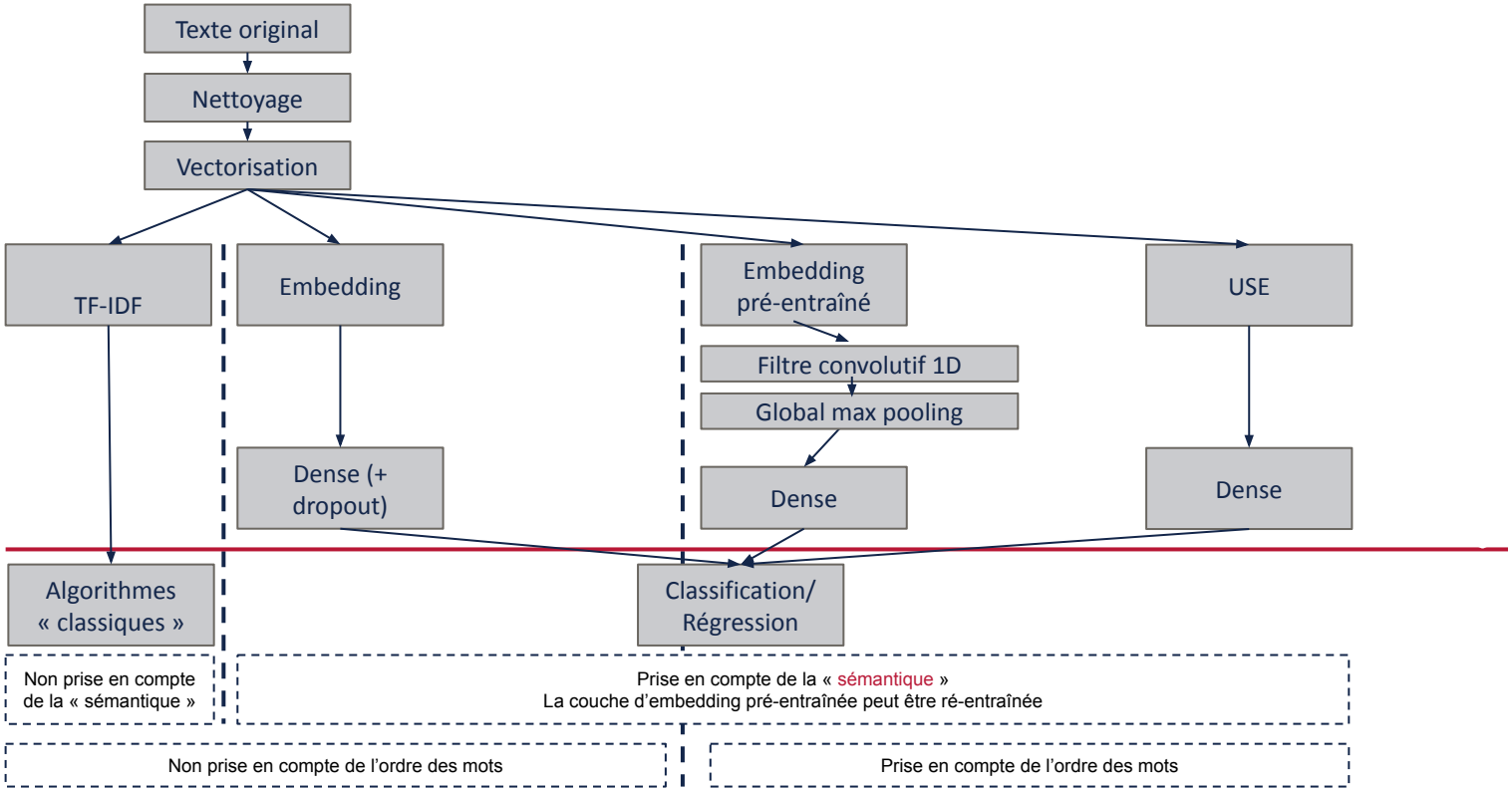
- **Présentation des approches**
- **Topic modelling et segmentation**
- **Apprentissage supervisé**
  - Prédiction de la note
  - Prédiction des thèmes

