**TD Noté : Analyse de Données du Titanic**

**Objectif**

L'objectif de ce TD est d'analyser le célèbre dataset du Titanic. Vous serez évalué sur votre capacité à préparer et analyser les données, coder proprement et construire un modèle de machine learning pour prédire la survie des passagers. Vous travaillerez par groupe de 2 à 3 personnes.

**Données**

Le dataset du Titanic contient des informations sur les passagers du Titanic, y compris leur âge, sexe, classe de billet, et s'ils ont survécu ou non.

Les données de train contiennent la variable Survived qui indique si le passaer a survécu, c’est variables qu’on cherchera à prédire.

**Livrables attendu**

Sur un nouveau répertoire git :

Un notebook qui explore rapidement les données

Un fichier process.py avec les étapes de transformations du jeu de données

Un fichier train.py qui prend le dataset préparé et entraine plusieurs modèle de ML, les compare et en choisit

Des fichiers de tests associés

Un fichier README.md avec vos nom et une description du projet

Bonus : un fichier predict.py qui prend en entrée le fichier test.csv et qui prédit pour chaque passager si il survit ou pas.

**Partie 1 : Exploration des données (notebook)**

* Chargez le dataset dans un DataFrame Pandas.
* Effectuez une exploration initiale pour comprendre la structure et les caractéristiques des données.
* Bonus : créer quelques visualisations pertinentes

**Partie 2 : Nettoyage des Données (process.py)**

* Identifiez et traitez les valeurs manquantes. (avec fillna ou dropna) justifiez vos choix (pour vous aider : [4 méthodes pour gérer les données manquantes en machine learning (larevueia.fr)](https://larevueia.fr/4-methodes-pour-gerer-les-donnees-manquantes-en-machine-learning-avec-pandas/) )
* Détectez et corrigez les anomalies ou les erreurs dans les données.

**Partie 4 : Modélisation et Évaluation (train.py)**

* Construisez un modèle de classification pour prédire la survie des passagers. Vous pouvez choisir parmi des modèles tels que la régression logistique, les arbres de décision, les forêts aléatoires, etc.
* Évaluez la performance de votre modèle en utilisant la précision dans un premier temps et chercher à savoir si on peut appliquer une meilleure metrics et l’utiliser.
* Divisez votre dataset en ensembles d'entraînement et de test pour évaluer votre modèle et faites une cross validation.

(vous pouvez faire des tests sur un notebook avant d’écrire le code sur train.py, de même pour le nettoyage)

**Partie 5 : Écrire des fichiers des tests**

* Écrire les fichiers de tests
* Publier votre travail sur un repo github ouvert
* Ajouter un linter et pytest dans github actions

**Bonus :** Conteneuriser votre code dans un docker, y ajouter une BDD Postgres et y intégrer le jeux de données nettoyé