03/06/2024

PRESENTATION PROJET MEMOIRE

THEME :

UTILISATION D'ALGORITHMES DE MACHINE LEARNING AVEC PYTHON POUR CONSTRUIRE DES MODELES DE PREDICTION. ETAT DE L'ART ET ANALYSE SUR UN CAS D'ETUDE.

Assadou Jocelyn APPIA

ANNEE 2024

MASTER 1

Table des matières

1ere PARTIE : ETAT DE L'ART

I- INTRODUCTION AU MACHINE LEARNING ET A SES APPLICATIONS EN PREDICTIONS.

Le terme « Machine Learning » ou « apprentissage automatique » en français, n'est plus un concept nouveau aujourd'hui.

Déjà Arthur Samuel (1959), défini le Machine Learning comme « (…) la discipline donnant aux ordinateurs la capacité d'apprendre sans qu'ils soient explicitement programmés ». Dans sa suite, Tom Mitchell (1997), aborde une approche un peu plus technique quand il expose le scénario suivant : « Étant donné une tâche T et une mesure de performance P, on dit qu'un programme informatique apprend à partir d'une expérience E si les résultats obtenus sur T, mesurés par P, s'améliorent avec l'expérience E » [1]. Dans le même sens, Fabien Benureau (2015) affirme que : « *l'apprentissage est une modification d'un comportement sur la base d'une expérience* »[2]. En 1990 la première application Machine Learning était le fonctionnement du filtre antispam. Nous remarquerons que les filtres sont faits automatiquement selon des paramètres définis et que le mécanisme sous-jacent le fonctionnement de l'ordinateur permet de bloquer certains messages soupçonnés d'être des spams jusqu'à ce qu'une action de notre part le qualifie comme tel.

Le machine learning (ML) est une branche de l'intelligence artificielle (IA) qui est axée sur la création de systèmes qui apprennent, ou améliorent leurs performances, en fonction des données qu'ils traitent[3].L'Intelligence Artificiel est défini comme l'ensemble des techniques mises en œuvre afin de construire des machines capables de faire preuve d'un comportement que l'on peut qualifier d'intelligent, fait aussi appel aux sciences cognitives, à la neurobiologie, à la logique, à l'électronique, à l'ingénierie et bien plus encore[4].

---

[1] Aurélien Géron(2017), Machine Learning with Scikit-Learn and TensorFlow, ISBN 9781491962299, Edition DUNOD, traduit de l'anglais par Anne Bohy. P14.
[2] Clhoé-Agathe Azencott(2022), Introduction au Machine Learning, 2e Edition DUNOD
[3] https://www.oracle.com/fr/artificial-intelligence/machine-learning/what-is-machine-learning/
[4] Clhoé-Agathe Azencott(2022) bis.

L'apprentissage automatique est une forme d'IA qui consiste en un système qui s'améliore grâce à l'expérience alors que l'IA peut être un simple ensemble de règle et d'heuristiques[5].

Historiquement, **l'Intelligence Artificielle** est un domaine scientifique issu des mathématiques, de l'informatique et de la biologie. Deux objectifs étaient présents dès l'origine :

- Soit créer une machine intelligente qui pourrait résoudre tout problème se présentant à elle.
- Soit, plus simplement, simuler un processus cognitif, c'est-à-dire « donner l'impression » que la machine est intelligente.

Le premier but correspond à l'intelligence artificielle dite « **générale** » (ou IA forte). La machine serait alors dotée d'émotions et de capacités de raisonnement poussé, lui permettant « d'apprendre à apprendre ». Cependant, elle n'existe pas encore (si elle apparaît un jour) en dehors de la science-fiction. Le second but correspond à l'intelligence artificielle dite « étroite », ou IA faible. C'est celle qui existe aujourd'hui. Elle ne peut résoudre les problèmes que l'un après l'autre : un modèle spécifique correspond à un seul processus cognitif. Il n'y a pas vraiment d'intelligence au sens courant pour les humains, mais une simulation de celle-ci[6].

Le machine Learning est un ensemble de méthodes statistiques appliquées à l'IA. Elles permettent en particulier à l'IA d'apprendre à partir de données d'exemple. L'algorithme le plus simple et le plus connu du ML est la régression linéaire, qui consiste à trouver la droite approximant un nuage de points (grâce à ça, on peut facilement « classifier » des données en fonction de leur position par rapport à cette droite de régression). Cette technologie permet de réaliser des prédictions en consultant des données (ou datas) en se basant sur des statistiques. Les premiers algorithmes ont été élaborés en 1950 : le plus célèbre d'entre eux est le Perceptron. Toute démarche requérant l'usage du Machine Learning doit passer par 4 étapes :

- Le pré-traitement des données
- La modélisation

[5] Ludovic DE MATTEIS Steeven JANNY - Solal NATHAN – Wenqi SHU-QUARTIER (2022), Introduction à l'apprentissage automatique, in CULTURE SCIENCE DE L'INGENIEUR Edition Ecole Normale Supérieure Paris-Saclay.

6 Virginie MATHIVET (2024), Machine Learning Implémentation en Python avec Scikit-learn (2e édition),

- Le déploiement
- La maintenance

Une fois les données filtrées, ordonnées et sauvegardées, elles peuvent être exposées aux algorithmes du Machine Learning[7].

