Aide-Mémoire : Traitement du Langage Naturel (NLP)

1. Introduction au NLP

Le Traitement du Langage Naturel (NLP) vise à exploiter et analyser des données textuelles grâce à l'intelligence artificielle. L'objectif est de transformer des textes en représentations numériques exploitables.

---

2. Concepts Fondamentaux

2.1 Représentations Numériques

Nous devons convertir des textes en représentations numériques pour permettre aux algorithmes de les traiter :

1. **Bag of Words (Sac de mots)** :
   - Compte les occurrences des mots dans un corpus.
   - Utilisation de `CountVectorizer` pour transformer les phrases en matrices.
2. **TF-IDF (Term Frequency – Inverse Document Frequency)** :
   - Tient compte de la fréquence des mots tout en pénalisant ceux récurrents dans le corpus.

2.2 Techniques de Vectorisation

a) Bag of Words Exemple de sortie:

```
# Exemple d'une matrice Bag of Words
'Matrix': [[1, 1, 2],
           [0, 1, 0],
           [1, 1, 0]]
```

- Les colonnes de la matrice représentent les mots du corpus.

b) TF-IDF Formule clé : - `tf-idf(t, d) = tf(t, d) * log(N / df(t))` - **tf(t, d)** : fréquence d'un terme dans un document. - **df(t)** : nombre de documents contenant le terme. - **N** : nombre total de documents.

---

3. Analyse et Traitement des Textes

3.1 Nettoyage des Données (Data Cleaning)

Étapes Clés :

1. **Tokenisation** : Découpe des phrases en unités lexicales. Exemple : `I love to play` → `['I', 'love', 'to', 'play']`
2. **Suppression des Mots Vides (Stop-Words)** :
   - Élimine les mots courants comme **le**, **que**, ou **est**.
3. **Ponctuations** : Exclusion de la ponctuation, selon le contexte.
4. **Lemmatisation** : Réduction des mots à leur forme de base :
   - Exemple : *manger*, *mangé*, *mangera* → *manger*.

Exemple Nettoyage avec spaCy

```
corpus = ["j'adore les frites !", "Manger les frites est un régal !"]

def clean_sentence(sentence):
    clean_words = []
    for token in nlp(sentence):
        if not token.is_stop and not token.is_punct:
            clean_words.append(token.lemma_.lower())
    return ' '.join(clean_words)

corpus_nettoye = [clean_sentence(frase) for frase in corpus]
print(corpus_nettoye)
# Sortie : ["adorer frites", "manger frites régal"]
```

---

4. Comparaison et Recherche de Similarité

Méthodes de Similarité :

1. **Jaccard Score** : Mesure la similarité entre deux ensembles de mots via leur intersection et union.
2. **Cosine Similarity** : Évalue l'angle entre deux vecteurs dans un espace multidimensionnel (proximité).
   - Valeur : **-1 (opposés)**, **0 (indépendants)**, **1 (très similaires)**.

Exemple avec Cosine Similarity :

```
from sklearn.metrics.pairwise import cosine_similarity

query = "manger des frites"
query_vec = vectorizer.transform([query])

results = cosine_similarity(vectors, query_vec).reshape(-1)
top_match = results.argsort()[-1]  # Meilleur résultat
print(f"Phrase la plus similaire : {corpus[top_match]}")
```

---

5. Visualisation

Visualisation des Fréquences des Mots

Utilisation de librairies comme `plotly` pour afficher les mots les plus fréquents dans un corpus.

```
fig = go.Figure([go.Bar(x=df['unigram'], y=df['count'])])
fig.update_layout(title="Top 30 des mots fréquents")
fig.show()
```

---

6. Modélisation et Classification

a) k-Nearest Neighbors (k-NN)

- Algorithme basé sur la proximité entre vecteurs de texte.
- Pipeline combinant TF-IDF et `KNeighborsClassifier` pour une catégorisation multilabel :

```
from sklearn.pipeline import Pipeline
from sklearn.neighbors import KNeighborsClassifier

pipeline = Pipeline([
    ('vectorizer', TfidfVectorizer(max_features=200, ngram_range=(1, 2))),
    ('classifier', KNeighborsClassifier(n_neighbors=3))
])

pipeline.fit(x_train, y_train)
predictions = pipeline.predict(x_test)
```

---

7. Cas d'utilisation Pratiques

- **Moteurs de Recherche** : Regrouper et ordonner les résultats pertinents pour une requête.
- **Filtrage de Spam** : Identifier les emails indésirables.
- **Systèmes de Recommandations** : Suggérer des vidéos, articles ou produits similaires aux habitudes des utilisateurs.

---

8. Points Clés à Retenir

1. Nettoyer les données est une étape cruciale (tokenisation, stop-words, etc.).
2. Essayez différentes méthodes de vectorisation (Bag of Words, TF-IDF) pour trouver ce qui fonctionne le mieux.

3. Visualiser les données (comme les mots fréquents) donne des insights importants.
4. La performance de vos modèles (ex. : k-NN) peut être améliorée par une bonne préparation des textes.
