**Brief : Développez une application pour prédire le prix d’une maison**

Vous êtes chargé de développer un algorithme qui pourra prédire le prix d’une maison en fonction d’autres paramètres, comme la surface en m², le nombre de chambres, etc.



Avant de faire cette modélisation, vous devez effectuer une phase d’exploration de vos données (**EDA** = **E**xploratory **D**ata **A**nalysis). Cette phase vous permettra de mieux comprendre vos données.

Après l’EDA, vous pouvez passer à la phase suivante, la préparation de vos données pour le machine learning. (feature scaling, data transformation, etc.)

Quand vos données seront prêtes vous pourrez entraîner votre modèle. En utilisant les métriques d’évaluation comme le *R²* et le *MSE* vous pourrez faire des aller-retour entre l’entraînement et le *feature engineering* afin d’améliorer vos résultats.

Vous trouverez le jeu de données sur ce [lien](https://www.kaggle.com/datasets/harlfoxem/housesalesprediction/discussion).

Ce dataset porte sur les ventes de maisons dans le comté de King dans l'État de Washington aux États-Unis entre 2014 et 2015.

Planning :

| Jérémy | Jérémy | Antoine | Antoine | Antoine |
| --- | --- | --- | --- | --- |
| **lundi** | **mardi** | **mercredi** | **jeudi** | **vendredi** |
| cours | Analyse des données | | Modélisation | |
| Jérémy | Jérémy | Antoine | Antoine |  |
| **lundi** | **mardi** | **mercredi** | **jeudi** | **vendredi** |
| Retour sur la première partie / modélisation / application | Application / déploiement |  | Soutenance / rex | *Vacances* |

Pour ce projet vous aurez besoin de certains packages Python :

* [Pandas](https://www.youtube.com/watch?v=ZyhVh-qRZPA&list=PL-osiE80TeTsWmV9i9c58mdDCSskIFdDS) pour charger vos données
* Un package pour visualiser vos données (matplotlib, seaborn, plotly)
* [Scikit-learn](https://scikit-learn.org/stable/modules/linear_model.html) pour la partie modélisation (voir cette série de [vidéos](https://www.youtube.com/watch?v=P6kSc3qVph0&list=PLO_fdPEVlfKoHQ3Ua2NtDL4nmynQC8YiS) pour en apprendre plus sur ce package incontournable, ne pas hésiter à explorer la documentation qui est très bien)

**Partie 1 : Analyser et modéliser des données**

Pour cette partie, je vous conseille de travailler avec 3 notebooks différents :

* data\_cleaning\_analysis.ipynb
* data\_preparation.ipynb
* data\_modelisation.ipynb

Pour passer vos données d’un notebook à l’autre vous pouvez utiliser les méthodes to\_csv() et read\_csv(). Par exemple, à la fin du notebook numéro 1, vous exportez vos données à l’aide de la méthode to\_csv() et ensuite vous importez vos données au début du notebook numéro 2 avec la fonction read\_csv().

Etape 1 : Explorer et nettoyer vos données

Quelques tâches de nettoyage :

* Valeur manquante / dupliquée
* Simplifier le nom des colonnes
* Harmoniser les valeurs
* Est-ce que toutes les valeurs ont un type de donnée pertinent ? Parfois, une feature qui contient des chiffres doit-être transformée dans un autre type de données. Une question que j’aime me poser est : “Est-ce qu’il est pertinent de calculer la moyenne de cette feature ?”
* etc.

L’exploration consiste à comprendre les données sur lesquelles on travaille en répondant à ces questions :

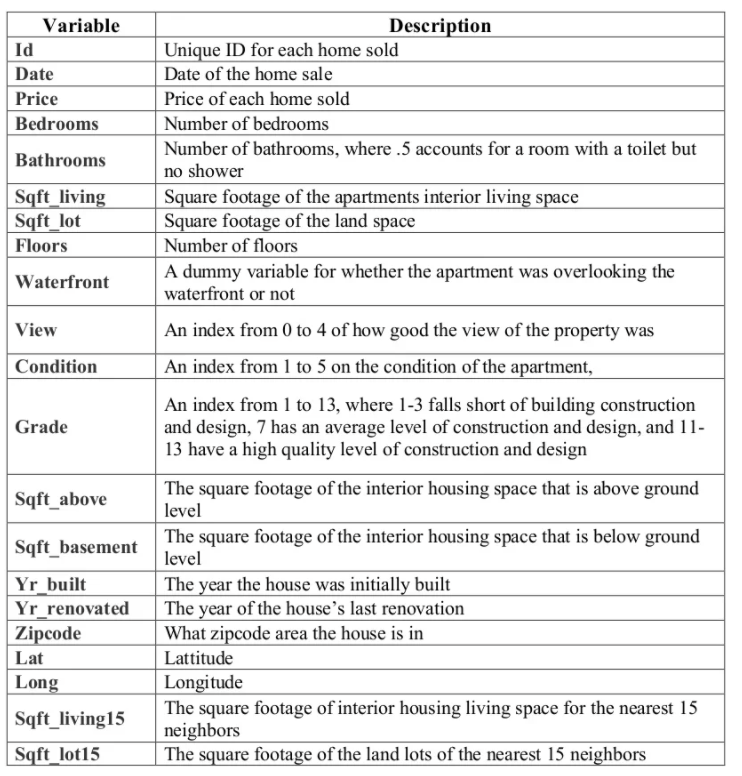
* Quelles sont les caractéristiques principales du jeu de données ?
* A quoi correspondent les features ?
* Quelles valeurs prennent chaque feature ?
* Quelles sont les corrélations entre les variables ?
* Quelles seront les features utiles pour mon modèle ?
* Etc.

