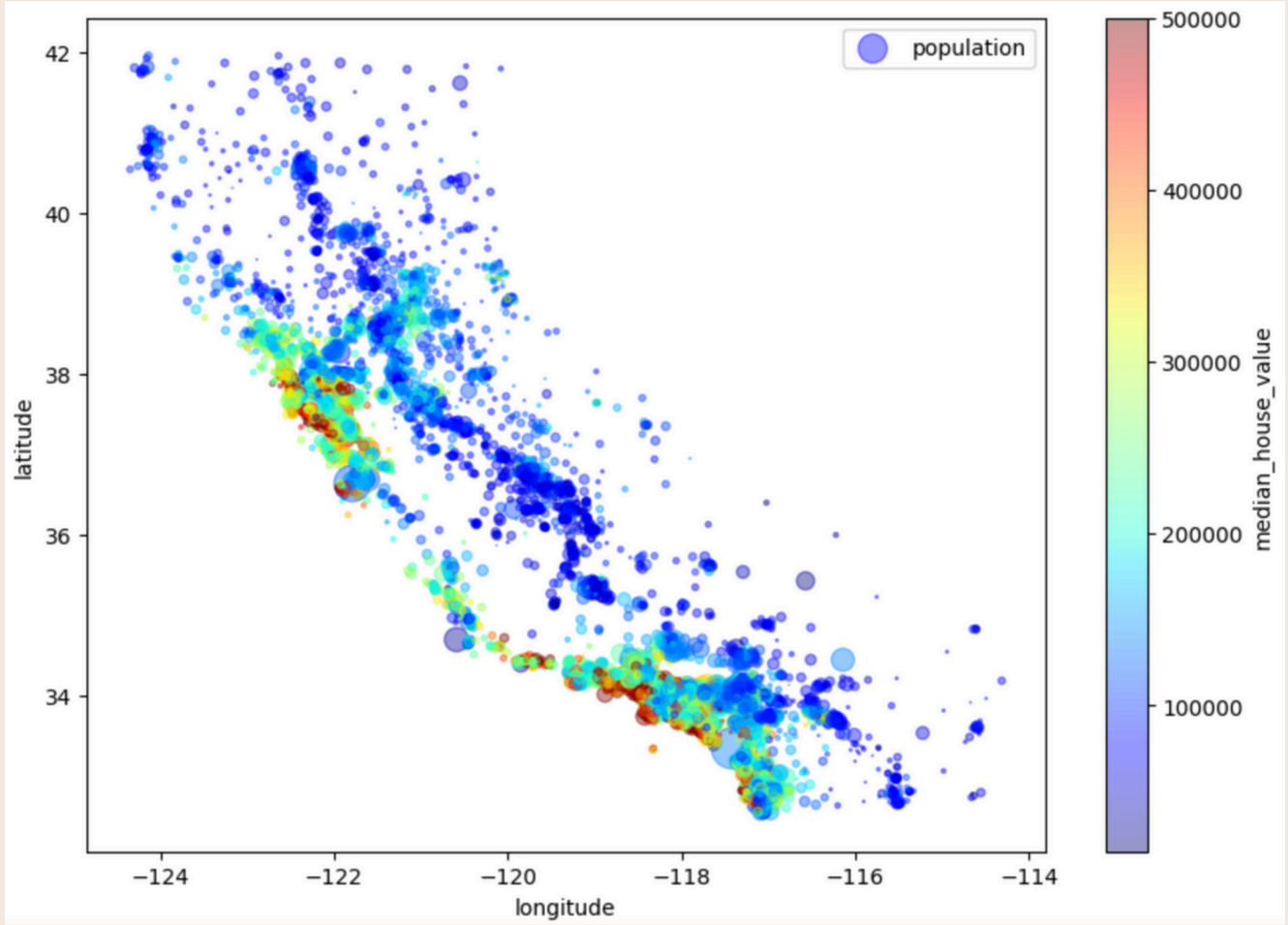
Project Immo

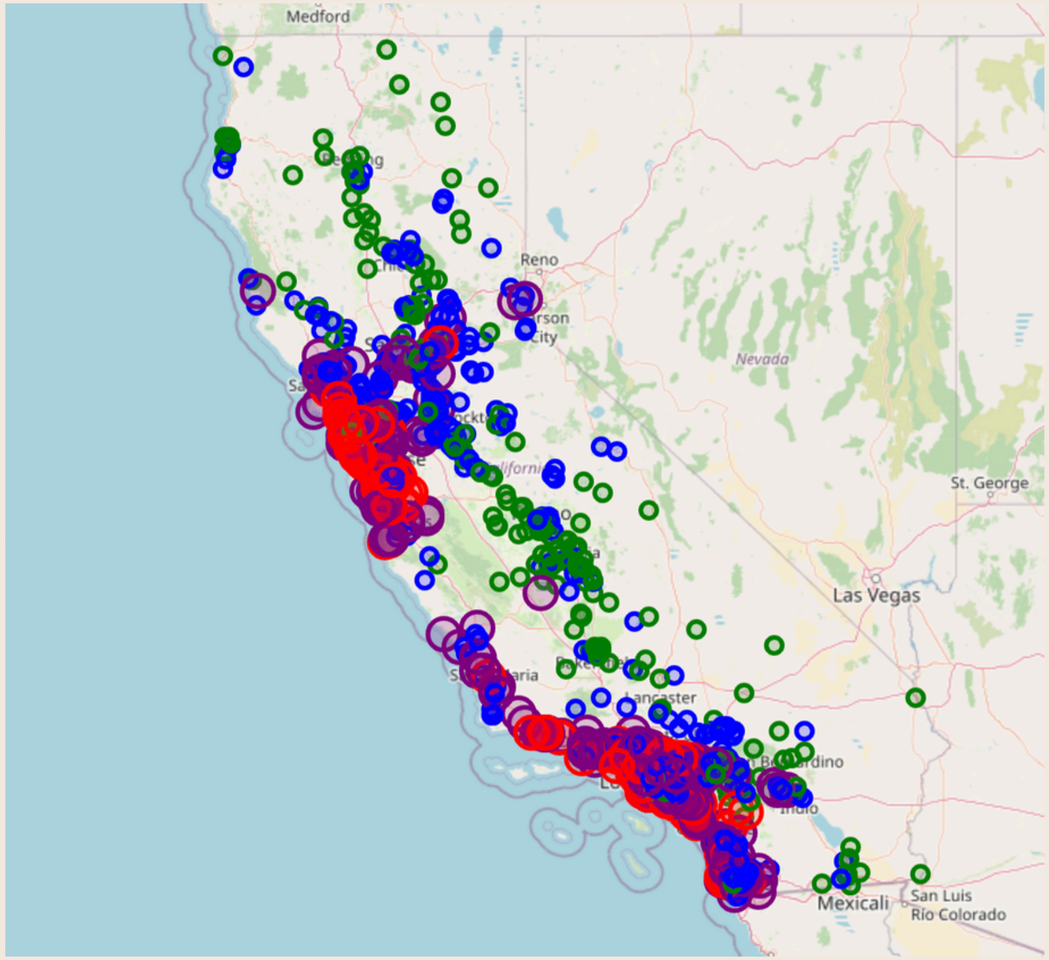
GUILLAUME, QUENTIN

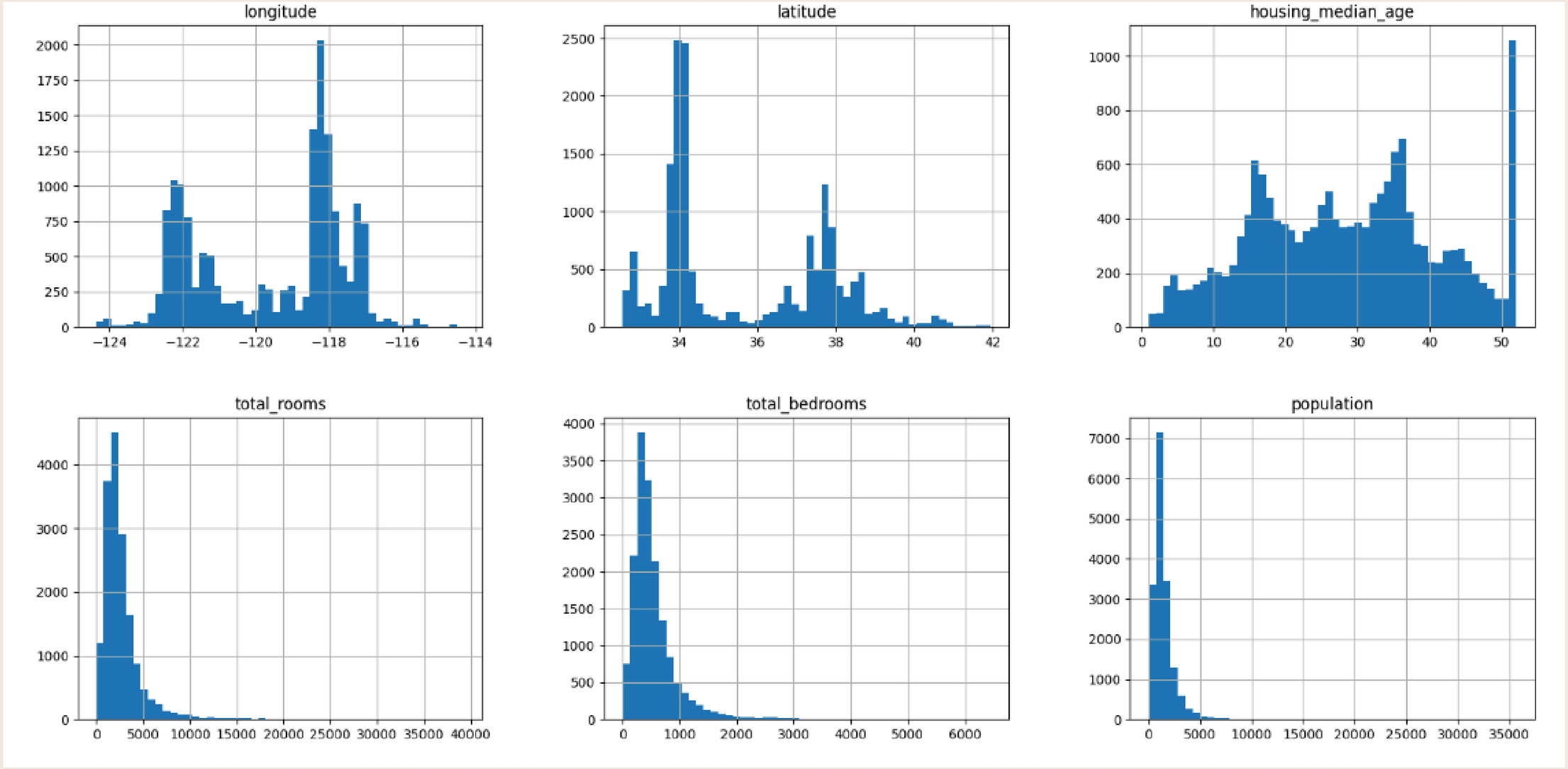
