Installes les différentes dépendances qu’on aura besoin

Une image contenant texte, capture d’écran, Police, logiciel

Description générée automatiquement

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

On télécharge le fichier excel de kagle

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

On test le fichier avec une fonction load\_data

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

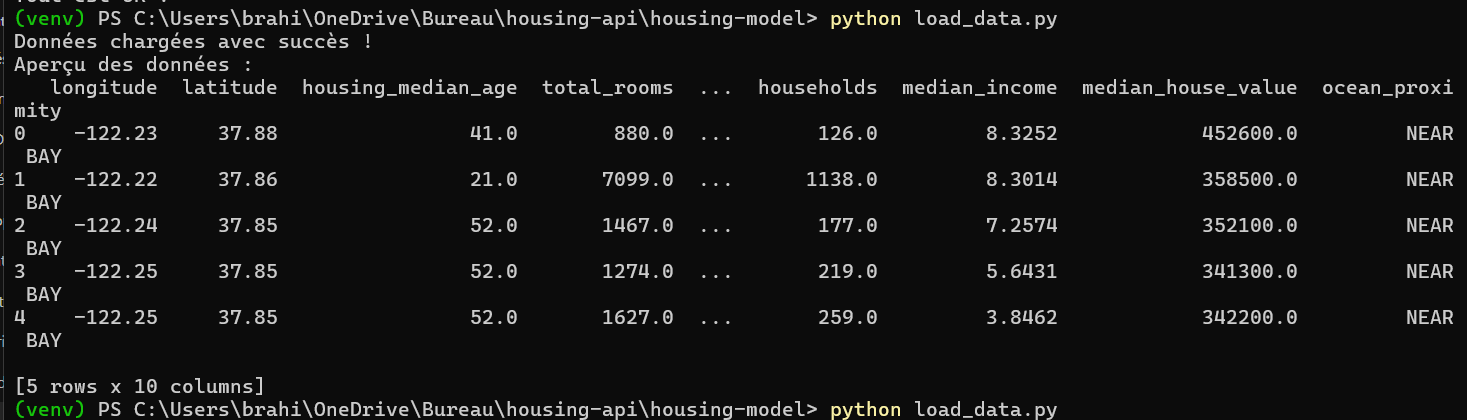
Description générée automatiquement

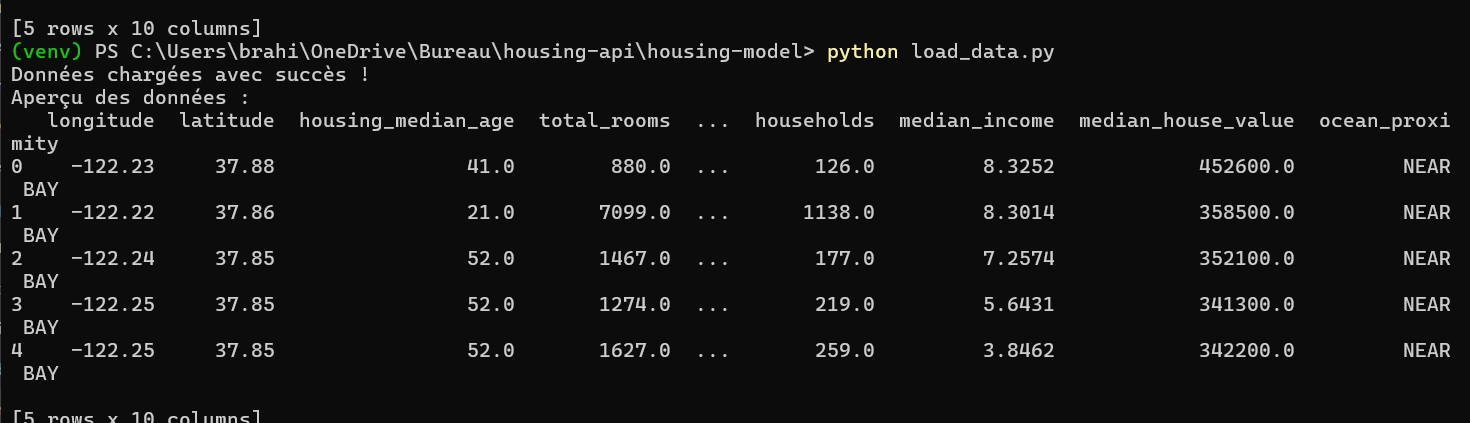
Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