Pour explorer vos données, vous pouvez utiliser des fonctions, des graphiques, des tableaux, etc.

Une bonne idée est d'utiliser la latitude et la longitude pour positionner des *data points* sur une carte. En les combinant avec une troisième dimension comme le prix, il est possible de créer une visualisation qui nous permet de bien comprendre nos données.

Afin d’en apprendre plus sur l’exploration des données, je vous invite à suivre ce [cours](https://openclassrooms.com/fr/courses/7410486-nettoyez-et-analysez-votre-jeu-de-donnees) et lire [cet article de toward data science](https://towardsdatascience.com/exploratory-data-analysis-eda-python-87178e35b14).

Pour mieux comprendre vos données, vous pouvez vous appuyer sur ces descriptions de variables :



[Pourquoi la colonne bathrooms peut prendre des décimales ?](https://www.kaggle.com/datasets/harlfoxem/housesalesprediction/discussion/24804)

Etape 2 : Préparer vos données pour la modélisation

Dans cette deuxième partie vous allez apprendre à préparer vos données pour les utiliser dans un algorithme de machine learning.

Voici quelques exemples de préparation :

* Séparer vos données en *train*, *validation*, *test sets*
* Créer des nouvelles variables
* Normaliser vos données
* Transformer vos données si nécessaire
* Etc.

[En savoir plus](https://towardsdatascience.com/5-steps-to-correctly-prep-your-data-for-your-machine-learning-model-c06c24762b73)

Etape 3 : Entraîner un algorithme de régression linéaire

Une fois que vos données sont nettoyées et préparées, vous pouvez passer aux choses sérieuses : entraîner un modèle de machine learning.

Il existe un package très populaire pour faire du machine learning avec python : scikit-learn. Vous pouvez découvrir les capacités de ce package avec cette [vidéo](https://www.youtube.com/watch?v=P6kSc3qVph0&list=PLO_fdPEVlfKoHQ3Ua2NtDL4nmynQC8YiS).

Pour cette étape, je vous conseille de vous limiter aux différentes variantes de l’algorithme de régression linéaire (ridge, lasso et elasticnet).

Malgré une sélection restreinte d'algorithmes, il y a beaucoup de marge de manœuvre, notamment avec votre pipeline de données. Je vous conseille également de creuser l’étape de [polynomial features.](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html)

Il est toujours intéressant de créer un point de comparaison avec un modèle naïf. (Voir *dummy regressor* dans la documentation de sklearn.)

Certains de ces algorithmes auront des hyperparamètres que le modèle ne peut pas optimiser par lui-même. Afin de trouver les paramètres optimaux, vous pouvez utiliser le **GridSearchCV** et le **RandomizedSeachCV.**

Pour vérifier que votre algorithme apprenne correctement, vous pouvez faire les graphiques des [courbes d’apprentissage](https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/).

Le domaine du machine learning est peuplé de métriques diverses et variées. Pour comparer et sélectionner le meilleur modèle parmi ceux que vous entraîner, je vous conseille d'utiliser le MSE (RMSE) et surtout le R² aussi appelé le coefficient de détermination. ([vidéo pour en savoir plus sur les métriques](https://www.youtube.com/watch?v=_TE9fDgtOaE&list=PLO_fdPEVlfKoHQ3Ua2NtDL4nmynQC8YiS&index=5))

Etape 4 : Analyse des résultats

Cette étape consiste à mesurer l’impact négatif que peuvent avoir certaines observations de notre jeu de données.

En vous appuyant sur ce [chapitre de cours](https://openclassrooms.com/fr/courses/4525326-realisez-des-modelisations-de-donnees-performantes/5754143-analysez-les-resultats)

**Partie 2 : Créer une application qui utilise votre modèle**

Utiliser votre modélisation pour créer une **application web qui permet de prédire le prix des maisons dans le comté de King.**

Pour cela vous utiliserez le package Streamlit qui permet de créer des applications web très rapidement. Cet outil ne permet pas de personnaliser aussi finement une application web que d’autres packages comme Django ou Flask mais il permet de créer des petites applications rapidement. C’est très pratique pour présenter votre modélisation en un temps record à un client. Ici on parle plutôt de développement Quick and Dirty.

Le design le plus simple pour votre application est de créer un formulaire où chaque champ correspond à une variable utilisée par votre modèle (comme les m² par exemple).

Concernant le fonctionnement de votre application, l’approche la plus simple est de charger votre modéle directement dans application streamlit. Une approche plus avancée est de créer une API pour votre modèle (avec Flask ou FastAPI par exemple).

Quand vous aurez une application qui fonctionne en local, il faudra déployer votre application. Pour cela je vous conseille d’utiliser Heroku (rechercher un tuto pour utiliser heroku).

Livrable :

* Un github avec les parties 1 et 2. Vous pouvez également rendre deux github distincts.
* présentation du projet

Modalités :

* Vous disposez de temps pour effectuer ce projet, profitez-en pour vous former en parrallèle grâce aux ressources fournies (et aux autres que vous trouverez)

Pas assez de ressource ?

* [Kaggle du projet](https://www.kaggle.com/harlfoxem/housesalesprediction) (Explorer vous-même le jeu de données avant de regarder les autres notebooks)
* [Machine learning](https://fr.coursera.org/learn/machine-learning) (Très bon cours sur le machine learning)

Modalités de la présentation :

* 3 minutes : Introduction du jeu de données, nettoyage effectuée
* 5 minutes : Exploration du dataset
* 10 minutes : Modélisations (feature engineering, data leakage, choix du modèle, pipeline, scores)
* 3 minutes : Démonstration de votre application
* 2 minutes : Conclusion, pistes d’amélioration, difficultés rencontrées
* 5 minutes : Questions-réponses