

# PROJET PREDICTION IMMOBILIÈRE

Ryan / Anne

01 - Introduction

---

02 - Data Visualization

---

03 - Pipeline

---

04 - Demo

# 01 - Introduction

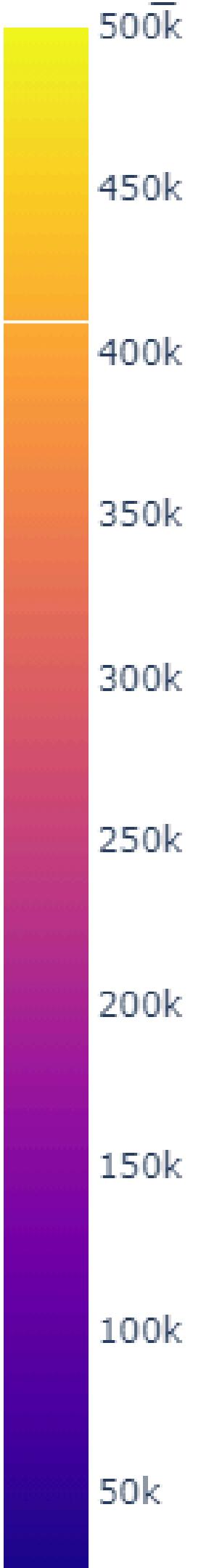
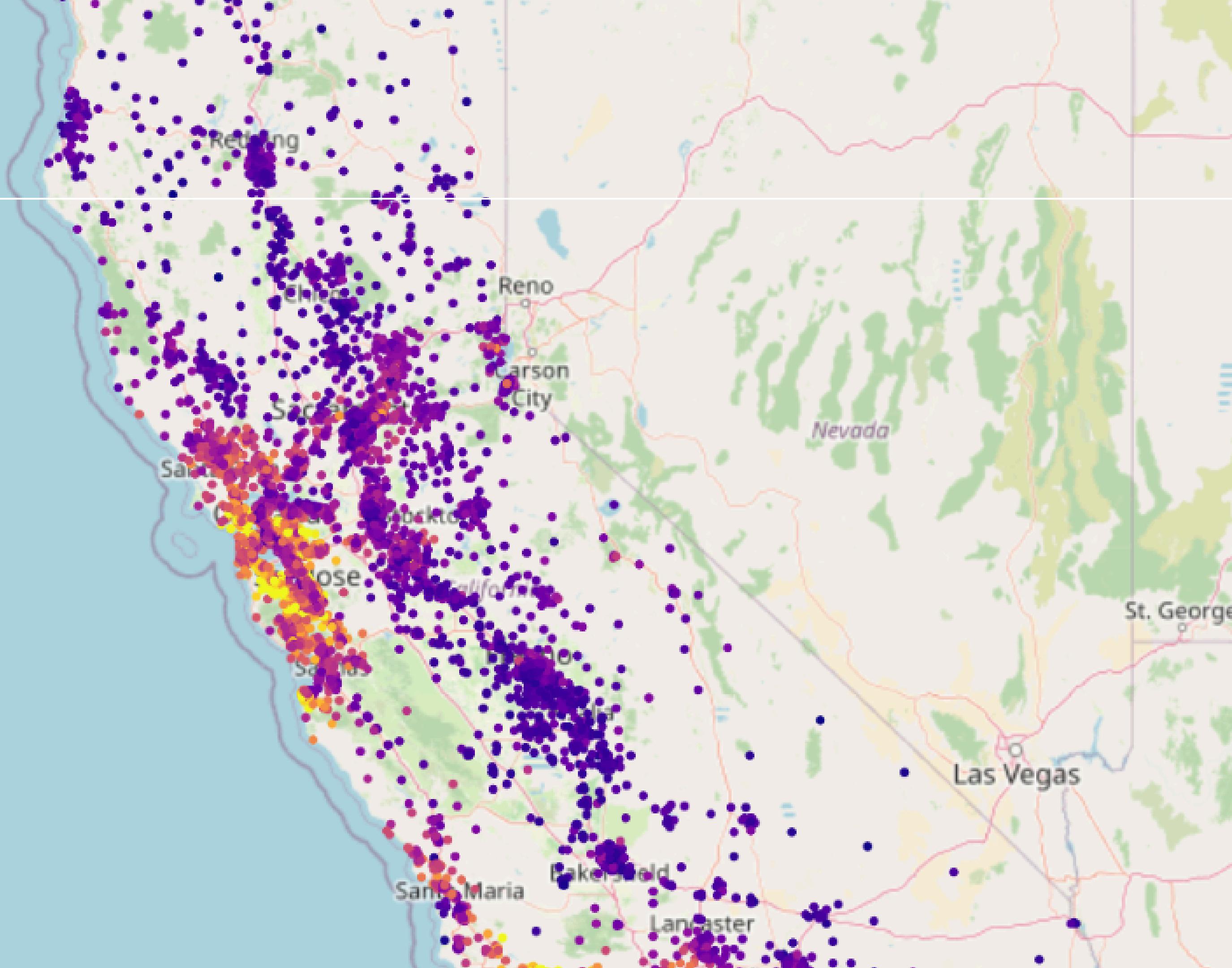
Dans le cadre d'une WebApp, notre objectif était de tester un modèle et de faire le choix pour notre prototype . Cette dernière à pour but de prédire le prix d'un bien en fonction de sa localisation.Pour ce faire nous avions une fichier de 9 colonnes et 16572