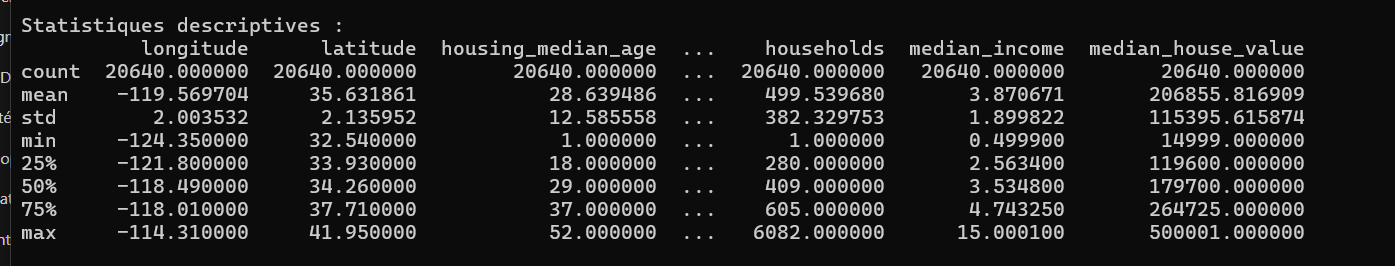
Description générée automatiquementcela fonctionne bien

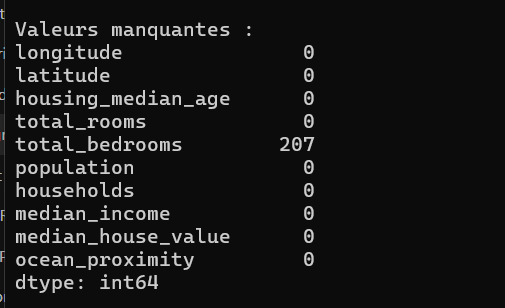
Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement









