# Silicon Valley

L'IA au service des agents immobiliers

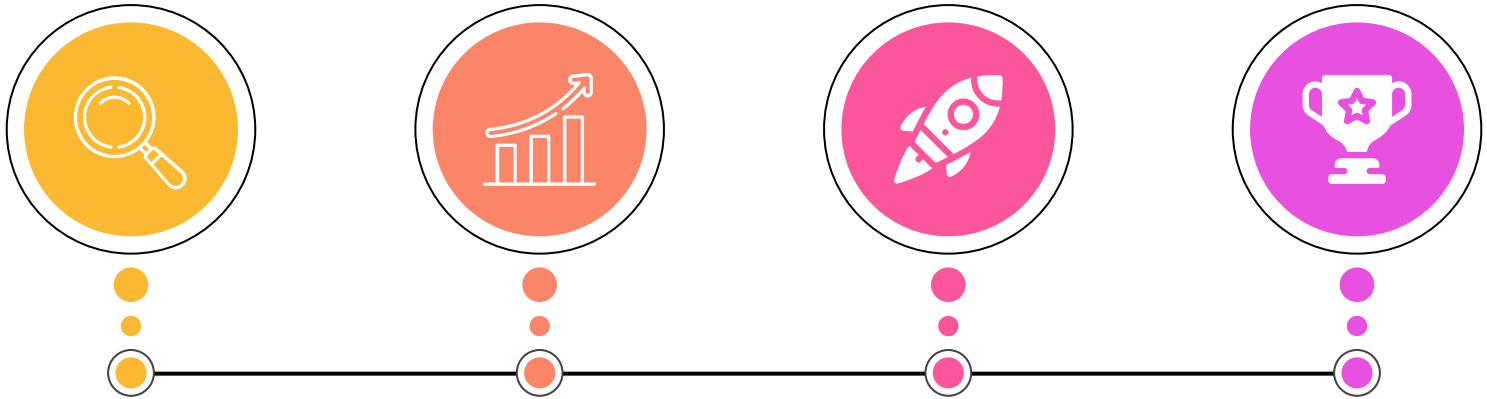
# Contexte du projet

Vous êtes développeur AI dans une startup de la Silicon Valley qui fournit des services dans le domaine de l'investissement immobilier. Les chargés de relation client ont mentionné que la demande a augmenté récemment et qu'il devient difficile de faire des estimations personnalisées. De ce fait, l'entreprise vous a confier d'automatiser cette tâche avec un modèle prédictif.

Pour cela, vous avez récupéré une base de données qui contient les prix médians des logements pour les districts de Californie issus du recensement de 1990.

L'objectif est de créer un modèle avec vos données (train) pour prédire la valeur du prix médian des maisons par district / bloc (medianHouseValue). A la fin du projet, vous devez évaluer ce modèle avec les données (validation) que seul votre client dispose (le prof).

# Notebooks



## EDA

Exploration / Missing values / Encoding / Visualisation

## INFERENCE

Analyse d'inférence statistique avec Statsmodels

## MODEL

Présentation de toutes les itérations

## PREDICT

Pre-cleaning et utilisation du modèle

# 01 EDA

**Encoding**
Encodage de ocean_proximity avec get_dummies (librairie Pandas)

*clean_df = pd.get_dummies(df, columns=['ocean_proximity'], prefix=["place"])*

**Dropping**
Suppression de la colonne Unnamed:0 qui n'a aucune utilité

*df.drop(['Unnamed: 0'], axis=1, inplace=True)*

**Missing values**
Traitement des 176 données manquantes sur total_bedrooms avec le KNN Imputer

*knn_imputer = KNNImputer(n_neighbors=2, weights="uniform")*
*df['total_bedrooms'] = knn_imputer.fit_transform(df[['total_bedrooms']])*

# 02 INFERENCE

**Variable dépendante** : median_house_value

***Accuracy de l'OLS regression*** *: 0.62*

**Coef**
Les variables explicatives, housing_median_age, total_bedrooms et households ont la meilleure corrélation avec median_house_value

**Standard error :**
total_rooms, total_bedrooms et place_INLAND ont les erreurs standards les moins élevés et ont donc une meilleure estimation.

# 03 MODEL

**Iteration 1 / R2 : 0.65**
Régression linéaire (avec toutes les features)

**Iteration 2 / R2 : 0.54**
Régression linéaire avec les 3 features les plus corrélées

**Iteration 3 / R2 : 0.0**
Dummy Regressor (avec toutes les features)

**Iteration 4 / R2 : 0.76**
Random Forest (avec toutes les features)

**Iteration 5 / R2 : 0.55**
Standard Scaler / PCA / Régression linéaire

# 04 PREDICT

Voir EDA pour le pre_cleaning puis :

**K-MEANS (3 clusters)**
Insertion ensuite dans le dataframe afin de créer de nouvelles features

| clusters_0 | clusters_1 | clusters_2 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |

**Robust Scaler**
Traitement de la distribution des données avec outliers

**KNN Regressor (5 neighbors)**
Afin d'obtenir, selon moi, la meilleure prédiction, le KNN Regressor était l'algorithme le plus adéquat en utilisant les points voisins suggérant le meilleur prix de vente.

**R2 = 0.90**