Intéressons-nous au second but qui semble être réaliste. Puisque le Machine Learning est la capacité d'apprendre tout seul selon les expériences qui lui ont été communiquées, il est opportun de s'interroger sur le fonctionnement et les outils qui permettent cette prouesse.

II- L'APPRENTISSAGE AUTOMATIQUE / ANALYSE PREDICTIVE

1- Qu'est-ce que l'apprentissage automatique ?

Le domaine de l'intelligence artificielle a pour objectif de parvenir à simuler l'intelligence humaine et en particulier l'apprentissage de nombreuses tâches. Deux méthodes sont alors possibles pour apprendre :

- L'apprentissage par cœur consiste à mémoriser explicitement tous les exemples possibles afin de pouvoir les restituer ;
- • L'apprentissage par généralisation a pour objectif d'extraire des règles implicites à partir d'une quantité d'exemples afin de les réappliquer à de nouvelles situations jamais rencontrées. L'apprentissage par cœur est relativement aisé pour une machine à condition de disposer des exemples. En revanche, l'apprentissage par généralisation est difficile car il demande d'extraire des règles qui ne sont pas explicitement mentionnées dans les exemples. Ce défi constitue le cœur l'apprentissage automatique.

Comme expliqué précédemment, l'apprentissage automatique est un champ de l'IA porté sur l'analyse statistiques de données d'apprentissage. Historiquement, cette branche est définie comme le développement de machines capables d'apprendre sans avoir été explicitement programmées à apprendre une tâche.

[7] Laurent Gimazane (), **Les différents algorithmes de l'IA Suite de "Les différents types d'IA, Enseigner l'intellligence artificielle,**

a- Les phases de l'apprentissage automatique

De manière générale, les algorithmes d'apprentissage automatique se séparent en plusieurs phases.

- Phase d'entraînement (ou d'apprentissage) : le modèle choisi est soumis à un grand nombre d'exemples significatifs. Le système cherche alors à apprendre des règles implicites en se basant sur ces données (appelées données d'entraînement). Cette phase d'entraînement précède généralement l'utilisation du modèle, bien que certains systèmes continuent d'apprendre indéfiniment s'ils disposent d'un retour sur les résultats (on appelle cela de l'apprentissage en ligne).

- Phase d'inférence : Le modèle entraîné peut être utilisé sur de nouvelles entrées. Les entrées fournies lors de la phase d'inférence peuvent être traitées même si elles n'ont pas été vues par le modèle lors de la phase d'apprentissage. En effet, grâce à l'extraction de règles implicites, le modèle peut se généraliser à des entrées inconnues**.**

b- Applications et quelques Limites

L'apprentissage automatique peut être appliqué à des problèmes complexes dans divers domaines. On trouve des algorithmes d'apprentissage automatique dans la santé pour l'aide au diagnostic médical et la création de vaccins (comme en 2019, où SAM a développé un vaccin contre la grippe), en robotique pour la vision par ordinateur et les commandes vocales, et dans l'industrie pour la détection de pannes et la gestion de systèmes. Ces algorithmes sont également utilisés dans les voitures autonomes, les jeux (comme AlphaZero, Stockfish, DeepBlue, AlphaGo), et en astronomie pour améliorer les observations. Bien que l'apprentissage automatique soit flexible et capable de résoudre des problèmes complexes, il présente certaines limites dont il faut être conscient :

- La précision des prédictions n'est jamais de 100 %, il existe donc une probabilité (même faible) de se tromper un jour, ce qui peut être bloquant pour certaines applications ;

- La correction et la détection d'erreur est difficile dans la plupart des cas, ce qui rend parfois le développement et l'entraînement du modèle difficile. Cette limite est directement liée à la question d'explicabilité ;
- Les performances du modèle sont fortement dépendantes des données d'entraînement, ce qui demande au concepteur de choisir des données suffisamment représentatives pour que le modèle soit généralisable ;
- Un problème récurrent dans l'apprentissage automatique est le fléau de la dimension. Il s'agit d'un phénomène arrivant lorsque l'on cherche à analyser des données de grande dimension[8].

Il ne faut pas oublier les nombreuses questions éthiques soulevées par l'apprentissage automatique et, plus largement, par l'intelligence artificielle. Par exemple, l'algorithme COMPAS, utilisé par les autorités américaines pour évaluer le risque de récidive des criminels, a été montré comme biaisé par une enquête de l'organisation ProPublica en 2016, jugeant les Afro-Américains avec un risque bien plus élevé. Cet exemple illustre que les algorithmes d'apprentissage automatique peuvent produire des modèles biaisés et restent influencés par leurs concepteurs et les données disponibles.

III- LES TYPES D'APPRENTISSAGE AUTOMATIQUE

Le Machine Learning consiste à créer un modèle issu d'un apprentissage (ou entraînement). Un « modèle » est donc un programme informatique créé non pas par un développeur, mais grâce à un algorithme d'apprentissage.

Il existe plusieurs formes d'apprentissage définies par les données en entrée (ou variables explicatives) et les données en sortie (ou variables cibles).

Dans chaque forme d'apprentissage, il existe plusieurs tâches de ML spécifiques, qui correspondent à des catégories de problèmes. Ces tâches sont indépendantes du domaine considéré.