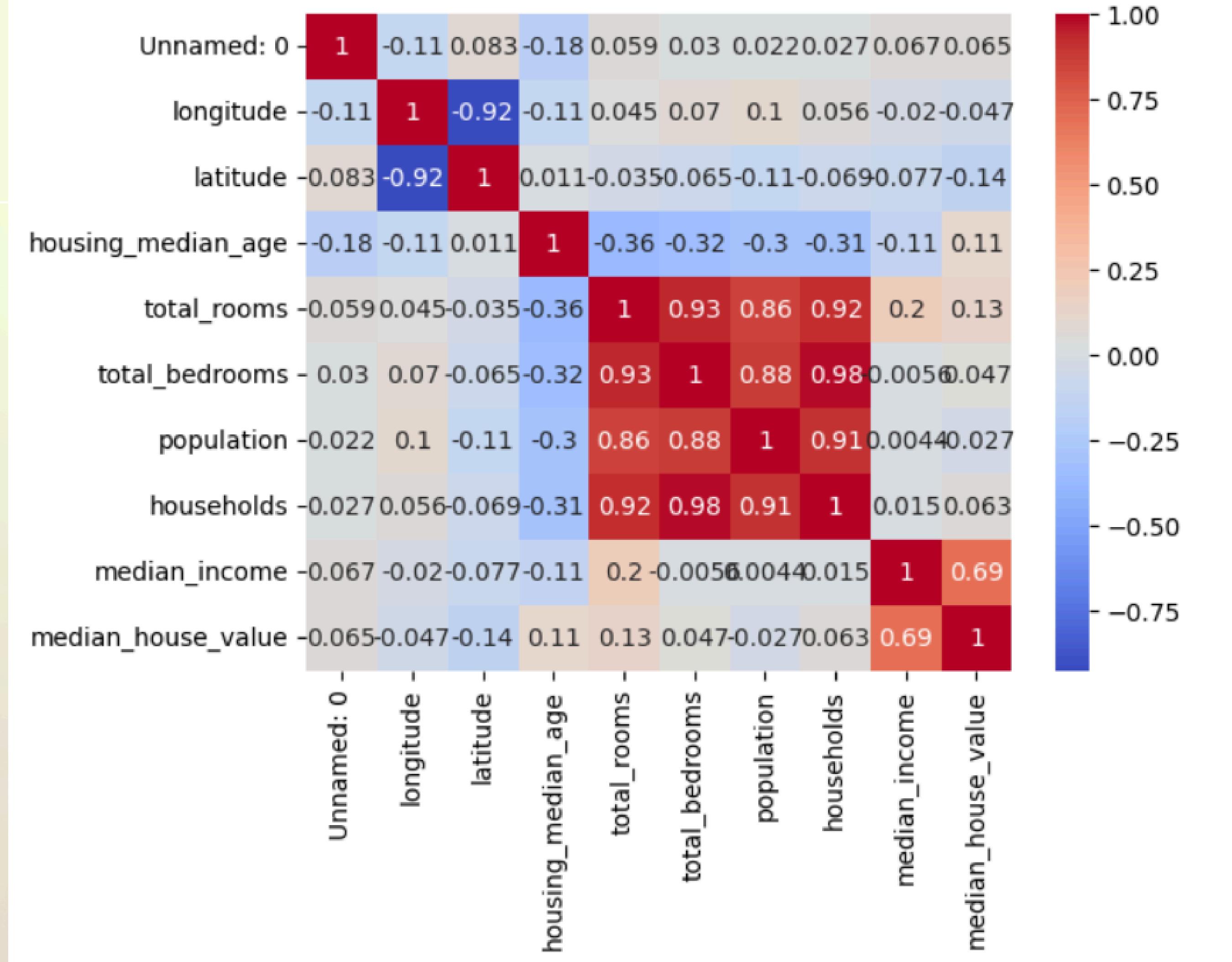
# 02 - data shape

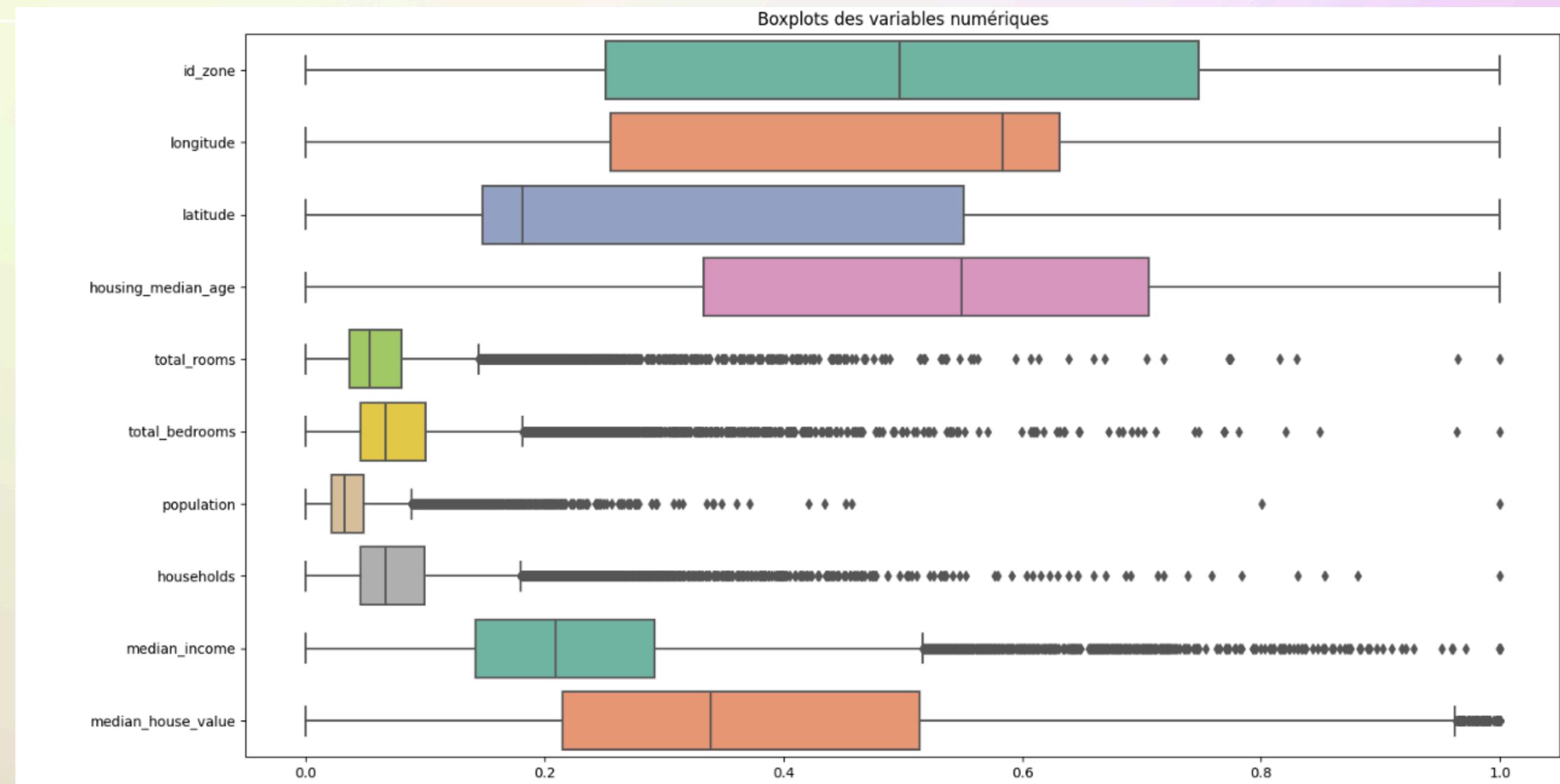
`df.info()`  
`df.describe()`  
`df.shape()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16512 entries, 0 to 16511
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
 ---  --  
 0   Unnamed: 0        16512 non-null   int64  
 1   longitude         16512 non-null   float64 
 2   latitude          16512 non-null   float64 