**Gérer les valeurs manquantes** :

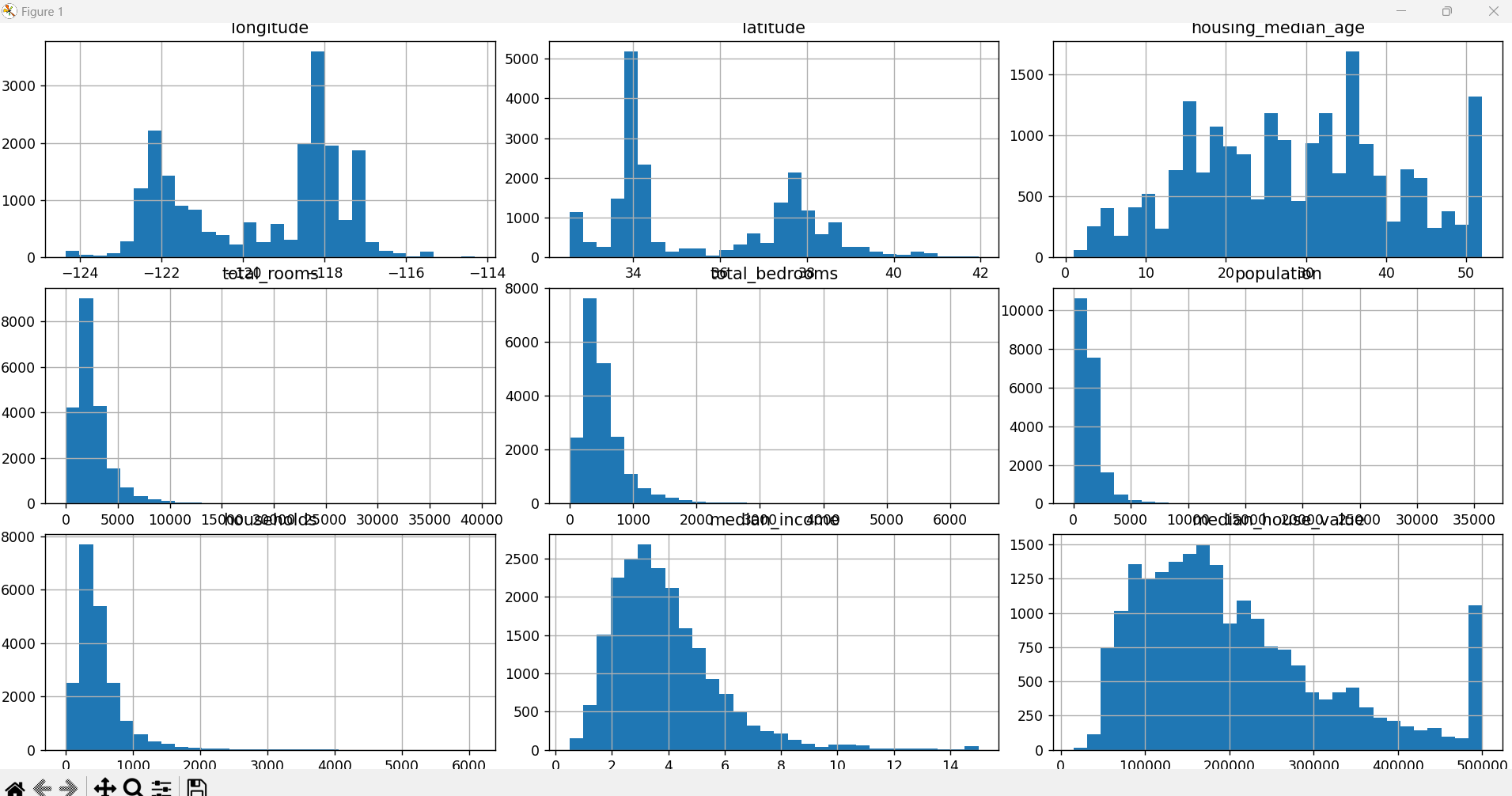
* On a identifié que la colonne total\_bedrooms contient 207 valeurs manquantes. Nous pouvons soit les remplacer par la médiane, soit les supprimer.

**Encodage des colonnes catégorielles** :

* La colonne ocean\_proximity contient des valeurs textuelles. Nous devrons les encoder pour les rendre exploitables par un modèle de machine learning.

**Normalisation des données numériques** :

* Pour garantir que toutes les caractéristiques numériques aient des échelles comparables, nous devrons les normaliser ou standardiser.