3.1 Présentation des approches

Objectifs

- **Prédire les notes en fonctions des avis**
- **Analyser les différences de résultats en fonction des approches**

Présentation des approches de modélisation



3.2 Topic modelling et segmentation

Objectifs

- **Identifier les topics et les mots-clés associés**

Illustration LDA - Topic modeling - Français

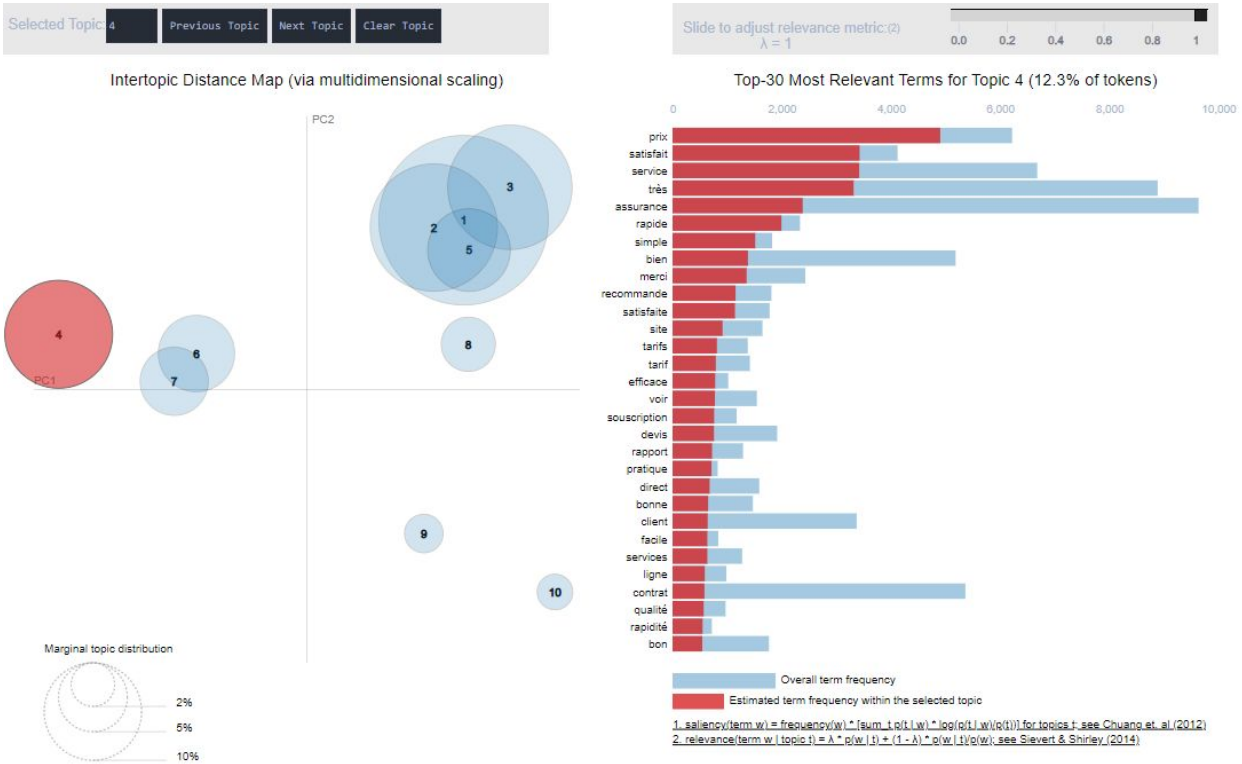
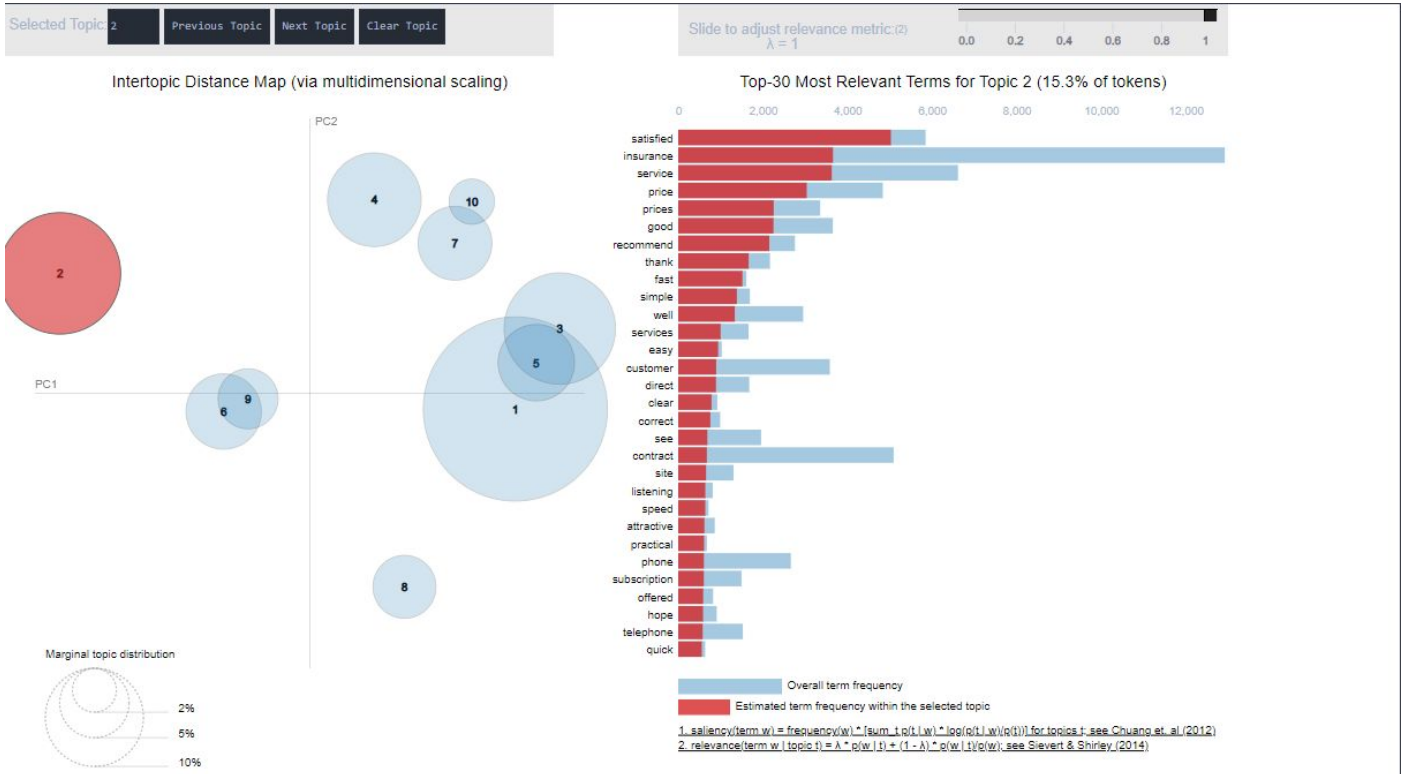


