Examen de statistiques descriptives

(Jérôme Lacaille et Florent Forest)

Vous soumettrez votre réponse sous la forme d'un fichier notebook (MACS3-SD20-Prenom_Nom.ipynb) que vous enverez par courrier électronique avant le 1er décembre aux deux adresses suivantes :

- jerome.lacaille@gmail.com
- forest@lipn-univ-paris13.fr.

Ce notebook utilise la toolbox 'tabata' que vous trouverez sur GitHub comme précisé dans le cours : https://github.com/jee51/tabata.

L'examen se compose de plusieurs questions, chaque question compte pour un nombre de points défini. La note totale sera la somme des points obtenus par votre présence lors des cours et TD (1/2 point par séance, soit 6 points en tout en comptant les TD), plus les points accumulés par cet examen (20 points), la note finale sera majorée à 20.

Attention cependant, toute journée de retard compte pour un demi point de moins et une recopie évidente (plagiat) du travail d'un de vos camarades compte pour -4 points par contrevenant.

Deadline avant pénalité : le 1er décembre à 0h00 (30 novembre minuit).

Bon travail !

Corrections

Le tableau ci-dessous servira d'évaluation, vous voyez ainsi comment la notation sera découpée en éléments de base. Surtout n'hésitez pas à commenter vos codes et ne laissez pas d'affichages sans légende.

Question	Elément	Points	Note
A Théorie	A.2 Proba/Stats	+1	
	A.3 Modèles linéaires	+1	
	A.4 Modèles non linéaires	+1	
	A.5 Apprentissage	+1	
	A.6 Machines de Boltzmann	+1	
---	---	---	---
B Pratique	B.1 Une interface data	+3	
	B.2 Comprendre un algo	+6	
	B.3 Créer un modèle	+6	
---	---	---	---

Les zones en jaune, rédigées en rouge, ci-dessous sont les questions qui vous sont posées.

A. Questions de cours

Cette partie contient quelques questions de cours permettant de m'assurer que les étudiants qui n'ont pas pu suivre un cours ont visionné la présentation en différé. Ces questions sont numérotées de 2 à 6, sachant que le premier cours a été présenté à l'université, les absents s'étant excusés (ceux qui ont suivi tous les cours et TD ont déjà 6 points d'avance).

Il s'agit de 5 couples de questions simples auxquelles vous répondrez dans la zone de texte juste après la question. Pour chaque cours une question porte sur la partie théorique et une autre sur les présentations industrielles.

A.2. "Probabilités et statistiques"

Jeudi 24 septembre.

A.2.1. Qu'est-ce qu'un boxplot ?

A.2.2. Décrivez l'algorithme très simple qui a été utilisé pour surveiller des mesures sur des bancs d'essais de moteurs (on en a aussi un peu reparlé la semaine suivante).

A.3. "Modèles linéaires"

Jeudi 1er octobre

A.3.1. A quoi sert le Lasso ?

A.3.2. Pourquoi doit-on rendre des mesures indépendantes du contexte d'acquisition pour surveiller un moteur ? Quelle solution est proposée dans le cours ?

A.4. "Modèles non linéaires"

Jeudi 8 octobre

A.4.1. Expliquez comment on construit une forêt aléatoire pour faire de la classification d'observations ?

A.4.2. Quel a été l'algorithme utilisé pour représenter l'état d'un moteur sur un écran ? A-t-il fallu opérer quelques prétraitements ?

A.5. "Apprentissage"

Jeudi 15 octobre

A.5.1. Qu'entend-on par la notion de "confiance" en un apprentissage ? Donnez votre avis sur les relations existant entre confiance, précision et taille de l'échantillon d'apprentissage. Que signifie qu'un modèle est robuste ?

A.5.2. Quel technique (assez simple) a été utilisée pour construire des tubes de confiances autours d'enregistrements de mesures ?

A.6. "Machines de Boltzmann"

Jeudi 22 octobre

A.6.1. Quel est le rôle de la température dans un recuit simulé ?

A.6.2. Comment peut-on tester si un capteur est en panne ? Que faire dans ce cas ?

B. Questions techniques

Cette question va vous obliger à regarder un code en construction, pas forcément écrit de manière parfaite mais vous allez devoir comprendre et expliquer son fonctionnement. Le code en question est dans la toolbox 'tabata' (https://github.com/jee51/tabata) que vous pouvez installer sur votre machine ou cloner dans un environnement "Google Colab" par exemple. Vous avez du apprendre à faire cela avec Florent en TD.

In [2]:
```
# Sous Colab, supprimez le # de la ligne suivante.
#!git clone https://github.com/jee51/tabata
```

In [1]:
```
import os
import tabata as tbt
```

B.1 Une interface data

Nous avons vu en cours que pour collaborer facilement à plusieurs, il vaut mieux fixer à l'avance quelques contraintes sur les interfaces. Dans 'tabata' on va travailler sur des listes de signaux : chaque observation est un signal multivarié formé de plusieurs variables enregistrées au cours d'un intervalle de temps. Les signaux sont stockés naturellement dans des DataFrames 'pandas', pour les listes de signaux on a construit un objet `Opset` assez sympatique car il contient en particulier une fonction d'affichage interactive.