On fais un nettoyage de donnée

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

Une image contenant texte, capture d’écran

Description générée automatiquement

Un fichier housing\_cleaned est directement crée contenant les données nettoyées

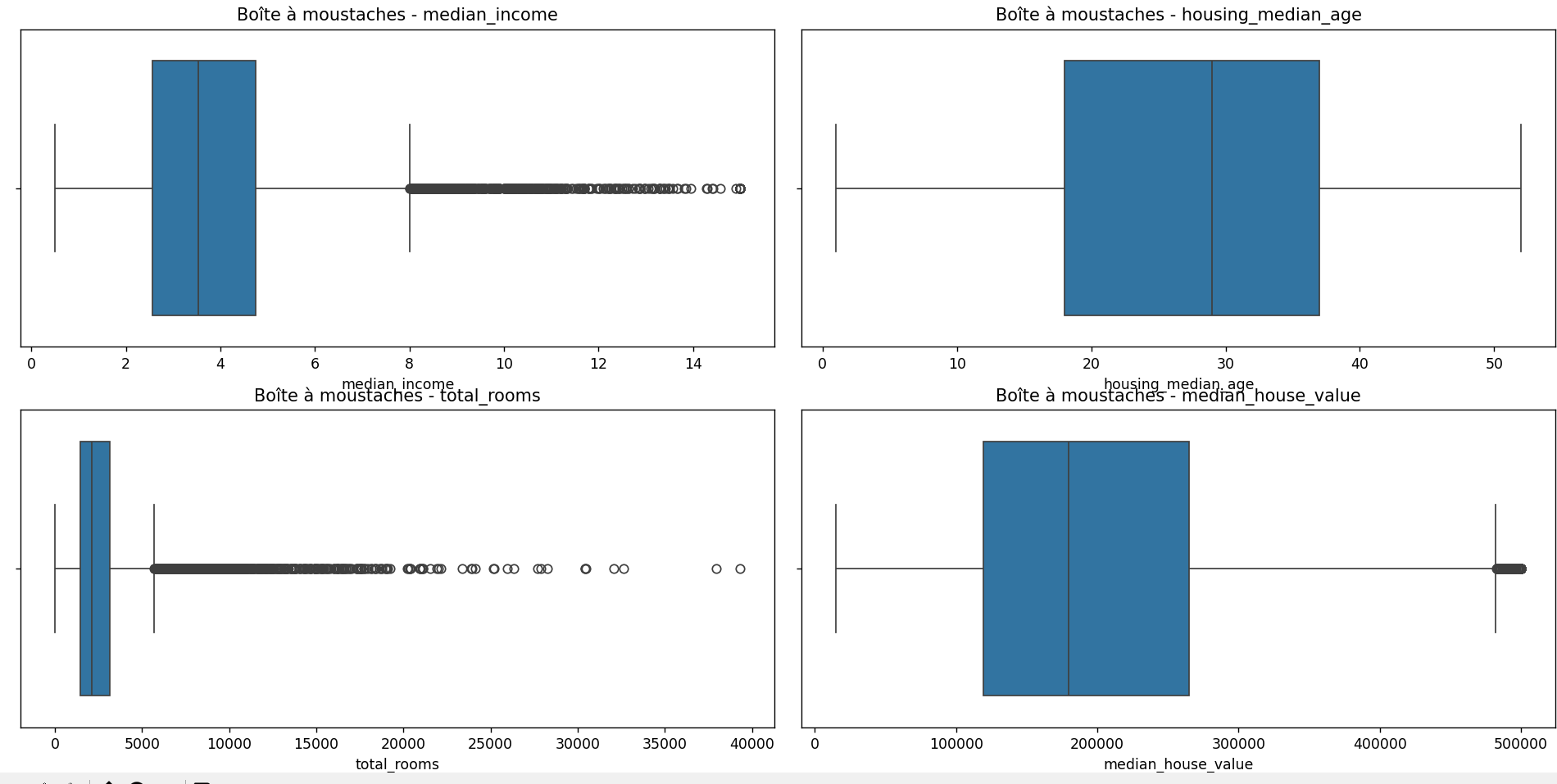
Une image contenant texte, capture d’écran, logiciel