 3   housing_median_age 16512 non-null   float64 
 4   total_rooms        16512 non-null   float64 
 5   total_bedrooms     16336 non-null   float64 
 6   population         16512 non-null   float64 
 7   households         16512 non-null   float64 
 8   median_income      16512 non-null   float64 
 9   median_house_value 16512 non-null   float64 
 10  ocean_proximity    16512 non-null   object  
dtypes: float64(9), int64(1), object(1)
memory usage: 1.4+ MB
```

# 03 - Data Visualization







# Preprocessing

- Remplacer les valeurs manquantes par leur moyenne
- Avec le `MinMaxscaler()` on a standardisé nos données.
- Pour prévenir les valeur manquante on a Remplacer les valeur manquante par les plus fréquentes
- On à encoder avec la méthode `OneHot`

## Pipeline

col\_trans: ColumnTransformer

▶ num\_pipeline ▶

▼ SimpleImputer

SimpleImputer()

cat\_pipeline

▼ SimpleImputer

SimpleImputer(strategy='most\_frequent')

▼ MinMaxScaler

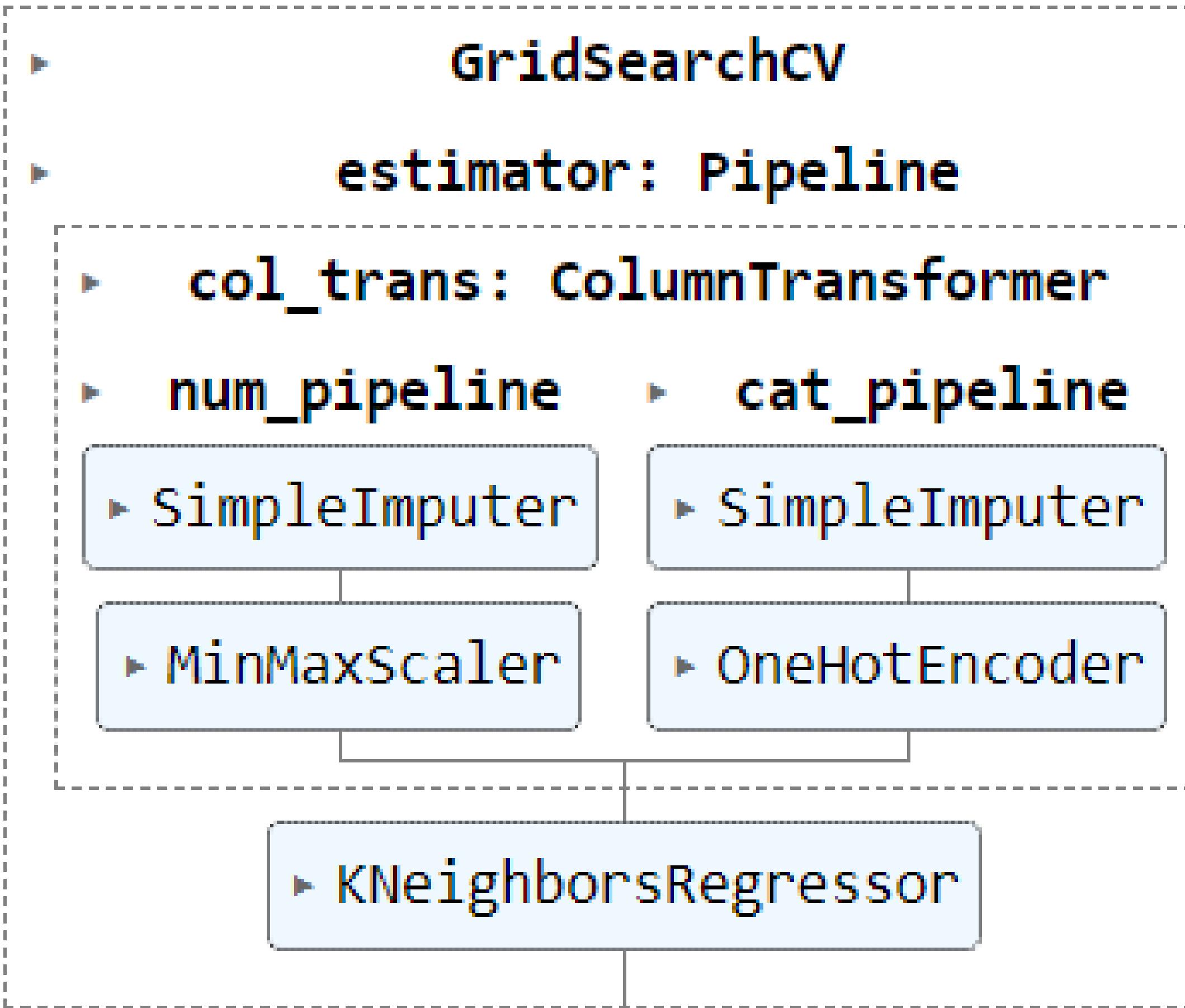
MinMaxScaler()

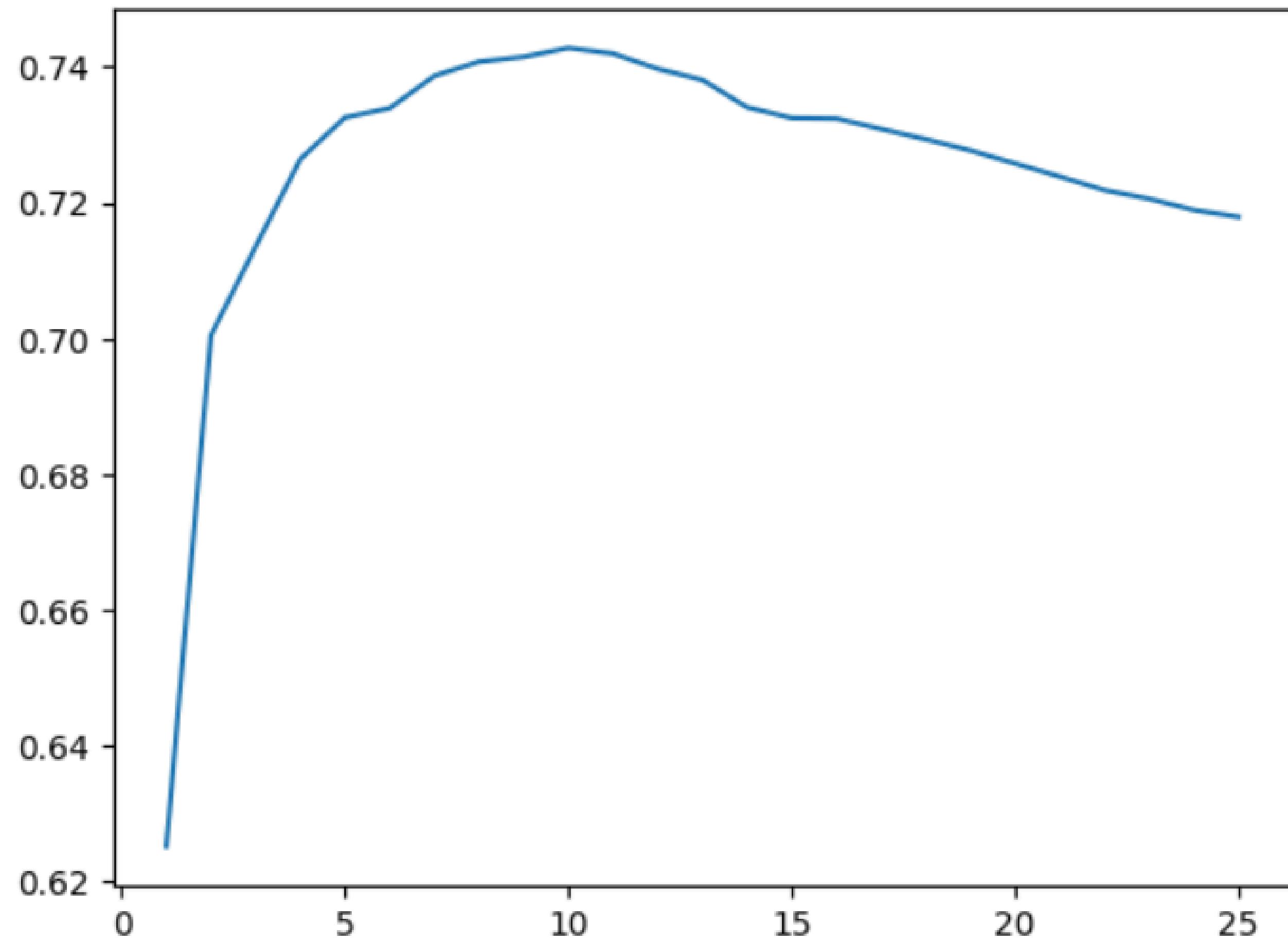
OneHotEncoder

OneHotEncoder(handle\_unknown='ignore', sparse=False)

▼ KNeighborsRegressor

KNeighborsRegressor(n\_neighbors=10)





# Thanks

Ryan / Anne