1. LE DATASET

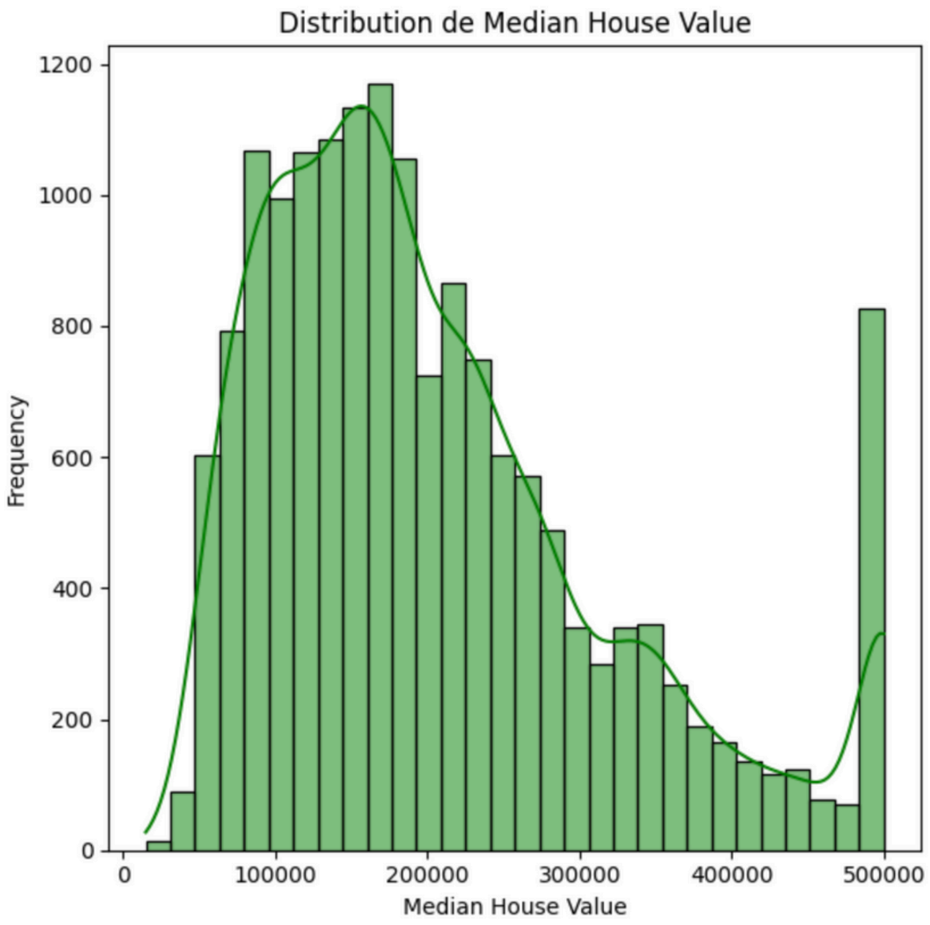
	Unnamed: 0	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	2072	-119.84	36.77	6.0	1853.0	473.0	1397.0	417.0	1.4817	72000.0	INLAND
1	10600	-117.80	33.68	8.0	2032.0	349.0	862.0	340.0	6.9133	274100.0	<1H OCEAN
2	2494	-120.19	36.60	25.0	875.0	214.0	931.0	214.0	1.5536	58300.0	INLAND
3	4284	-118.32	34.10	31.0	622.0	229.0	597.0	227.0	1.5284	200000.0	<1H OCEAN
4	16541	-121.23	37.79	21.0	1922.0	373.0	1130.0	372.0	4.0815	117900.0	INLAND

