Proposition de projet

GIF-7005: Introduction à l'apprentissage machine

Équipe 10: Stéphane Caron, Philippe Blais, Philippe Blouin-Leclerc, Samuel Lévesque et William Bourget

26 octobre 2018

Description du projet

Dans le cadre du projet à réaliser dans le cours GIF-7005, notre équipe avons choisi la problématique proposée par l'entreprise Coveo. Le problème consiste à bâtir un modèle d'apprentissage supervisé qui, à partir d'une nouvelle recherche, retourne un ensemble d'au plus 5 documents pertinents à un utilisateur donné.

Jeu de données

Pour entraîner notre modèle, nous aurons accès à des données portant sur différentes recherches réalisées par des utilisateurs. Nous pourrons donc utiliser certaines informations générales sur l'utilisateur et sur la recherche effectuée, par exemple:

- le langue de la recherche
- le pays de provenance
- la ville
- la requête effectuée
- ....

De plus, nous aurons accès aux clics effectués par l'utilisateur en réponse à sa recherche. Ces clics nous donnent l'information sur les documents choisis par celui-ci. Cela sera d'ailleurs ce que nous tenterons de modéliser avec notre algorithme d'apprentissage.

Méthodologie anticipée

Pour bâtir notre modèle, nous prévoyons utiliser deux types de méthodes. Dans la première phase, nous pensions commencer par utiliser un modèle simple et intuitif dans un contexte de classification. Pour ce faire, nous pensions utiliser un algorithme commme celui des *k-PPV* (Alpaydin 2010).

Dans la seconde phase, nous prévoyons tester des réseaux de neurones (Goodfellow, Bengio, and Courville 2016), qui sont d'ailleurs très populaires pour les systèmes de recommendation.

Les méthodes citées plus haut sont des méthodes supervisées. Cela veut dire que le modèle apprend à prédire une réponse connue. Nous aimerions aussi tester si des méthodes non-supervisées pourraient nous aider à entraîner des modèles supersivés par la suite. En d'autres mots, nous allons considérer la possibilité de regrouper les caractéristiques des recherches dans des groupes similaires et utiliser cette information comme variable dans notre modèle supervisé.

Finalement, nous prévoyons aussi utiliser le concept de *word2vec* pour trouver des représentations vectorielles des mots qui feront partis de nos documents et des titres de nos documents. On utilisera cette librairie pour transformer nos données textuelles en vecteurs comparables entre eux. On pourra ainsi trouver des requêtes similaires et voir si leurs documents pertinents sont les mêmes. Si c'est le cas, on pourra se servir de l'historique de clicks des requêtes similaires pour proposer des documents pertinents.

Plan détaillé du projet

Voici un plan par étapes que nous souhaitons suivre pour la réalisation de ce projet:

- Importation et nettoyage des données d'entraînement, de validation et de test
- Faire les correspondances entre requêtes et clicks
- Vectoriser les requêtes pour qu'elles soient utilisables avec des algorithmes de classification et de régression standards. On souhaite notamment transformer les données textuelles en vecteurs numériques à l'aide de la librairie *word2vec*. On souhaite également extraire l'information sur l'utilisateur et la visite en question question d'avoir des informations plus personnelles sur l'utilisateur. On pense se servir de ces paramètres historiques, comme les clicks passés de l'utilisateur, pour générer des suggestions similaires.
- Vectoriser l'information sur les documents pour avoir une représentation numérique et vectorielle du document. Malheureusement, nous n'avons pas accès au contenu des documents, mais nous sommes en mesure d'avoir une idée du contenu en se servant du titre. On se servira encore une fois de la librairie *word2vec* pour faire cette conversion. On souhaite également utiliser des approches plus classiques en traitement de la langue naturelle, telles des matrices d'occurences avec la fonction *CountVectorizer* de la librairie *scikit-learn*.
- Utiliser les caractéristiques des documents avec celles des requêtes pour optimiser un modèle k-PPV axé sur la distance pour faire de la classification multi-classes (chaque classe étant un document). On peut attribuer un score à chaque document pour une requête et on retourne les 5 documents ayant le score le plus haut. Tester plusieurs paramètres.
- Utiliser les mêmes caractéristiques des requêtes, mais sans celles des documents pour optimiser des modèles de classification multi-classes tels *Naive Bayes* et les *SVM* avec un nombre de classes égal au nombre de documents. Tester plusieurs hyperparamètres pour optimiser les hyperparamètres.
- Utiliser ces mêmes caractéristiques dans un modèle neuronal avec un nombre de sorties égal au nombre total de documents et utiliser les caractéristiques d'entrée comme couche d'entrée. Tester plusieurs combinaisons de fonctions d'activation et de nombre de neurones/couches pour obtenir une solution satisfaisante.

Références

Alpaydin, Ethem. 2010. *Introduction to Machine Learning.* 2nd ed. The MIT Press.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning.* MIT Press.