Nous distinguons trois types d'apprentissage automatique :

[8] Ludovic DE MATTEIS Steeven JANNY - Solal NATHAN – Wenqi SHU-QUARTIER (2022), Introduction à l'apprentissage automatique, in CULTURE SCIENCE DE L'INGENIEUR Edition Ecole Normale Supérieure Paris-Saclay.

- L’apprentissage supervisé
- L’apprentissage non-supervisé
- L’apprentissage par renforcement

Dans le cadre de notre approche, **nous allons utiliser deux types d’apprentissage (supervisé et non-supervisé)** soit 3 méthodes pour étudier le phénomène du prix moyen de vente des biens selon les départements en France et précisément selon les communes issues d’un département au choix. Ces départements sont la Savoie et la Haute-Savoie. La relation étudiée est prix moyen de vente des biens selon la surface moyenne d’abord de faire interagir d’autre variables qui explique au mieux la variation de prix selon les départements et les des départements choisis.

1- Apprentissage supervisé

L'apprentissage supervisé se caractérise par l'utilisation de données d'entraînement étiquetées, c'est-à-dire des données pour lesquelles les sorties souhaitées sont connues. Si l'on note les N entrées xi et les sorties cibles associées, on obtient l'ensemble de données $D = \{xi , yi \}i\in[1,N]$. L'objectif est d'entraîner le modèle choisi afin qu'il puisse prédire correctement les sorties pour des entrées non étiquetées. Bien que la construction de l'ensemble d'apprentissage nécessite souvent un effort humain, l'apprentissage supervisé permet d'automatiser et d'accélérer des tâches qui seraient autrement laborieuses ou impossibles à réaliser. Les principales tâches de l'apprentissage supervisé incluent **la classification, la régression et la prévision.**

Pour notre cas d’exercice, nous verrons la classification et la régression.

a- La Classification.

Quand la variable est qualitative, on parle de classification. La classification consiste à sélectionner une catégorie (valeur) parmi toutes celles possibles. Dans cette tâche, la sortie

attendue est une catégorie, comme « malade » ou « sain » dans le domaine médical. C'est l'une des tâches les plus courantes, ce qui explique la disponibilité de nombreux algorithmes. L'algorithme d'apprentissage reçoit donc de nombreux exemples appartenant à toutes les catégories à prédire. Des algorithmes comme le perceptron, les k plus proches voisins, les réseaux de neurones et les arbres de décision sont couramment utilisés pour cette approche.

Exemple de méthode : K-NN (nearest neighbors).

b- La Régression.

La régression est une tâche qui analyse des données continues afin de trouver des relations entre des variables (généralement entre une variable dépendante et plusieurs variables indépendantes) pour prédire un résultat théorique en l'absence de mesures disponibles, comme des prévisions futures. La régression est principalement utilisée dans les modèles de prédiction et de prévision. Un algorithme de régression apprend et crée ses modèles à partir des caractéristiques des états actuels ou historiques des variables pour prédire une valeur continue. Cela peut mener à une relation de régression linéaire simple ou à des relations logarithmiques, exponentielles ou polynomiales plus complexes.

2- Apprentissage non supervisé

L'apprentissage non supervisé englobe tous les types d'apprentissage où il n'y a **pas de sortie connue**, pas d'enseignant pour instruire l'algorithme d'apprentissage.

Dans l'apprentissage non supervisé, l'algorithme d'apprentissage reçoit simplement les données d'entrée et il lui est demandé d'extraire des connaissances à partir de ces données.

a- Le Clustering

Le clustering consiste à partitionner l’ensemble de données en groupes, appelés clusters. L’objectif est de diviser les données de manière que les points d’un même cluster soient très similaires et les points de différents clusters différents. Les algorithmes de clustering

attribuent (ou prédisent) un nombre à chaque point de données, indiquant à quel cluster appartient un point particulier. k-Means Clustering est l'un des algorithmes de clustering les plus simples et les plus couramment utilisés. Il essaie de trouver des centres de cluster représentatifs de certaines régions des données. L'algorithme alterne entre deux étapes : assigner chaque point de données au centre de cluster le plus proche, puis définir chaque centre de cluster comme la moyenne des points de données qui lui sont attribués. L'algorithme est terminé lorsque l'affectation des instances aux clusters ne change plus.[9]

D'autres méthodes d'apprentissage non-supervisé existent, ce sont :

- L'association
- Détection d'anomalies
- La réduction dimensionnelle.

2e PARTIE : CAS PRATIQUES

Donnée utilisée (Data) : Indicateurs Immobiliers par commune et par année (prix et volumes sur la période 2023)

Source : https://www.data.gouv.fr/fr/datasets/r/d7881695-1cb5-44c1-900c-00c7158ab766

[9] Andreas C. Müller and Sarah Guido (2017) Introduction to Machine Learning with Python, Editor: Dawn Schanafelt. P 168.