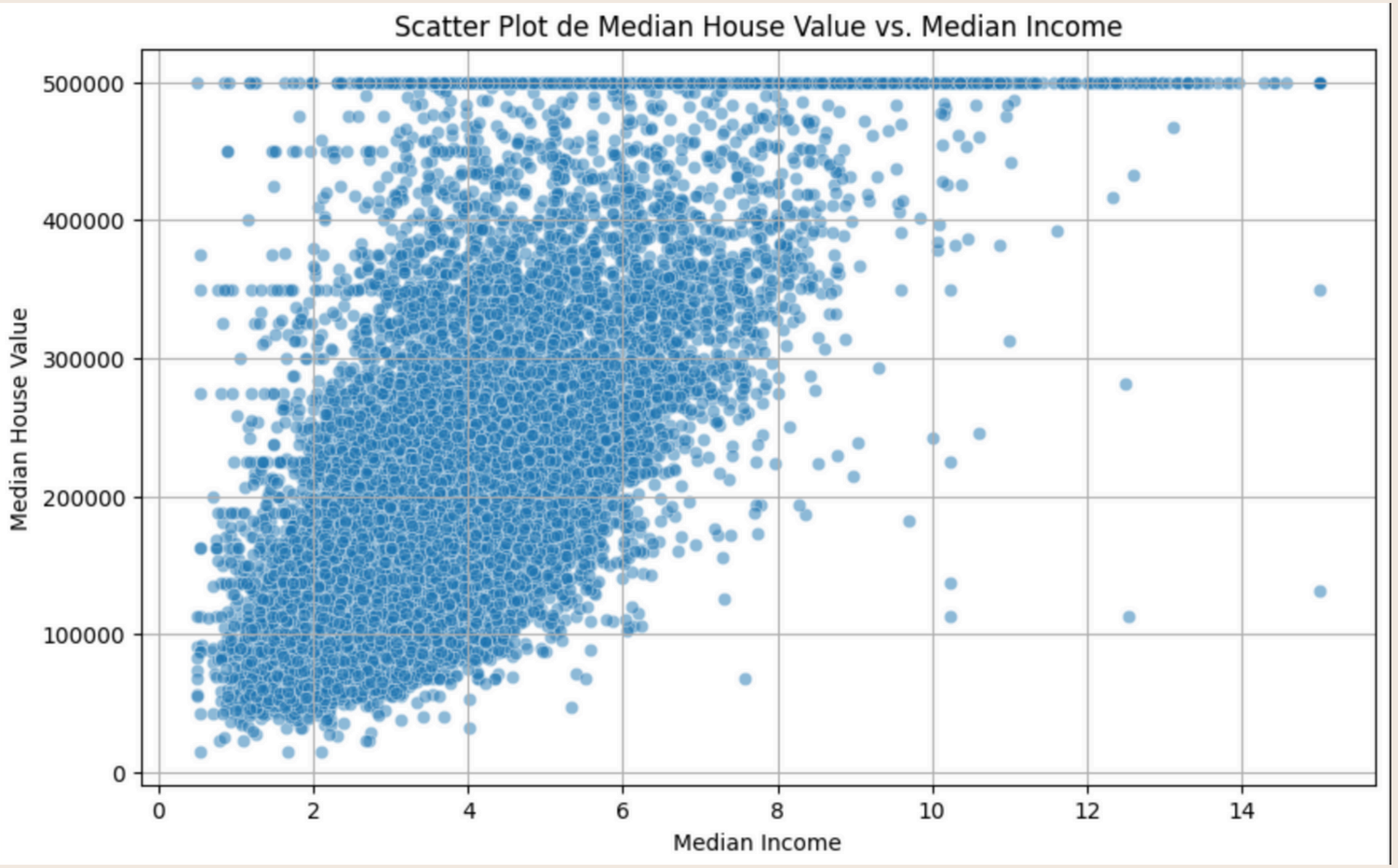
2. ANALYSE DES DONNÉES

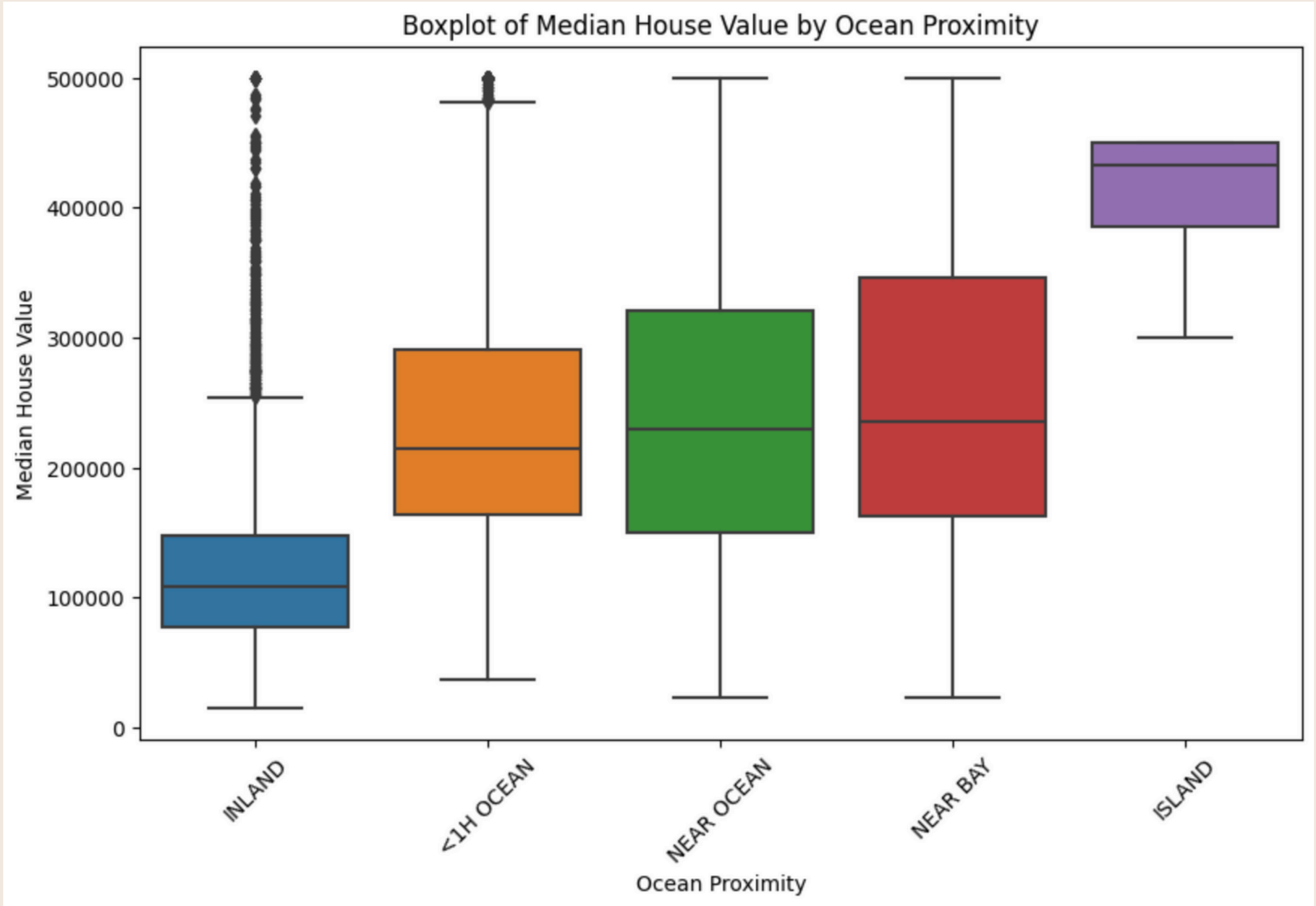


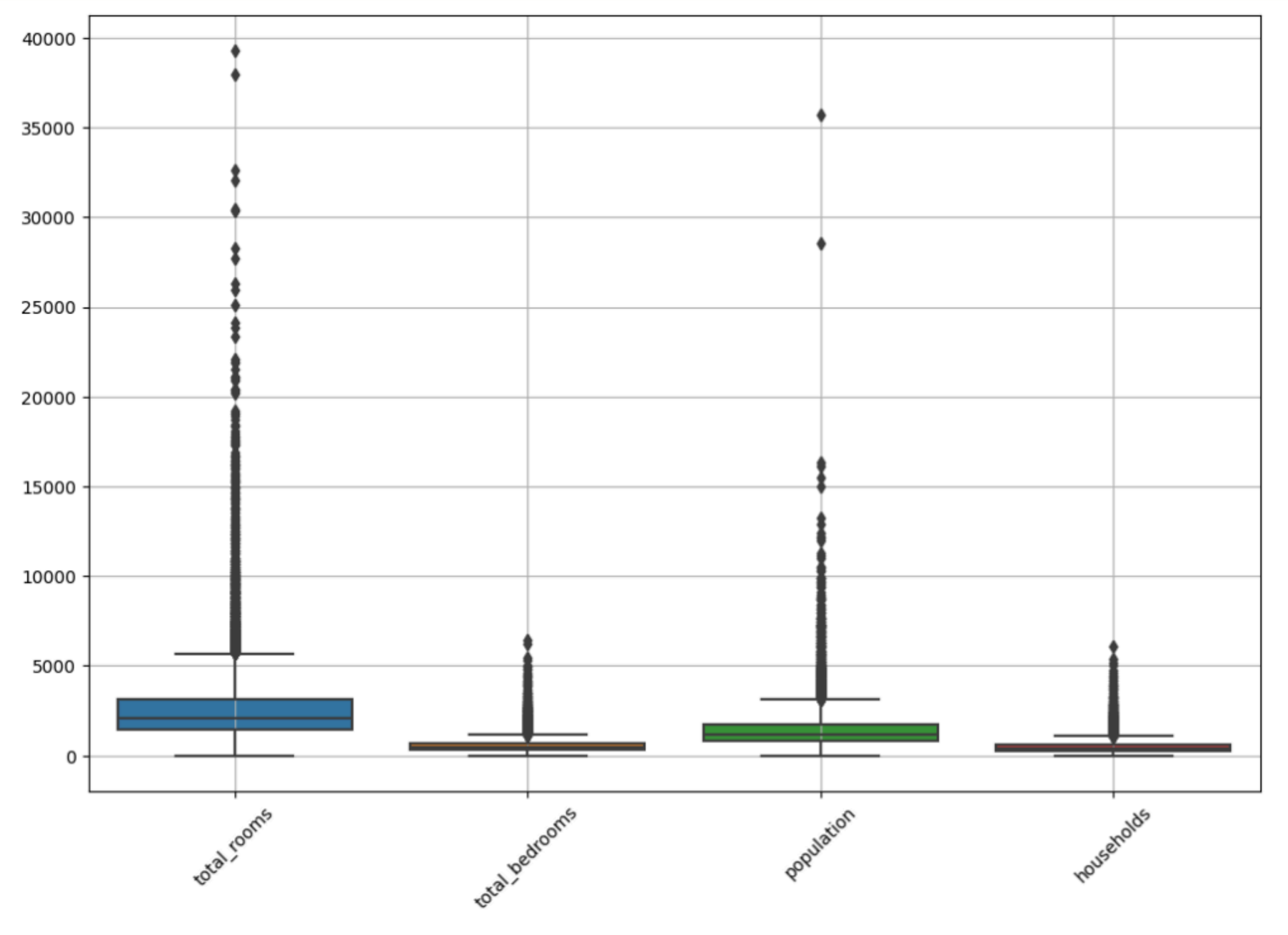



longitude
latitude
housing_median_age
total_rooms
total_bedrooms
population


Distribution de Median House Value
Frequency
Median House Value
0
200
400
600
800
1000
1200
0
100000
200000
300000
400000
500000


Scatter Plot de Median House Value vs. Median Income
Median House Value
Median Income
0
100000
200000
300000
400000
500000
0
2
4
6
8
10
12
14


Boxplot of Median House Value by Ocean Proximity
Median House Value
500000
400000
300000
200000
100000
0
INLAND
<1H OCEAN
NEAR OCEAN
NEAR BAY
ISLAND
Ocean Proximity


40000
35000
30000
25000
20000
15000
10000
5000
0
total_rooms
total_bedrooms
population
households

3. NETTOYAGE DES DONNÉES

- Suppression de la colonne Unnamed: 0
- Standardisation des données
- Remplacement des NaN par la median
- Convertit ocean_proximity en variables binaires 0 et 1

