Piscine python avancé - TP fil rouge

Vinh Pham-Gia (vinhpg45@gmail.com) — Long Do Cao (ldocao@krystals.ai)

Mars 2020

Date attendue de rendu : Au moins un commit au 16 mars 2020 à 18h

1 Pré-requis

Vous êtes libre de choisir votre environnement de développement (jupyter notebook accepté).

Nous vous conseillons de mettre en pratique les notions explorées au cours de cette piscine. A titre d'exemple, vous pouvez initialiser un repo Git qui nous permettra de revoir votre code et de faire des commentaires.

2 Objectifs de ce TP

L'objectif de ce TP est de construire un jeu de données agrégées. Ces agrégations sont courantes et sont utilisées par les algorithmes de *Machine Learning* ou *Deep Learning* (non traités lors de ce module).

Pour cela, nous disposons d'un jeu de données **initial_dataset.csv** qui contient le chiffre d'affaires (CA) quotidien d'une enseigne d'e-commerce de 2017 à 2019. Ce CA est réparti par produits vendus (ordinateurs, téléphonie, accessoires) et par magasin (Paris, Bordeaux, Mont de Marsan).

Nous souhaitons transformer ces données pour obtenir le CA des villes du Sud-Ouest (Bordeaux et Mont de Marsan) par jour pour les ventes de périphériques (ordinateur et téléphone). Nous voulons aussi des colonnes qui permettent de caractériser notre donnée (*features*).

Le code proposé doit être suffisament **flexible** pour pouvoir récupérer le jeu de données pour un type de produit en particulier (ordinateur, téléphone) ou pour l'ensemble des deux. Bien entendu, l'option de créer une colonne par type de produits est à exclure (imaginez que le catalogue référence plusieurs milliers de références différentes...)

Pour des soucis de compatibilité, le jeu de données doit être nettoyé pour ne contenir que **des caractères alphanumériques compatibles avec le clavier anglais (pas d'accent)**.

Les colonnes désirées pour le fichier de sortie sont les suivantes :

- date
- ca : chiffre d'affaires cumulé pour les villes (Bordeaux et Mont de Marsan) et les périphériques (ordinateur et téphone)
- ca_last_year : chiffre d'affaires du même jour pour l'année passée
- ca_last_year_same_weekday : chiffre d'affaires du jour ouvré similaire de l'année passée (ex : pour le mercredi 13 février 2019, prendre le CA du mercredi 14 février 2018)
- weekday : jour de la semaine
  Valeurs acceptées : ['Monday', 'Tuesday', ..., 'Sunday']
- is_weekend : booléen pour déterminer si la date considérée est un jour de week-end
- is_bankholiday : booléen pour déterminer si la date est un jour férié
- distance_between_closest_bank_holiday : nombre de jours (en absolu) qui sépare la date considérée du jour férié le plus proche
- is_school_holiday : booléen pour déterminer si la date considérée est un jour de vacances scolaires (pour chaque zone, créer un booléen avec la zone en suffixe)

N.B : Pour revenir à nos algorithmes de Machine Learning ou Deep Learning: un cas d'usage envisageable serait par exemple de développer un module de prévision du chiffre d'affaires en fonction des caractéristiques du jour considéré.

3 Livrables attendus

1. Fichier **processed_data_{equipement}_v1.csv** contenant les colonnes précédemment listées (pensez à valider que le fichier ne contient qu'une ligne par jour)
2. Graphe qui représente l'évolution du CA mensuel de 2017 à 2019

4 Conseils et recommandations

Afin de traiter ce TP, vous pouvez utiliser les pistes de réflexion suivantes :

- Utiliser les fonctions déjà existantes pour extraire les informations souhaitées d'une date.
- Récupérer les jours fériés et vacances scolaires via les APIs python jours-feries-france et vacances-scolaires-france.