Illustration LDA - Topic modeling - Anglais



3.4 Classification - prédiction des thèmes

Objectifs

- **Comprendre le projet et son contexte**
- **Comprendre l’intérêt des modèles prédictifs**

Approche méthodologique

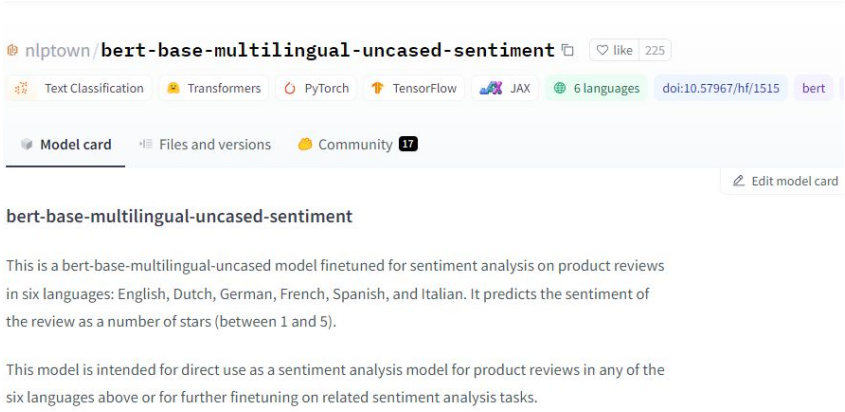
- **Dans la partie précédente, une liste de thèmes est définie avec des mots-clé associés. C'était une phase d'annotations de textes. Pour chaque commentaire, on peut alors détecter la présence de ce thème en vérifiant la présence des mots-clés. Mais il peut y avoir des faux positifs ou faux négatifs.**
  - Faux positifs : un mot-clé ne suffit pas de détecter le thème
  - Faux négatifs : on n'a pas repéré les mots-clés associés pour détecter ce thème
- **Pour remédier à ce problème, on peut entraîner un modèle avec la base labellisée, puis on réapplique ce modèle.**
  - On pourra examiner les scores élevés pour lesquels le label était initialisée négatif. Après vérification, nous pourrons confirmer s'il s'agit des faux négatifs et ainsi changer son label.
  - On fait de même pour les classes positives avec scores faibles.
  - Pourquoi ce processus pourrait fonctionner ?

3.3 Prédiction de la note

Objectifs

- **Comprendre le projet et son contexte**
- **Comprendre l’intérêt des modèles prédictifs**

Sentiment analysis

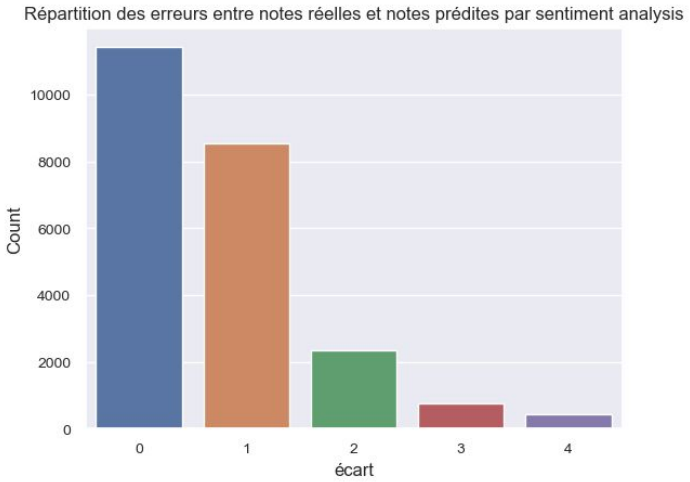


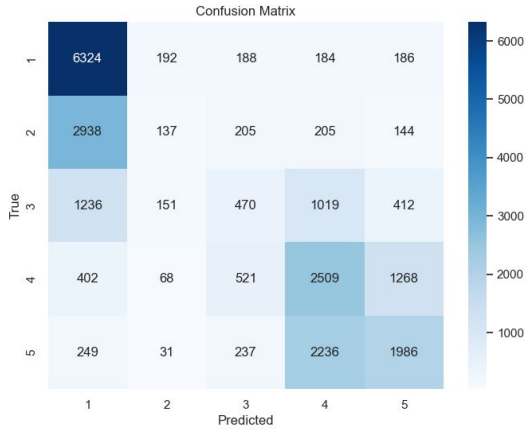
Classification multi-label : Notes de 1 à 5

Implémentation d'un modèle BERT d'analyse de sentiments multiclasse sur nos données

Sentiment analysis

Analyse des erreurs






```
Accuracy: 0.49
Classification Report:
              precision    recall  f1-score   support

           1       0.57      0.89      0.69      7074
           2       0.24      0.04      0.07      3629
           3       0.29      0.14      0.19      3288
           4       0.41      0.53      0.46      4768
           5       0.50      0.42      0.45      4739

    accuracy                           0.49     23498
   macro avg       0.40      0.40      0.37     23498
weighted avg       0.43      0.49      0.43     23498
```