Description :

Ce jeu de données centralise pour l'année 2023 plusieurs variables agrégées sur le marché de l'immobilier résidentiel à l'échelle des communes :

- Le nombre de mutations
- Le nombre de ventes de maisons et d'appartements
- La proportion de ventes de maisons et d'appartements
- Le prix moyen des biens vendus
- Le prix moyen au m2 des biens vendus
- La surface moyenne des biens vendus

Le champ d'identification (et de jointure) des communes est basée sur le code INSEE (COG 2022)[10]

Ces données sont dérivées d'un traitement de la base DVF géolocalisée, voici les mutations prises en comptes :

- Mutation mono ventes (pas de ventes en lots)
- Prix entre 15 000 € et 10 000 000 €
- Surfaces des appartements (entre 10m² et 250m²) et surfaces des maisons (entre 10m² et 400m²)
- Prix au m² entre 330 €/m² et 15 000 €/m²

Action de prétraitement des données : Le fichier a fait objet d'une identification des codes communes ('**INSEE_COM')** aux des communes correspondants, ensuite identifier les communes selon les départements correspondants. Les documents utilisés pour les différentes méthodes sont : donnees_biens_associe_dept.csv (avec identification des départements) et donnee_bienvendu_depart.csv (avec identification des communes).

Variable et Hypothèses : La variable à expliquer est le Prix moyen des biens vendus.

Les hypothèses formulées sont :

H0 : Le prix moyen des biens vendus est identique dans toutes la France.

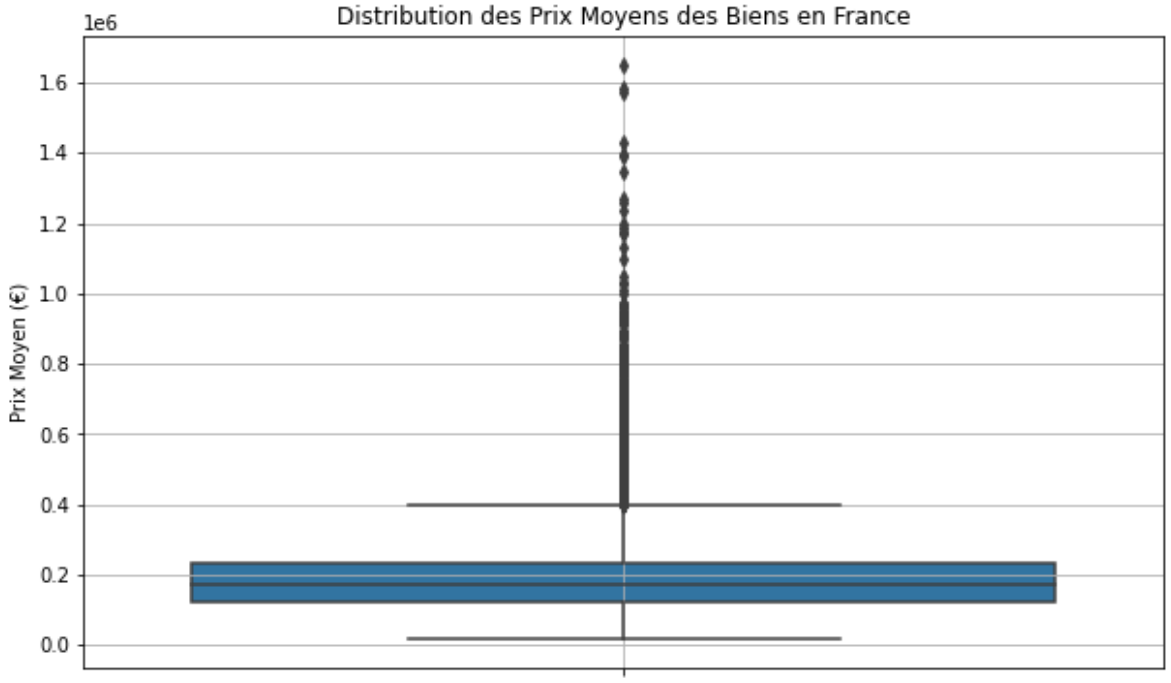
H1 = Le prix moyen des biens vendus dépends est différents selon les départements.

[10] https://www.data.gouv.fr/fr/datasets/indicateurs-immobiliers-par-commune-et-par-annee-prix-et-volumes-sur-la-periode-2014-2023/

A- METHODE DE LA REGRESSION

ANALYSE STATISTIQUE DE LA DONNEE :

```
# Créer une boîte à moustaches pour la distribution des prix moyens
plt.figure(figsize=(10, 6))
sns.boxplot(y=datadepartement['PrixMoyen'])
plt.title('Distribution des Prix Moyens des Biens en France')
plt.ylabel('Prix Moyen (€)')
plt.grid(True)
plt.show()
```



Interprétation :

- 25% des départements ont un prix moyen des biens inférieur à 1.234812^{e}+05 €.
- 50% des départements ont un prix moyen des biens inférieur à 1.714625^{e}+05 €.
- 25% des départements ont un prix moyen des biens supérieur à 2.343594^{e}+05 €.

Il y'a un nombre assez important en dehors de la boite de moustache, ces valeurs anormales (aberrantes) suggèrent une distribution anormale due aux valeurs extrêmes.

