

Dans ce qui suit, google est votre ami.

I/ Tutoriel analyse et machine learning en python

Vous trouverez un petit set de données accompagné d'un notebook qu'il vous faudra compléter.

Si vous n'avez jamais fait de python, ce n'est pas grave. Le test est construit de façon progressive, pour vous accompagner dans la montée en complexité. Après avoir installé *python3*, *matplotlib*, *numpy*, *sklearn* et *jupyter* (ces installations ne sont pas compliquées, sinon vous pouvez passer par anaconda), ouvrez et complétez le notebook (*test_python_data_science_Dataswati.ipynb*).

II/ Analyse biblio

Pour cette partie, nous vous laissons l'occasion de montrer votre capacité à synthétiser et formaliser des questions ouvertes pour faire des propositions d'expérimentation, et en même temps en profiter pour découvrir vous aussi de nouveaux sujets et en apprendre un peu plus sur l'état de l'art.

Il faut garder en tête que nous sommes une startup, et que le temps dont nous disposons pour un sujet donné est limité, il faut donc faire rapidement un arbitrage entre le niveau de profondeur dans lequel on rentre dans un sujet, la capacité à comprendre et synthétiser, puis tester et valider des concepts très variés et différents (ce qui peut être très frustrant au passage).

Nous allons vous faire une proposition de questions ouvertes que nous n'avons pas encore eu le temps de traiter, et nous vous laissons carte blanche pour faire une recherche bibliographique, valider l'intérêt, penser un périmètre d'expérimentation, des modalités de validation, etc.

Le rendu minimal attendu est une synthèse courte (de 2-3 pages maximum) du sujet choisi et des pistes envisagées, avec une sélection de quelques publications, articles de blog ou dépôts de code pertinents (~5 à 10 refs max), l'intérêt pour notre question, et un protocole d'expérimentation. Ne cherchez pas être exhaustif, ni à maîtriser le sujet en profondeur dans le temps imparti ! Le but est de montrer votre compréhension des problématiques, votre priorisation des sujets et votre démarche expérimentale.

Pour la recherche, google reste votre ami, mais nous recommandons fortement <http://www.arxiv-sanity.com>. Ne négligez pas certains articles de blogs, ou des dépôts github, car la communauté machine learning publie intensivement des choses pointues sur le web, hors cadre académique.

Essayez de faire l'exercice en quelques demi-journées, car c'est le temps dont on dispose en moyenne en conditions réelles. Un code illustratif est un grand plus, si vous avez identifié des dépôts ou des librairies en python, ou utilisables depuis python (C, C++). Vous pouvez par exemple regarder ici : <https://paperswithcode.com>

Pensez-vous à faire des mindmaps, utiliser des posts-its, ou tout autre méthode que vous préférez pour une gestion efficace et évitez de trop vous disperser. Nous espérons que cet exercice est une opportunité pour vous de commencer à rentrer dans des sujets Data Science.

Parmi les nombreuses données que nous traitons, les **time series** reviennent très souvent. On peut distinguer plusieurs types : capteurs réguliers avec des valeurs continues, variables discrètes de type automate/actionneurs, relevés manuels échantillonnés de façons irrégulières, etc.

L'ensemble de ses variables caractérise le process industriel, et amène généralement des informations complémentaires, et il est souhaitable de pouvoir toutes les exploiter. Nous sommes également confrontés au fait que les clients veulent anticiper des événements négatifs, mais sans avoir de données labélisées.

Dans ce cas nous passons par des approches **semi-supervisées** ou **non-supervisées**, par exemple en passant par du **clustering**, de la **propagation de labels** et/ou de l'**active learning**, avant de repasser sur de la classification lorsque les données ne sont pas trop déséquilibrées, ou alors travailler en **détection d'anomalies** lorsque les événements restent rares.

Tout l'enjeu consiste donc à **construire des représentations et des distances pertinentes** pour manipuler algorithmatiquement et comparer

Dans le cadre ce test, nous vous proposons d'**expérimenter et de comparer différentes représentations de séries temporelles**, en travaillant par exemple sur un cas de clustering (il faut donc construire ou trouver un dataset pour expérimenter). Certaines sont calculées, d'autres sont apprises, plusieurs sont expérimentales et leur pertinence et leur robustesse doivent être testées. Parmi celles que nous avons identifié (vous pouvez en chercher d'autres) :

- Il est aussi possible de **travailler directement sur les valeurs des séries elles-mêmes**, et de travailler avec une distance comme le **dynamic time warping** qui compare les formes. Cette approche reste calculatoire et n'est robuste que pour des séries aux formes régulières.
- Pour des séries temporelles « régulières et continues », l'approche la plus simple reste d'extraire différentes valeurs clés calculées par exemples sur des bins (voir **tsfresh** par exemple). C'est l'approche que nous privilégions en ce moment, et peut représenter un **benchmark** pour comparer les autres approches.
- Des approches qui cherchent à encoder les séries temporelles de différentes manières :
 - symboliques, comme **SAX**
 - basées sur des motifs : **matrix profile** et **shapelets**
- Une approche idéale consisterait à trouver des représentations invariantes qui capturent les propriétés intrinsèques de la dynamique sous-jacente. Nous avons identifié les 2 sujets suivants, qui se développent mais restent encore peu maîtrisés :
 - **Décomposition en modes dynamiques (DMD)**, par exemple avec l'opérateur de **Koopman**
 - **Analyse topologique (TDA)** en cherchant des **homologies invariantes/persistantes**

En terme de distance, vous pouvez utiliser des distances géométriques (euclidienne, etc), statistiques (kolmogorov-smirnov, wasserstein, etc), apprises (metric learning).

Le rapport, le périmètre de l'expérimentation et le code éventuellement fourni doivent rester minimalistes, sans chercher à être exhaustif. L'idée est simplement de comprendre le sujet et de se faire rapidement une idée de ce qui peut être fait ou non. Il existe déjà des bibliothèques sous python pour ces sujets. Nous vous encourageons à repartir de ce qui existe déjà par ailleurs.