Classification de textes avec Tensorflow

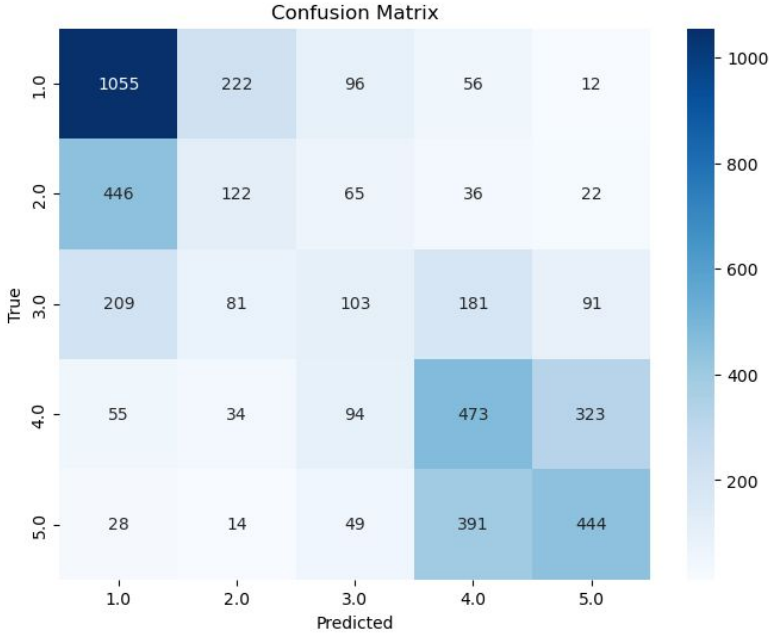
Model basique avec couche d'embedding

```
model = Sequential([
    Embedding(input_dim=max_words, output_dim=embedding_dim, input_length=max_sequence_length),
    Flatten(),
    Dense(64, activation='relu'),
    Dense(len(label_encoder.classes_), activation='softmax')
])
```

```
Accuracy: 0.47
Classification Report:
              precision    recall  f1-score   support

         1.0       0.59      0.73      0.65      1441
         2.0       0.26      0.18      0.21       691
         3.0       0.25      0.15      0.19       665
         4.0       0.42      0.48      0.45       979
         5.0       0.50      0.48      0.49       926

    accuracy                           0.47      4702
   macro avg       0.40      0.41      0.40      4702
weighted avg       0.44      0.47      0.45      4702
```



Classification de textes avec Tensorflow

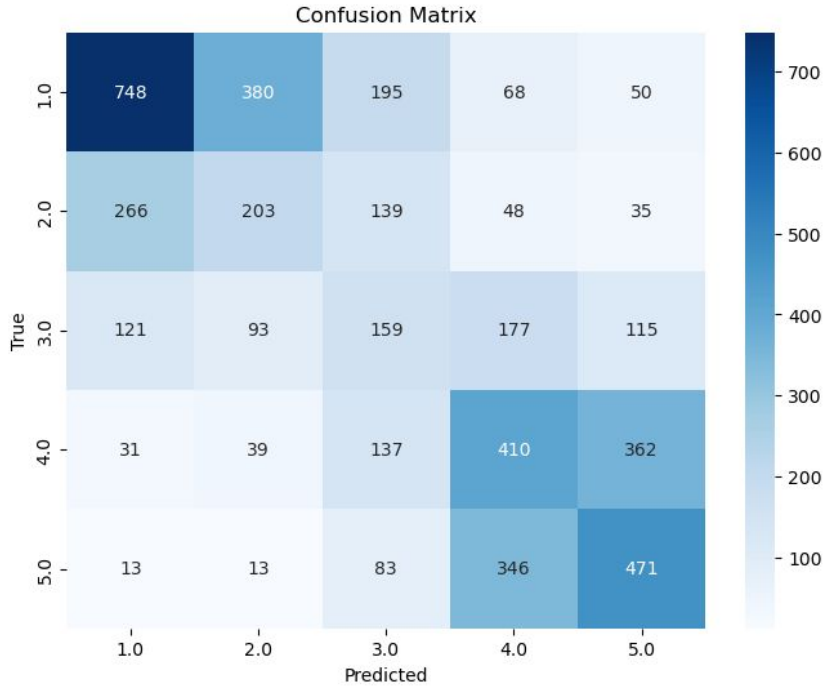
Model avec embedding pré-entrainé (Glove)

```
model = Sequential([
    Embedding(input_dim=max_words, output_dim=embedding_dim, weights=[embedding_matrix],
              input_length=max_sequence_length, trainable=False),
    Conv1D(128, 5, activation='relu'),
    GlobalMaxPooling1D(),
    Dense(64, activation='relu'),
    Dense(len(label_encoder.classes_), activation='softmax')
])
```

```
Accuracy: 0.42
Classification Report:
              precision    recall  f1-score   support

         1.0       0.63      0.52      0.57      1441
         2.0       0.28      0.29      0.29       691
         3.0       0.22      0.24      0.23       665
         4.0       0.39      0.42      0.40       979
         5.0       0.46      0.51      0.48       926

    accuracy                           0.42      4702
   macro avg       0.40      0.40      0.39      4702
weighted avg       0.44      0.42      0.43      4702
```