Analysons la courbe de distribution du prix moyens des biens vendus.

```
#DISTRIBUTION DU PRIX MOYEN EN FRANCE


plt.figure(figsize=(10, 6))
sns.histplot(datadepartement['PrixMoyen'], bins=30, kde=True)
plt.title('Distribution du prix moyen des biens vendus EN FRANCE')
plt.xlabel('prix_moyen_biens_vendus')
plt.ylabel('Fréquence')
plt.show()
```

```
:
#ASYMETRIE ET APPLATISSEMENT DE LA COURBE EN FRANCE

prix_moyen_par_dept = datadepartement.groupby('Departement')['PrixMoyen'

prix_moyen = prix_moyen_par_dept['PrixMoyen']

skewness = skew(prix_moyen)
kurt = kurtosis(prix_moyen)

print(f"Skewness: {skewness}")
print(f"Kurtosis: {kurt}")
```

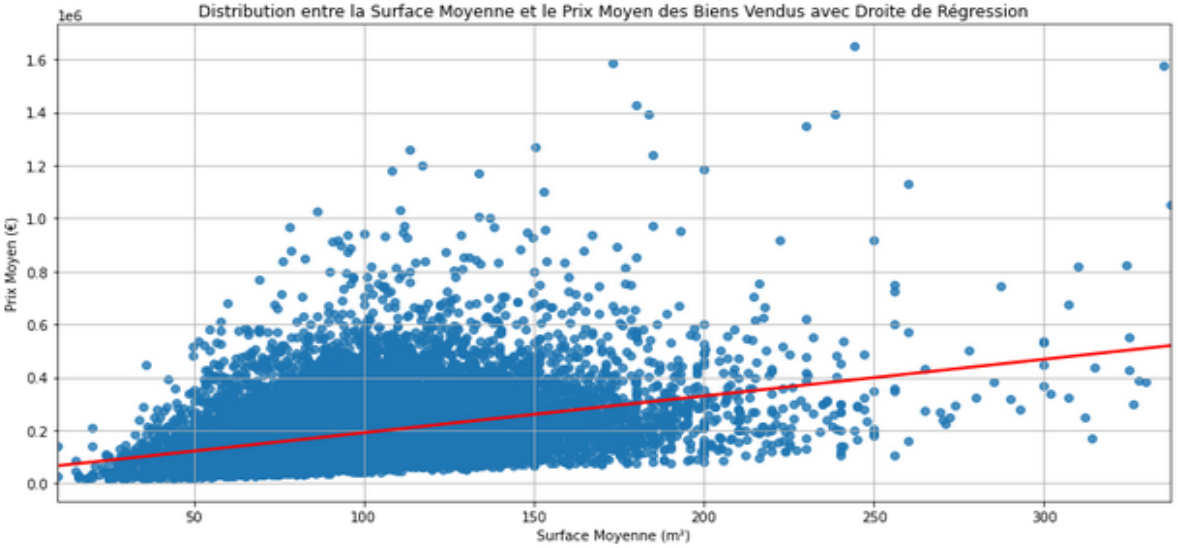
Interprétation :

- Le prix Moyen de bien vendu est de 15.000 euros minimums contre 1650890 euros maximum dans la France.
- Le Skweness = 1.64 > 0 on a une asymétrie positive vers la droite due aux valeurs extrêmes avec aucune incidence sur la moyenne.
- Le Kurtosis = 3,46 > 3 On a une très forte concentration des valeurs autours de la moyenne qui font qu'on a une asymétrie positive vers la droite. On a une distribution **leptokurtique** qui s'observe par le pique de la courbe autour de la moyenne

(distribution qui s'élève assez haut puis retombe assez brutalement) et un prolongement de la distribution vers la droite (valeur positive).

- **RELATION ENTRE PRIX MOYEN ET SURFACE MOYENNE**

```
# Créer le nuage de points avec la droite de régression

plt.figure(figsize=(12, 6))
sns.regplot(x='SurfaceMoy', y='PrixMoyen', data=datadepartement, ci=None,
plt.title('Distribution entre la Surface Moyenne et le Prix Moyen des Bie
plt.xlabel('Surface Moyenne (m²)')
plt.ylabel('Prix Moyen (€)')
plt.grid(True)
plt.tight_layout()
plt.show()
```




```
# la correlation entre les deux variables


correlation = datadepartement['SurfaceMoy'].corr(datadepartement['PrixMoy

print(f"La corrélation entre la surface moyenne et le prix moyen est de
```

Interprétation : On observe une corrélation positive entre la surface moyenne et le prix moyen des biens vendus.

- DROITE DE REGRESSION ENTRE DEUX VARIABLES

```
# Calculer les coefficients de la droite de régression
import scipy.stats as stats

# Calculer les coefficients de la droite de régression
slope, intercept, r_value, p_value, std_err = stats.linregress(datadepart

# Calculer le coefficient de détermination (R²)
r_squared = r_value**2

print("Équation de la droite de régression :")
print(f"PrixMoyen = {intercept:.2f} + {slope:.2f} * SurfaceMoy")
print(f"Coefficient de corrélation (r) : {r_value:.2f}")
```

Interprétation :

La droite de régression ainsi définie entre le prix moyen des biens vendus et la surface moyenne en France est égale à : Prix Moyen (Y) = 52167.24 + 1386.20 * Surface Moyenne(X).

Avec une corrélation positive entre le prix moyen et la surface moyenne, l'on peut conclure que la surface moyenne explique seulement 13% de la variation totale du prix moyen. On peut constater que le prix moyen dépend aussi d'autres variables.