Un exemple de jeu de données est fourni avec 'tabata', ce sont des vols successifs d'un avion. Il y a 6 variables : l'altitude, une température mesurée dans le moteur qui reflète un peu la température extérieure, la vitesse air (Total Air Speed), la vitesse verticale, le poids de l'avion et une mesure de poussée du moteur.

In [2]:
```
# Exécutez ce code pour charger les données et voir l'affichage interactif.
datafile = os.path.join(os.path.dirname(tbt.__file__),'notebooks/data/in/AFL1
S = tbt.Opset(datafile)
S.plot()
```

(*Cette toolbox utilise Plotly pour les affichages ce qui peut poser certain problèmes si vous n'utilisez pas un environnement standard, JupyterLab par exemple. Il faudra si vous préférez cet environnement que vous installiez les extensions nécessaires. Des scripts d'installation sont proposés.*)

In [ ]:
```
# Sous "Google Colab" utilisez les méthodes '.ploc()'.
S.plotc()
```

L' `Opset` offre d'autres interfaces, vous pouvez les explorer en lançant le notebook que vous trouverez sous 'tabata/notebooks/opset_doc.ipynb'. L'exécution de ce code va créer une version épurée des données 'AFL1EB_C.h5' qui correspond au résultat d'une partie de l'examen de l'an passé.

(*Prenez le temps de regarder les autres notebooks exemples qui se trouvent dans ce répertoire.*)

B.1.1. Avez vous des idées pour améliorer cette API (Application Programming Interface) ? Pourquoi ai-je choisi d'utiliser la librairie Plotly plutôt que Matplotlib ? Que pensez-vous de l'interactivité réalisée sous le notebook ? Est-ce que le notebook n'est pas un moyen de collaboration pratique ? Pourquoi ?

B.1.2. Il est facile avec des DataFrames pandas de faire des statistiques sur une variable, mais imaginons que l'on souhaite par exemple sortir la distribution des altitudes maximales à l'aide d'un graphe sympatique : une boîte à moustaches par exemple. Comment feriez vous ? Proposez un code.

In [ ]:

B.2 Comprendre un algorithme

La toolbox 'tabata' contient un outil assez amusant qui permet de sélectionner des instants. Voyons comment cela fonctionne.

In [3]:
```
# N'oubliez pas d'exécuter le code 'tabata/notebooks/opset_doc.ipynb' avant p
cleanfile = os.path.join(os.path.dirname(tbt.__file__),'notebooks/data/out/AF
S = tbt.Selector(cleanfile)
S.plot()
```

1. Depuis l'onglet 'Plot' cliquez sur un point qui vous intéresse de la courbe Altitude : par exemple la fin de la montée de l'avion. Puis passez à l'enregistrement suivant et recommencez, faites cela sur quelques enregistrements. Vous pouvez sauter des enregistrements qui ne vous plaisent pas, le 'record_10' par exemple (le n°6 dans la liste). pas la peine d'en faire trop, une dizaine de clics devraient suffire, mais sélectionnez bien toujours le même point d'intérêt sur chaque courbe.
2. Passez sur l'onglet 'Learn'. Tout en laissant ALT sélectionné, cliquez sur Learn. Le bouton doit être bleu, il va passer à l'orange quand l'apprentissage va s'exécuter, puis deviendra vert à la fin.
3. Retournez sur l'onglet 'Plot'. La courbe rouge en dessous de l'affichage de l'altitude montre une bosse. On dirait un probabilité de présence du point que l'on a cliqué.
4. Regarderz d'autres enregistrements. Voyez ! Sur des enregistrements que vous n'avez pas encore vu l'algorithme a deviné ce que vous souhaitiez lui monter.

In [ ]:
```
# Même chose que tout à l'heure.
# Sous "Colab" utilisez '.plotc()'
S.plotc()
```

Sous "Google Colab" la sélection d'un point sur une courbe ne fonctionne pas, vous utiliserez la scroll-barre 'Point'. Attention, d'une observation à l'autre le point de sélection est concervé, vous le verrez en faisant un aller-retour (Next-Previous). Pensez donc à en changer la position, et si vous voulez sauter une observation, ramenez le point en 0.

B.2.1. Comment ça marche ? Quelle a été l'astuce ?

(*Cette question est sans doute la plus difficile de l'épreuve. J'attends une réponse claire. Avez vous d'autres idées, pourquoi ce truc marche-t-il ?*)

B.2.2. En étudiant l'API du `Selector`, vous allez voir qu'il est possible d'extraire un intervalle entre deux points d'intérêts. Faites cela, créez un dataset temporaire.

(*Facile ! C'est déjà fait dans les notebooks examples.*)

In [ ]:

B.3 Créer un modèle.

Participer à un développement collaboratif.

B.3.1 Le détecteur d'instants que vous venez de manipuler est encore assez frustre, il faudrait peut-être créer des indicateurs plus malin. Avez-vous des idées ?

Nous avons vu en cours qu'un tube de confiance est un outils assez sympatique pour détecter des anomalies sur des signaux temporels assez récurrents. La toolbox 'tabata' a un module `Tubes` expérimental qui souhaite faire cela.

Dernière question bonus.

B.3.1. Proposez de vraies améliorations statistiques au tube de confiance comme on a vu dans le cours.

In [ ]:

N'oubliez pas d'enregistrer ce document et nous l'envoyer par mail. N'hésitez pas à nous contacter si vous avez besoin d'aide.