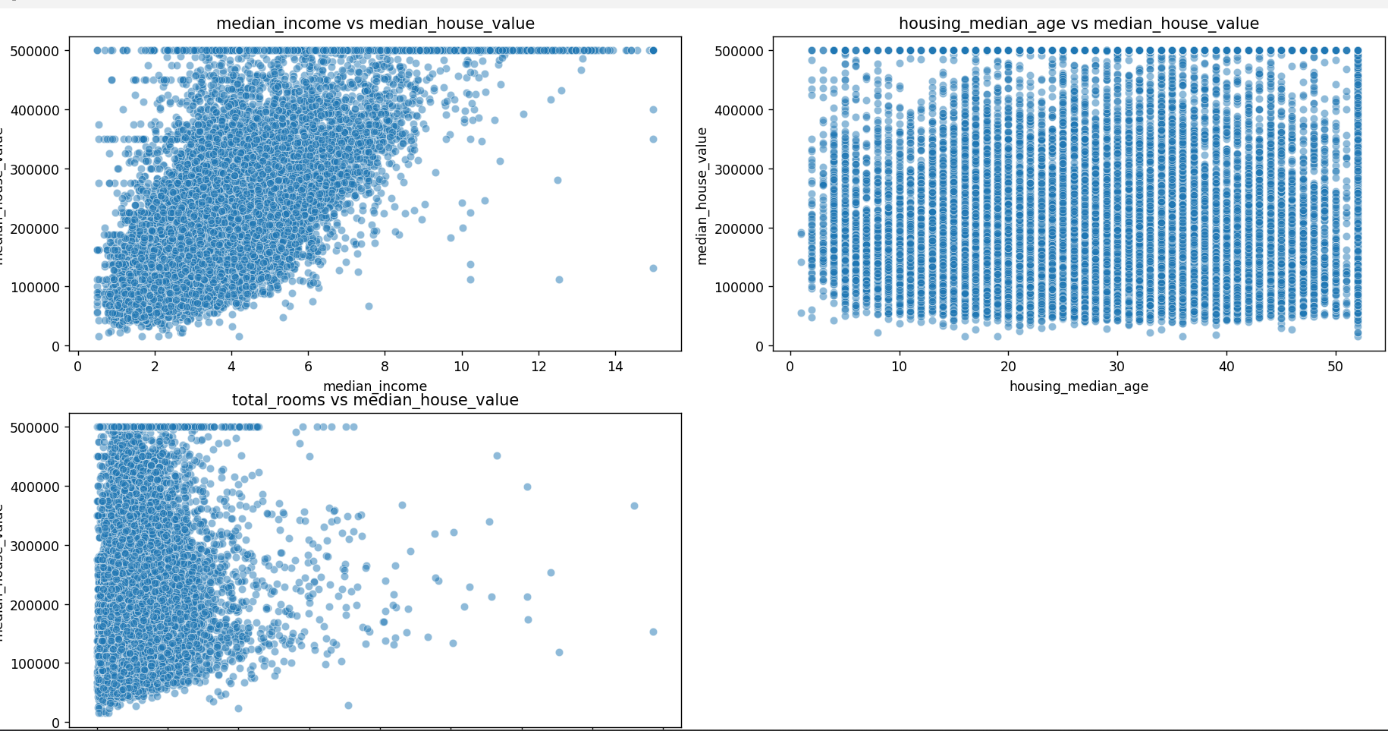
Description générée automatiquement

On explore ensuite les différentes relations

Une image contenant texte, capture d’écran

Description générée automatiquement





**median\_income vs. median\_house\_value:**

* Une relation positive est visible : lorsque le revenu médian (median\_income) augmente, la valeur médiane des maisons (median\_house\_value) augmente également.
* On observe un plafonnement à 500 000, ce qui suggère une limite supérieure dans les données.

**housing\_median\_age vs. median\_house\_value:**

* Aucune tendance claire n’apparaît, mais certaines maisons plus anciennes semblent avoir une valeur plus élevée.

**total\_rooms vs. median\_house\_value:**

* Une légère dispersion est visible, mais il n’y a pas de relation évidente ou linéaire.

