IFT3335 – TP2

Utiliser la classification pour la désambiguïsation de sens de mots

Ce TP est à réaliser en groupe de 2-3 personnes
Il correspond à 15% de la note globale
Date de la remise : 11 décembre avant 23:59.

(Compte tenu de la période des examens, il est toléré que la remise soit faite plus tard, mais avant le 20 décembre 2022, 23:59, et ceci sans pénalité. Après cette date butoir, la pénalité de 10% par jour sera appliquée)

Ce TP a pour but de pratiquer les algorithmes de classification (apprentissage automatique supervisé). Nous allons traiter le problème de désambiguïsation de sens de mots qui vise à classifier un mot utilisé dans un contexte dans la classe de sens appropriée. Les algorithmes de classification sont déjà implantés dans la bibliothèque Scikit-Learn (scikit-learn: https://scikit-learn.org/stable/). Votre travail dans ce TP consiste à utiliser Scikit-Learn sur des collections de données et à examiner l'impact de différents algorithmes et les différentes caractéristiques (features).

1. La tâche de désambiguïsation de sens de mots

La désambiguïsation de sens de mots consiste à déterminer le sens d'un mot dont plusieurs sens sont possibles. Par exemple le mot « souris » en français peut faire référence à un animal ou à un périphérique informatique. Dans le contexte de « Le chat attrape la souris », le sens référé est celui d'animal. Ce sens peut -être déterminé en utilisant le contexte du mot dans la phrase.

C'est une tâche de base pour la compréhension de textes. On effectue cette tâche typiquement en utilisant une approche de classification. On suppose qu'on a un ensemble de sens déjà définis, et on dispose d'un ensemble d'exemples d'entraînement - des textes (phrases) contenant des occurrences du mot ambigu, dans lesquels le sens de chaque occurrence du mot visé est annoté manuellement. En utilisant ces textes comme exemples, on entraîne un classifieur. Ce classifieur sera utilisé pour désambiguïser le mot dans de nouveaux textes, c'est-à-dire de classifier le mot dans le sens approprié.

Comme pour beaucoup de tâche d'apprentissage automatique, il est difficile de savoir quelle méthode est la plus appropriée pour désambigüiser le sens de mots. Il faut donc essayer plusieurs méthodes, avec des caractéristiques différentes, pour en choisir une qui fonctionne le mieux pour la tâche. Ce TP vous place dans ce contexte global, et on vous demande de tester différentes méthodes.

2. Préparatifs

Pour vous familiariser avec Scikit-Learn, consultez le tutoriel sur Scikit-Learn dans le cours MOOC en apprentissage automatique. On vous conseille aussi de lire la documentation de Scikit-Learn en ligne : https://scikit-learn.org/stable/

Les fonctions suivantes sont particulièrement importantes pour ce TP :

1. Prétraitement

Ceci vous permet de charger les données et faire des prétraitements sur les données, e.g. la sélection des données à traiter, transformation des données et attributs, etc. Pour commencer, importez un ensemble de données existant dans le package (e.g. iris dataset, digits pour de la reconnaissance d'écriture…), et essayez avec ces données.

2. Classification

Une fois les données chargées, vous pouvez maintenant choisir un algorithme et l'appliquer sur les données. Ceci se sépare en phase d'entraînement et phase de test.

Une fois l'algorithme choisi, importez-le puis initialisez-le. Testez sur les données de test pour voir la performance.

3. Sélection et pondération des features

Vous pouvez maintenant tenter de sélectionner des attributs à utiliser pour la classification.

Selon les données que vos traitez, vous pouvez avoir envie de ne pas considérer certaines dimensions ou certains caractéristiques (features).

Il y a différentes méthodes pour sélectionner un sous ensemble de caractéristiques à utiliser dans la classification. Ceci est utile quand vos données sont bruitées, avec beaucoup d'attributs/caractéristiques qui n'aident pas à la classification. Un nettoyage (une sélection) est très bénéfique dans ce cas. Cette sélection aide aussi à accélérer les traitements car on aura moins de caractéristiques à considérer.

Jouez librement avec les collections textuelles inclues dans Scikit-Learn. Notamment, vous devez transformer un texte en un ensemble d'attributs/caractéristiques. Par défaut, chaque mot différent est considéré comme un attribut ou une caractéristique (feature). Comme un mot peut apparaître plusieurs fois dans un texte, et son importance peut être différente selon sa fréquence d'occurrences, il est donc approprié d'utiliser la fréquence pour pondérer l'importance d'un mot dans un texte. Cette transformation en un vecteur de caractéristiques et de pondérer les caractéristiques sont réalisées dans Scikit-Learn grâce aux classes CountVectorizer, et TfidfVectorizer. La première classe permet de créer un vecteur de caractéristiques pondérées par leur fréquence, et la deuxième va pondérer les caractéristiques selon le schéma TF-IDF : TF – Term Frequenct, IDF – Inverse Document Frequency. Voir https://fr.wikipedia.org/wiki/TF-IDF pour une brève description.