4. MODÈLE

KNN

```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.neighbors import KNeighborsRegressor
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.metrics import r2_score

# Suppression de la colonne 'Unnamed: 0'
if 'Unnamed: 0' in final_data.columns:
    final_data = final_data.drop('Unnamed: 0', axis=1)

# Préparation des données
X = final_data.drop('median_house_value', axis=1)
y = final_data['median_house_value']

# Séparation des données
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=200)

# Identification des types de caractéristiques
numeric_features = X_train.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_features = X_train.select_dtypes(include=['object']).columns.tolist()

# Transformateurs pour les caractéristiques numériques
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

# Transformateurs pour les caractéristiques catégorielles
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])
```

```python
# Préprocesseur avec ColumnTransformer
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# Pipeline complet
pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('knn', KNeighborsRegressor(n_neighbors=5))
])

# Configuration des paramètres pour le GridSearchCV
param_grid = {
    'knn__n_neighbors': [3, 5, 7, 10],
    'knn__weights': ['uniform', 'distance']
}

# Création de l'objet GridSearchCV
grid_search = GridSearchCV(pipeline, param_grid, cv=5, scoring='r2', verbose=1)

# Entraînement du GridSearchCV
grid_search.fit(X_train, y_train)

# Meilleurs paramètres trouvés
print("Meilleurs paramètres: ", grid_search.best_params_)

# Meilleur score R^2 obtenu
print("Meilleur score R^2: ", grid_search.best_score_)

# Prédiction avec le meilleur modèle trouvé
y_pred = grid_search.predict(X_test)

# Calcul de R^2
r_squared = r2_score(y_test, y_pred)
print("R^2 score avec KNN :", r_squared)
```

```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
Meilleurs paramètres:  {'knn__n_neighbors': 10, 'knn__weights': 'distance'}
Meilleur score R^2:  0.7170471343985434
R^2 score avec KNN : 0.7347990403384093
```

5. WEBAPP

Prédictions de valeurs immobilières

Cette application prédit les valeurs immobilières basées sur les entrées utilisateur.

Choisissez un fichier CSV pour évaluation

Drag and drop file here

Limit 200MB per file • CSV

Browse files

housing-train-data-6628a4723213d886993351.csv 1.2MB

Le score R^2 pour les données fournies est: 0.92

Total Rooms

5000

Housing Median Age

35

Median Income

3,00

Total Bedrooms

1000

Ocean Proximity

<1H OCEAN

Population

3000

Households

500

Latitude

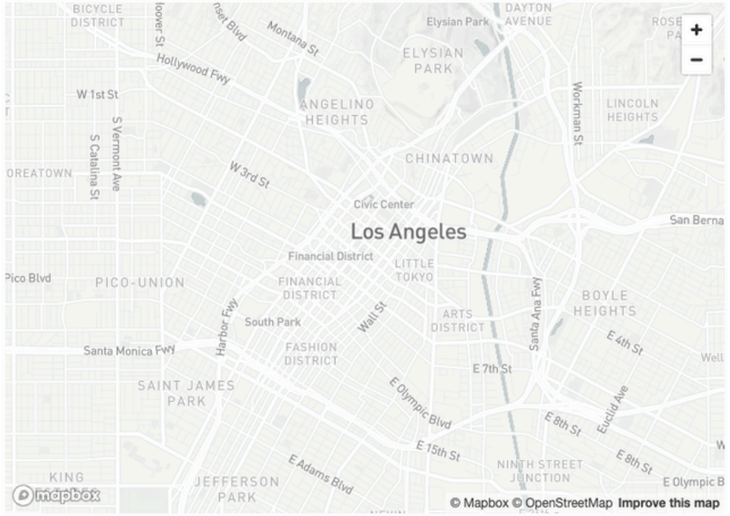
34.0500

32.0000 42.0000

Longitude

-118.2500

-124.0000 -114.0000



Prédire

THANK YOU SO MUCH!

thank you so much