**On passe ensuite à l’étape de préparation des données**

Pour cette étape, voici les actions que nous allons réaliser :

1. **Créer les variables indépendantes (X) et dépendantes (y) :**
   * X contiendra toutes les colonnes explicatives (sauf median\_house\_value).
   * y contiendra uniquement la variable cible (median\_house\_value).
2. **Encoder la colonne catégorielle ocean\_proximity :**
   * Cette colonne contient des valeurs textuelles qu'il faudra transformer en données numériques.
3. **Séparer les données en ensembles d'entraînement et de test :**
   * Diviser les données en 80 % pour l'entraînement et 20 % pour le test.
4. **Normaliser les données :**
   * Standardiser les colonnes numériques pour améliorer la performance des modèles.

Une image contenant texte, capture d’écran

Description générée automatiquement

**Séparation des données :**

* Elle divise les données en variables indépendantes (features) X et la variable cible (target) y, ici median\_house\_value.

**Encodage des colonnes catégoriques :**

* La colonne ocean\_proximity, qui contient des chaînes de caractères (par exemple, "NEAR BAY", "INLAND", etc.), est transformée en valeurs numériques grâce à un encodage One-Hot.

**Standardisation des colonnes numériques :**

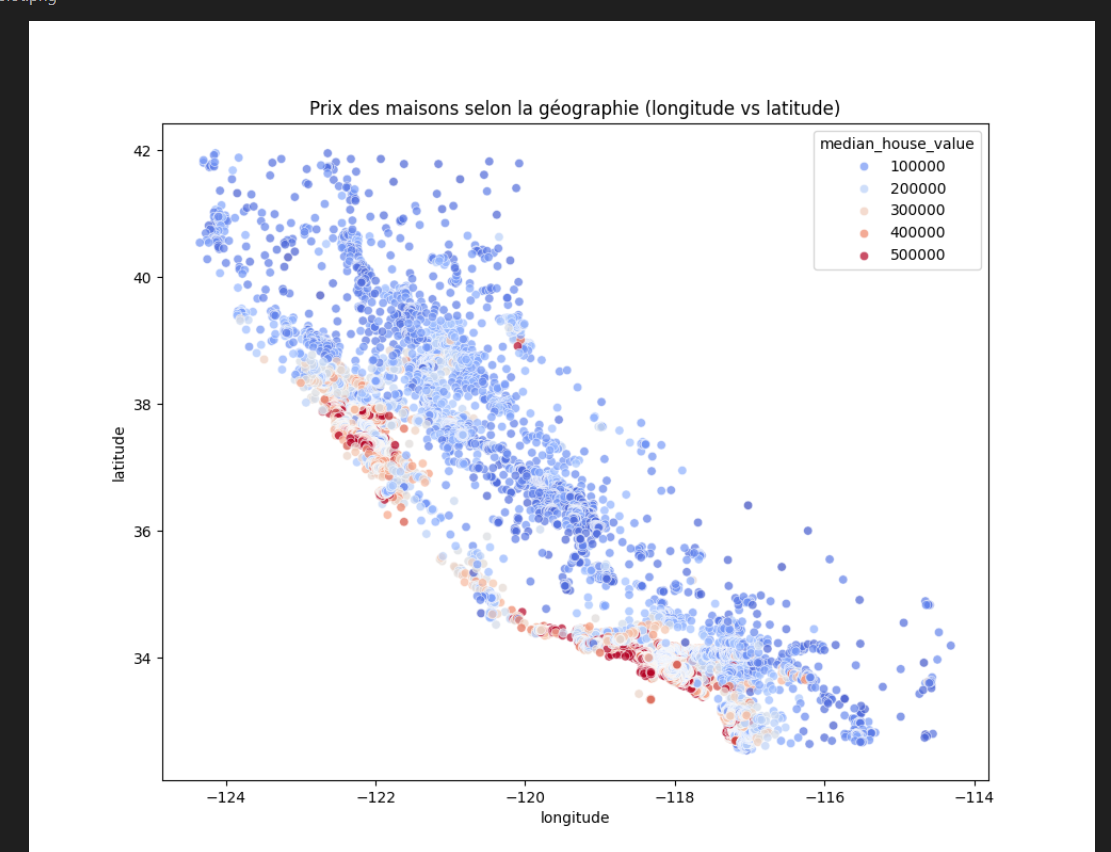
* Les colonnes numériques comme median\_income ou total\_rooms sont mises à l'échelle (standardisées) pour que leurs valeurs aient une moyenne de 0 et un écart-type de 1. Cela permet aux modèles de converger plus rapidement et d'éviter que les colonnes avec des valeurs élevées dominent les autres.