Lisez les options proposées dans Scikit-Learn. Lors de votre choix, il vous est notamment possible de préciser si le résultat de cette transformation produit un ensemble d'attributs (mots) binaire (présent ou absent) ou avec un poids numérique (fréquence, tf transformé et avec idf).

Une pratique courante dans le domaine de classification de textes et de recherche d'information est de tronquer les mots pour ne garder que les racines. Par exemple, le mot « computer » sera tronqué en « comput ». Ceci a pour but de créer une représentation unique pour une famille de mots semblables (computer, computers, computing, compute, computes, computed, computation). Il est supposé que les différences morphologiques ne changent pas le sens de ces mots. Ce processus est appelé « stemming » (troncature).

Il y a des méthodes de stemming standards disponibles en python, dont celle proposée par la bibliothèque NLTK (voir ici pour un tutoriel sur la troncature avec NLTK). Vous allez utiliser le stemmer Porter – le plus répandu pour cette tâche. Pour intégrer le

stemmer de NLTK, voir la page https://scikit-learn.org/stable/modules/feature_extraction.html.

2.1. Corpus à traiter pour le TP

Pour ce TP, nous allons utiliser un ensemble de phrases en anglais annotées, contenant le mot ambigu *interest*, qui peut correspondre à 6 sens différents, selon le dictionnaire Longman :

Sense 1 = 361 occurrences (15%) - readiness to give attention
Sense 2 = 11 occurrences (01%) - quality of causing attention to be given to
Sense 3 = 66 occurrences (03%) - activity, etc. that one gives attention to
Sense 4 = 178 occurrences (08%) - advantage, advancement or favor
Sense 5 = 500 occurrences (21%) - a share in a company or business
Sense 6 = 1252 occurrences (53%) - money paid for the use of money

Le texte annoté contient le résultat d'une analyse de partie-de-discours (part-of-speech) + annotation de sens du mot *interest*. Voici un exemple :

```
[ yields/NNS ] on/IN [ money-market/JJ mutual/JJ funds/NNS ]
continued/VBD to/TO slide/VB ,/, amid/IN [ signs/NNS ] that/IN [
portfolio/NN managers/NNS ] expect/VBP [ further/JJ declines/NNS ]
in/IN [ interest_6/NN rates/NNS ] ./.
$$
[ longer/JJR maturities/NNS ] are/VBP thought/VBN to/TO indicate/VB
[ declining/VBG interest_6/NN rates/NNS ] because/IN [ they/PP ]
permit/VBP [ portfolio/NN managers/NNS ] to/TO retain/VB
relatively/RB [ higher/JJR rates/NNS ] for/IN [ a/DT longer/JJR
period/NN ] ./.
```

Dans cet exemple, les crochets [ ] enferment un groupe nominal. Chaque mot est suivi de sa catégorie grammaticale (e.g. /NNS), et le mot ambigu, *interest*, est annoté de son sens (_6, c'est-à-dire le 6ième sens). Les ponctuations sont elles-mêmes leur propre catégorie (comme dans ./. à la fin d'une phrase). Les phrases sont séparées par une ligne de $$.

Ce corpus contient 2369 instances de mot *interest*. Une description de ce corpus peut être trouvée ici : http://www.d.umn.edu/~tpederse/Data/README.int.txt. Le corpus qu'on utilise dans ce TP est pris du site http://www.d.umn.edu/~tpederse/data.html.

2.2. Le processus de désambiguïsation

Pour déterminer le sens du mot, on utilise les informations dans son contexte. Les informations contextuelles doivent être extraites au préalable (ceci n'est pas fait par Scikit-learn). Il y a plusieurs types d'informations contextuelles. Voici deux types d'information contextuelle :

- L'ensemble des mots avant et les mots après (dans un sac de mots, sans ordre). Dans le premier exemple proposés ci-dessus, si on retient les 2 mots avant et 2 mots après (le mot *interest*) sont {*declines, in*, *rate*, .}.
- Les catégories des mots autour. Pour les mêmes 4 mots du premier exemple, nous allons avoir : « NNS », « IN », « NNS », « . » (la ponctuation). Ces catégories sont

généralement prises en compte en ordre (C_{-2}=*NNS*, C_{-1}=*IN*, C_1=*NNS*, et C_2=.) afin de tenir compte de la structure syntaxique.

Ces deux groupes de caractéristiques sont ceux que vous devez utiliser au minimum. Mais il y a certaines variations possibles (que vous pouvez tester) :

- Lors de sélection des mots autour, il est possible de ne pas tenir compte des mots outils (stopword en anglais) très fréquents et peu informatifs, tels que *in* ou les ponctuations. (*in* et *further* sont des mots outils – voir la liste de stopwords en anglais). Cette option est incluse dans les algorithmes Scikit-Learn.
- Troncature des mots, en utilisant un algorithme de stemming, comme expliqué plus tôt.