- REGRESSION LINEAIRE MULTIPLE

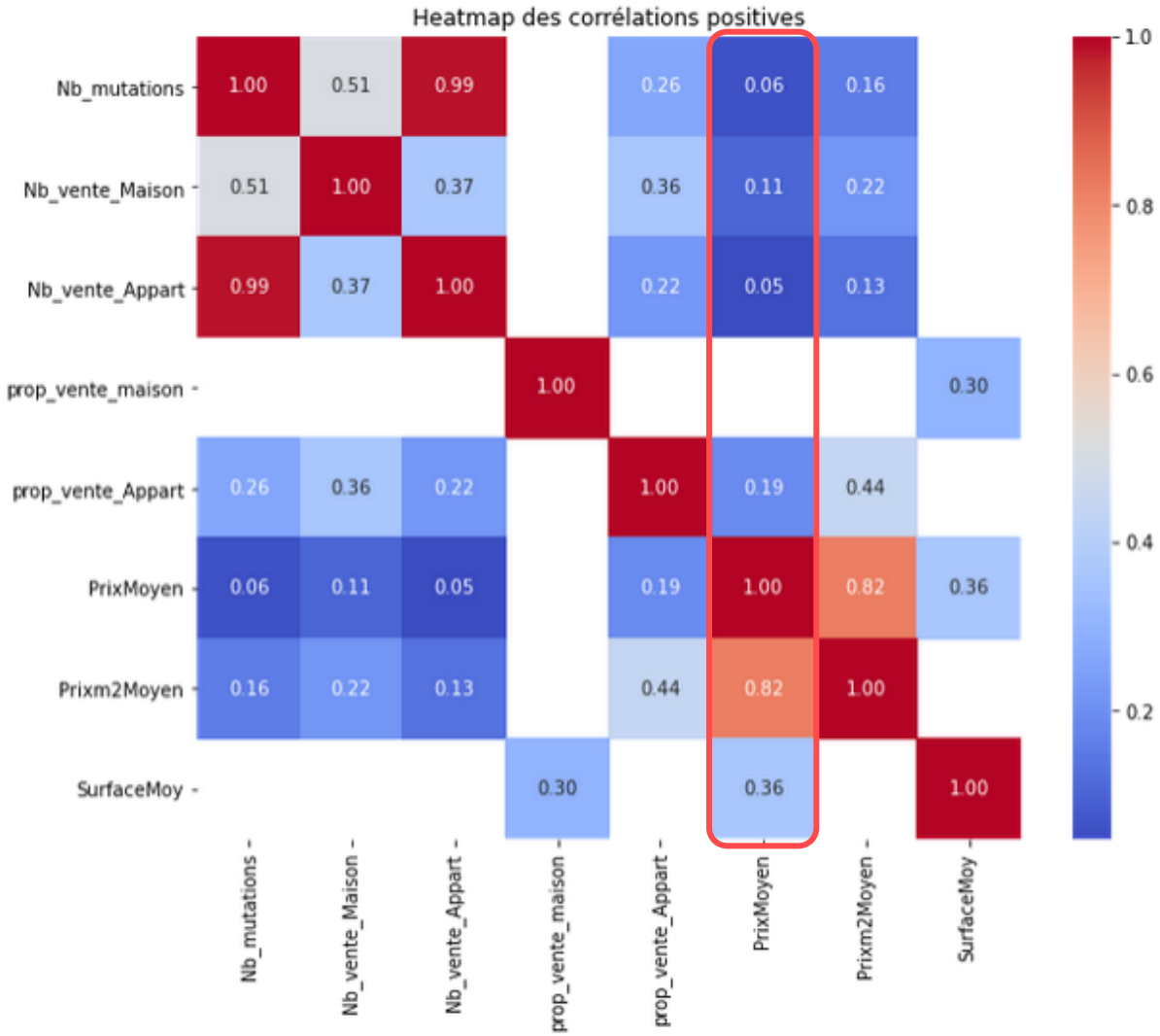
```
# Sélectionner les caractéristiques et la variable cible
## Prix moyen du bien vendu (y) depend de ....
''
X = datadepartement[['Nb_mutations', 'Nb_vente_Maison','Nb_vente_Appart'
y = datadepartement['PrixMoyen']
```

On considère toutes les variables disponibles dans la base de données mis en relation avec le prix moyen des biens vendus en France.

```python
# Calculer la matrice de corrélation
corr_matrix = datadepartement.corr()

# Afficher la matrice de corrélation
print(corr_matrix)

# Créer une heatmap pour visualiser la matrice de corrélation
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Heatmap des corrélations')
plt.show()
```

Affichons que les variables qui ont une corrélation positive.

```python
# Garder uniquement les corrélations positives
positive_corr_matrix = corr_matrix[corr_matrix > 0]

# Afficher la matrice de corrélation avec les corrélations positives
print(positive_corr_matrix)

# Créer une heatmap pour visualiser les corrélations positives
plt.figure(figsize=(10, 8))
sns.heatmap(positive_corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Heatmap des corrélations positives')
plt.show()
```