**Division en ensemble d'entraînement et de test :**

* Les données sont divisées en deux parties : 80 % pour entraîner le modèle (X\_train, y\_train) et 20 % pour tester sa performance (X\_test, y\_test).

Une image contenant texte, capture d’écran, Police

Description générée automatiquement



J’ai fais en sorte que les graphiques générés automatiquement sont stockées sous format png directement pour éviter de bloquer d’autres scripts lié à la partie entrainement modèle ML

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

* Visualisation géographique pour examiner la distribution des prix des maisons en fonction de la latitude et de la longitude.

**Données nettoyées :**

* Les valeurs manquantes dans total\_bedrooms ont été remplacées par la médiane.
* Les données nettoyées ont été enregistrées dans le fichier housing\_cleaned.csv.

**Préparation des données pour le modèle :**

* Les données ont été divisées en :
  + **Jeu d'entraînement :** 16,512 exemples (80 % des données).
  + **Jeu de test :** 4,128 exemples (20 % des données).
* Les colonnes numériques ont été standardisées, et la variable catégorielle ocean\_proximity a été encodée.

**On passe maintenant à l’étape d’entrainement de modèle ML**

**Choisir un modèle :**

* Nous utiliserons **Régression Linéaire** comme premier modèle pour sa simplicité et son efficacité pour ce type de données.

**Entraîner le modèle :**

* Entraîner le modèle sur le jeu de données d'entraînement.

**Évaluer la performance :**

* Calculer des métriques comme le **Mean Squared Error (MSE)** et le **R²** sur les données de test.