Classification de textes avec Tensorflow

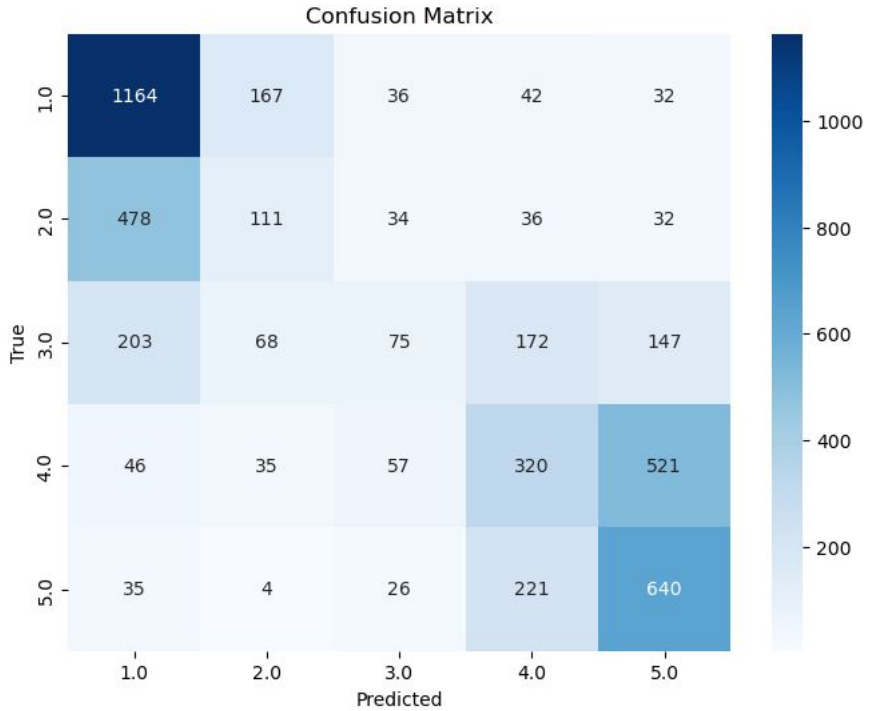
Model avec USE (Universal sentence embedding)

```
model = tf.keras.Sequential([
    tf.keras.layers.Input(shape=(512,)),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(len(label_encoder.classes_), activation='softmax')
])
```

```
Accuracy: 0.49
Classification Report:
              precision    recall  f1-score   support

         1.0       0.60      0.81      0.69      1441
         2.0       0.29      0.16      0.21       691
         3.0       0.33      0.11      0.17       665
         4.0       0.40      0.33      0.36       979
         5.0       0.47      0.69      0.56       926

    accuracy                           0.49      4702
   macro avg       0.42      0.42      0.40      4702
weighted avg       0.45      0.49      0.45      4702
```



Classification de textes avec Tensorflow

Les modèles ont des comportements similaire avec tout de mêmes des particularités.

On remarque notamment que les notes moyennes (3) ont beaucoup de mal à être prédites, les résultats pour cette classe sont proches de l'aléatoire

4 | Application

- Application streamlit de sentiment analysis (prédit une note de 1 a 5)

Application streamlit