Heatmap des corrélations positives
Nb_mutations
Nb_vente_Maison
Nb_vente_Appart
prop_vente_maison
prop_vente_Appart
PrixMoyen
Prixm2Moyen
SurfaceMoy
1.00 0.51 0.99 0.26 0.06 0.16
0.51 1.00 0.37 0.36 0.11 0.22
0.99 0.37 1.00 0.22 0.05 0.13
1.00 0.30
0.26 0.36 0.22 1.00 0.19 0.44
0.06 0.11 0.05 0.19 1.00 0.82 0.36
0.16 0.22 0.13 0.44 0.82 1.00
0.30 0.36 1.00
1.0
0.8
0.6
0.4
0.2

- DROITE DE REGRESSION

```
# Supprimez les colonnes non pertinentes pour la régression (par exemple,
X = datadepartement.drop(['PrixMoyen', 'Departement'], axis=1)
y = datadepartement['PrixMoyen']

```

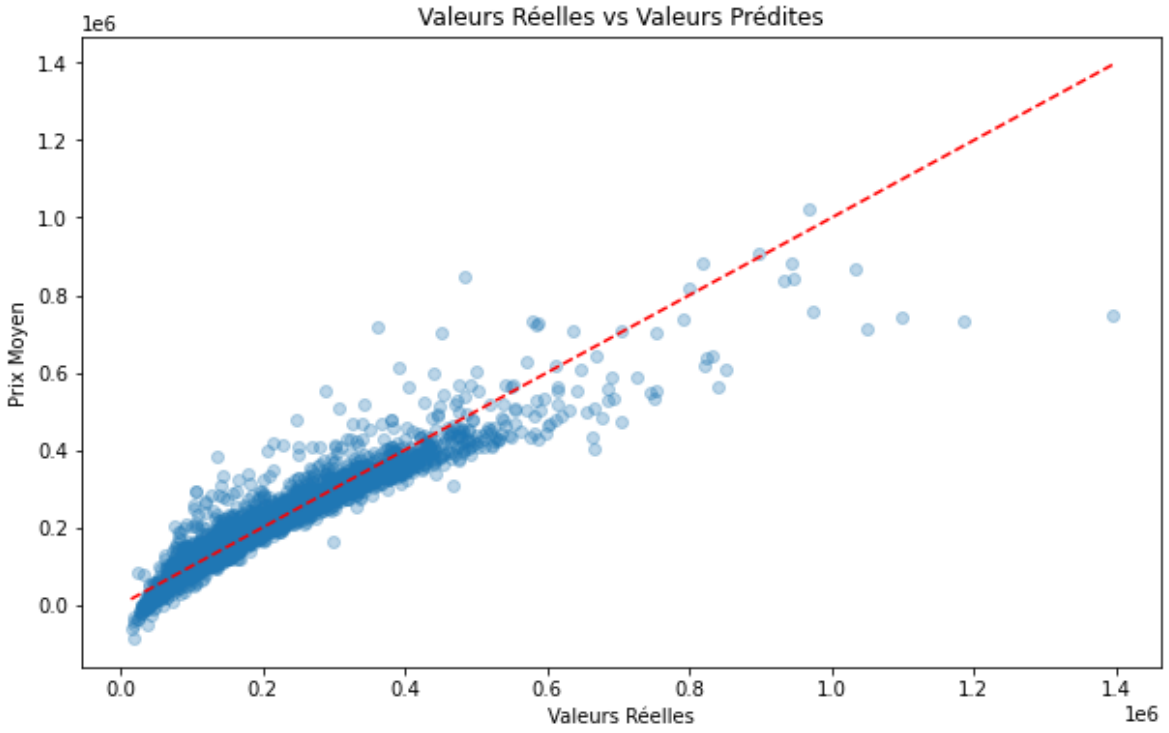
```
# Diviser les données en ensembles d'entraînement et de test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,

```

```
# Entraîner le modèle de régression linéaire multiple
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
# Faire des prédictions sur l'ensemble de test
y_pred = model.predict(X_test)
```

```
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, alpha=0.3)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], '--r')
plt.xlabel('Valeurs Réelles')
plt.ylabel('Prix Moyen')
plt.title('Valeurs Réelles vs Valeurs Prédites')
plt.show()
```



- Evaluation du modèle

```python
# Évaluer les performances du modèle
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)

print(f"Coefficient de détermination (R²) : {r2:.2f}")
print(f"Erreur quadratique moyenne (MSE) : {mse:.2f}")

# Afficher les coefficients de la régression
coefficients = pd.DataFrame(model.coef_, X.columns, columns=['Coefficient
print(coefficients)
```

```
Coefficient de détermination (R²) : 0.90
Erreur quadratique moyenne (MSE) : 1094497395.73
                  Coefficient
Nb_mutations         9.774949
Nb_vente_Maison     34.132774
Nb_vente_Appart    -24.357826
prop_vente_maison  230.929503
prop_vente_Appart -230.929503
Prixm2Moyen         92.668419
SurfaceMoy        1738.270137
```

Interprétation : Le modèle explique 90% de la variation du prix moyen des bien vendus.

- Equation de la Droite

```python
# Construire l'équation de la droite de régression
intercept = model.intercept_
equation = f"PrixMoyen = {intercept:.2f}"
for var in variables_pos_corr:
    coef = coefficients.loc[var, 'Coefficient']
    equation += f" + ({coef:.2f} * {var})"

print("Équation de la droite de régression :")
print(equation)
```

PrixMoyen = -163511.83 + (9.77 * Nb_mutations) + (34.13 * Nb_vente_Maison) + (-24.36 * Nb_vente_Appart) + (-461.86 * prop_vente_Appart) + (92.67 * Prixm2Moyen) + (1738.27 * SurfaceMoy)

Qu’en est-il de notre hypothèse de départ ?