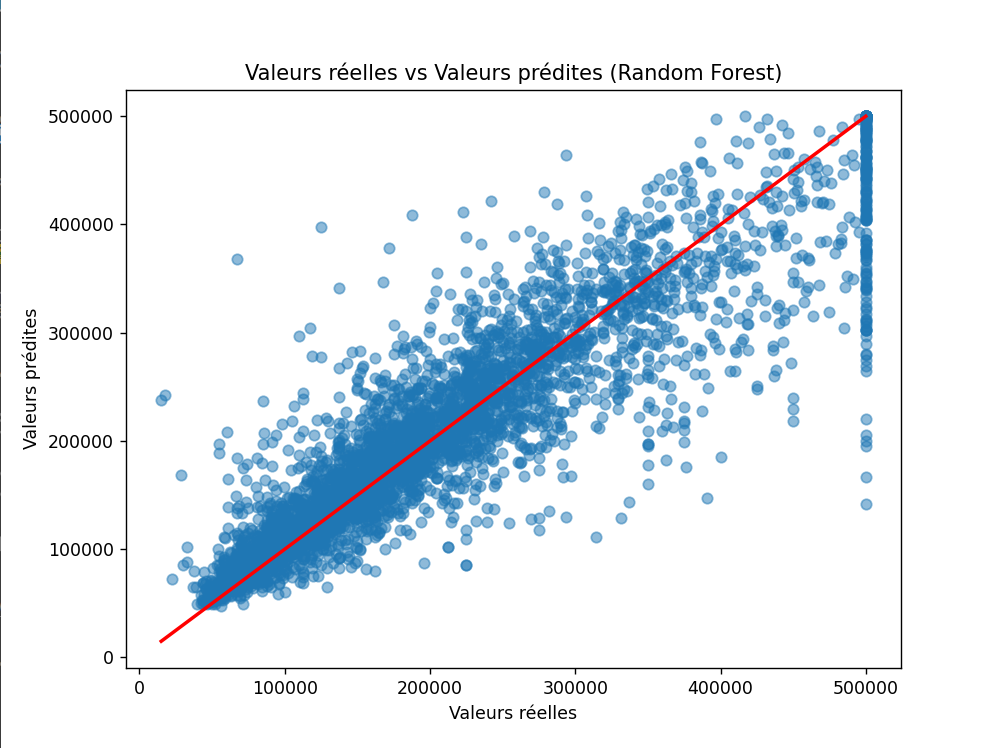
**Visualiser les résultats :**

* Comparer les valeurs prédites avec les valeurs réelles.

Je décide par la suite de tester différent modèle pour voir lequel serait le plus adapter

Une image contenant texte, capture d’écran, Police

Description générée automatiquement



**Axes** :

* **Axe horizontal (x)** : Les **valeurs réelles** (median\_house\_value) tirées de votre jeu de test.
* **Axe vertical (y)** : Les **valeurs prédites** par le modèle Random Forest.

**Points bleus** :

* Chaque point représente une prédiction du modèle pour une maison donnée.
* Un point proche de la diagonale rouge signifie que la prédiction est très proche de la valeur réelle.

**Ligne rouge (diagonale)** :

* C'est la ligne où les valeurs prédites sont **exactement égales** aux valeurs réelles.
* Plus les points bleus sont proches de cette ligne, plus le modèle est précis.

**Distribution des points** :

* La plupart des points sont bien alignés autour de la diagonale rouge, ce qui indique que les prédictions du modèle sont cohérentes avec les valeurs réelles.
* Cependant, il y a quelques écarts (points plus éloignés), ce qui est normal dans une tâche de régression avec des données réelles.

**Prédictions pour les valeurs maximales** :

* On remarque un **effet de saturation** pour les valeurs proches de 500 000 (maximum du dataset). Cela peut s'expliquer par la présence de valeurs limites dans les données ou une limite dans la capacité du modèle à extrapoler.

**Performance globale** :

* Avec un **R² de 0.82**, le modèle capture une bonne partie de la variance des données. Cela correspond à la densité de points proches de la diagonale.

On passe maintenant à l’implémentation de l’API

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

Donc après avoir sauvegardé le meilleur modèle «  Random forest » dans un fichier best\_model, on s’attaque à l’api

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Une image contenant capture d’écran, texte, logiciel, Icône d’ordinateur

Description générée automatiquement

L’API fonctionne bien ici

On passe à l’etape test de l’API

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Cela fonctionne bien

**Contenu :** { "prediction": 430457.35 } : L'API a renvoyé une prédiction, ici 430457.35. Cela correspond à la valeur estimée du prix médian de la maison en fonction des caractéristiques fournies. «  430 457 dollars »

**On passe à l’étape finale du projet : trouver un moyen de rendre portable l’API « dockerizer »**

Je construis l’image

**Une image contenant texte, capture d’écran, Police

Description générée automatiquement**

Une image contenant texte, capture d’écran, logiciel, Police

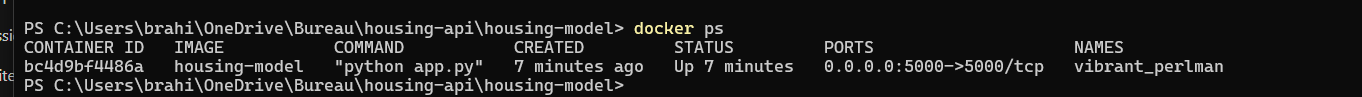
Description générée automatiquement

On remarque ici que l’API est en cours d'exécution dans un conteneur Docker et répond correctement au point de vérification

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

On a bien une prédiction obtenue et Docker a bien servi l'application Flask. L'API est accessible via le conteneur.



On a bien l’api