Dans la littérature, d'autres types de caractéristiques ont été proposées et utilisées. On vous conseil de consulter la page suivante pour une présentation sommaire :
https://en.wikipedia.org/wiki/Word-sense_disambiguation
Vous êtes encouragés à explorer d'autres caractéristiques. L'utilisation des caractéristiques supplémentaires sera prise en compte. Si ces caractéristiques supplémentaires sont non-triviales, des points de bonus peuvent être accordés dans la correction.

2.3. Les tâches à réaliser

1. **Préparatif :**
   Vous devez faire un programme capable d'extraire les caractéristiques à partir des textes annotés. Réfléchissez et testez en particulier le choix des éléments à utiliser comme caractéristique pour la classification.
2. **Tâche de base**
   Vous devez tester la performance de différents algorithmes de classification. Pour ce TP, on vous demande de tester les algorithmes suivants disponibles dans Scikit-Learn, qui sont présentés dans le cours MOOC :
   a. Naive Bayes : Choisissez Multinomial Naïve Bayes https://scikit-learn.org/stable/modules/naive_bayes.html#multinomial-naive-bayes
   b. Arbre de décision : https://scikit-learn.org/stable/modules/tree.html#classification
   c. Forêt aléatoire : https://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees
   d. SVM : https://scikit-learn.org/stable/modules/svm.html
   e. et MultiLayerPerceptron (en essayant différents nombres de neurones cachés) : https://scikit-learn.org/stable/modules/neural_networks_supervised.html#multi-layer-perceptron

   Pour ces algorithmes, utilisez d'abord les 2 types de caractéristique décrits en haut. Testez avec différentes tailles de fenêtre de contexte (1, 2, 3, …, ou même toute la phrase), et observez les variations de performance de désambiguïsation en fonction de cette taille. Cependant, ne faites pas ceci pour tous les algorithmes (il y en a trop). Testez seulement sur un algorithme, et vous supposez que ces caractéristiques produisent les mêmes effets sur les autres algorithmes.
3. **Tâches supplémentaires (optionnelles)**

Réfléchir à des caractéristiques pouvant être intéressantes à explorer, et justifier pourquoi. Définissez et implémentez ensuite une méthode intéressante pour les tester.

4. **Méthode supplémentaire (optionnelle)**
   En plus des méthodes d'apprentissage automatique classiques, on utilise de plus en plus l'apprentissage profond pour des tâches en traitement de la langue naturelle. Notamment, les études récentes exploitent généralement un modèle pré-entraîné comme BERT. Dans cette tâche optionnelle, vous êtes invités à explorer l'utilisation de BERT pour la classification du sens de mots. Un modèle BERT de base est utilisé pour générer une représentation pour la phrase. Cette représentation sera passée à une couche de perceptron pour faire la classification. Explorer cette option si vous avez fait les tâches obligatoires, et que vous avez du temps et des habilités à explorer BERT (TensorFlow, Karas, …).
5. **Analyse dans un rapport**
   Analysez les résultats de classification avec différents algorithmes, et comparer leur performance. Comme réflexion globale, vous êtes invités à réfléchir sur la pertinence de chaque algorithme pour traiter ce problème de classification de sens de mots, ses forces et ses faiblesses, et des améliorations possibles.

3. À rendre

Vous devez rendre tous les programmes utilisés pour l'extraction des caractéristiques. Allouez une petite de votre rapport pour donner une description de ces programmes et de leur utilisation.

En plus des programmes, vous devez aussi rendre un rapport, de longueur d'environ 10 pages. Dans votre rapport, vous devez décrire votre pré-traitement, les expériences que vous avez réalisées, les résultats obtenus, et enfin des comparaisons et des analyses sur les résultats. Vos analyses doivent porter, au minimum, sur la performance des différents algorithmes, l'impact de différentes options - stemming, le nombre de neurones cachés, la taille de fenêtre pour la désambiguïsation. Vous êtes encouragés à faire des analyses sur d'autres aspects. Décrivez librement ce que vous observez d'intéressant dans ces expériences.

4. Barèmes d'évaluation

Ce TP correspond à 15% de la note globale. Voici les critères utilisés pour noter ce TP :

- Programme d'extraction de caractéristiques : 2 points
- Tests avec Naïve Bayes : 2 points
- Tests avec arbre de décision : 2 points
- Tests avec forêt aléatoire : 2 ponts
- Tests avec SVM : 2 points
- Tests avec MultiLayerPerceptron : 2 points
- Rapport : 3 points (on tient compte de la description, les analyses et comparaisons entre différents algorithmes et options, les analyses de résultats et les conclusions. La bonne structure et la clarté de description sont attendues.)
- Bonus : Jusqu'à 2 points de bonus si des caractéristiques non-triviales sont développées et testées, et/ou si vous avez inclus la méthode basée sur BERT dans vos tests et analyses.