Pour répondre à l’hypothèse de départ :

H0 : Le prix moyen des biens vendus est identique dans toutes la France.

H1 = Le prix moyen des biens vendus dépends est différents selon les départements.

Le test de l’ANOVA est le plus approprié (test pour évaluer la relation entre une variable quantitative et variable quantitative).

```python
import statsmodels.api as sm
from statsmodels.formula.api import ols

# Définir le modèle
model = ols('PrixMoyen ~ C(Departement)', data=datadepartement).fit()

# Effectuer l'ANOVA
anova_table = sm.stats.anova_lm(model, typ=2)

# Afficher les résultats
print(anova_table)
```

```
                    sum_sq       df           F  PR(>F)
C(Departement)  9.821956e+13     93.0  140.293241     0.0
Residual        2.252070e+14  29916.0         NaN     NaN
```

<u>Interprétation</u> :

La F-value est = 0, alors on prend 5% de risque pour rejeter H0. On estime qu’il existe une différence significative entre les prix moyens des biens vendus et les départements.

B- METHODE DE LA K-NN

L'algorithme des k-plus proches voisins (k-NN : pour k-nearest neighbors en anglais) est une méthode intuitive et facilement paramétrable pour traiter des problèmes de classification avec un nombre quelconque d'étiquettes. Son principe est très simple : pour chaque nouveau point *X*, on commence par identifier l'ensemble de ses k-plus proches voisins parmi les points d'apprentissage, que l'on note Vk(*X*). Ensuite, on attribue au nouveau point xxx la classe majoritaire au sein de Vk(*X*). Bien entendu, il est nécessaire de choisir $1 \leq k \leq n$ pour que cela ait un sens.

Le choix de la valeur de K pour effectuer une prédiction avec K-NN varie en fonction du jeu de données. En général, utiliser un petit nombre de voisins (dans notre cas k = 3) risque de provoquer du sous-apprentissage (underfitting : une faible précision sur les données d'entrainement, une simplification excessive et un échec à capturer les tendances). À l'inverse, utiliser un grand nombre de voisins (une grande valeur de K) rend la prédiction plus fiable. Cependant, si l'on utilise K égal au nombre total d'observations (K = N), on risque de sur-apprentissage (overfitting : haute précision sur les données d'entraînements, faible précision sur les données de test, complexité excessive), ce qui entraîne une mauvaise généralisation du modèle sur de nouvelles observations.

- CAS D L'ETUDE:

Objectif : Prédire le prix moyen des biens vendus par département selon les variables qui expliquent la variation du prix moyen.

```
1 datadept = datadept.dropna()
2 datadept.count()
```

```
INSEE_COM       30010
Nom_DEPT        30010
Nb_mutations    30010
NbMaisons       30010
NbApparts       30010
propmaison      30010
propappart      30010
PrixMoyen       30010
Prixm2Moyen     30010
SurfaceMoy      30010
dtype: int64
```

Notre DATASET est constitué de 30010 lignes.

Transformons le nom des départements en variables catégorielles :

```
##Transformation de la variable cartégorielle en numerique

datadept['Nom_DEPT'] = datadept['Nom_DEPT'].astype('category').cat.codes
```

Traitement, préparation de la donnée et normalisation :

```
# Utiliser toutes les colonnes sauf 'prix moyen' comme caractéristiques

X = datadept.drop(columns = ['PrixMoyen'])

# Utiliser 'prix moyen' comme la colonne cible

y = datadept['PrixMoyen']

# Diviser les données en ensembles d'entraînement et de test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rar
```

```
# 4. Normalisation des données
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Application du modèle K-NN pour la régression et prédiction :

```
# Application du modèle K-NN pour la régression
k = 3
#instance du modèle KNeighborsRegressor de la bibliothèque sklearn
# pour la régression K-NN.
knn = KNeighborsRegressor(n_neighbors=k)
#Entrainement du modèle  et ajustement.
knn.fit(X_train, y_train)
```

```
KNeighborsRegressor(n_neighbors=3)
```

```
# Prédiction sur l'ensemble de test
y_pred = knn.predict(X_test)
```

Le nombre K = 3, pour chaque nouvelle prédiction, le modèle K-NN utilisera les 3 voisin les plus proches.

- EVALUTAION DU MODELE

```python
# Évaluation du modèle
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
print(f"R² Score: {r2:.2f}")
```

```
Mean Squared Error (MSE): 691010452.68
Root Mean Squared Error (RMSE): 26287.08
R² Score: 0.94
```

Interprétation :

En moyenne les carrés des erreurs de prédiction sont de 691.010.452.68 euros au carrées. La prédiction du modèle s'écarte en moyenne de 26.287.08 euros du prix réel.

Le modèle nous renseigne que 94% de la variance des prix est expliquée par le modèle.

```python
# Comparaison des valeurs réelles et prédites
comparaison_des_valeurs = pd.DataFrame({'Actual': y_test, 'Predicted': y_pre
print(comparaison_des_valeurs.head())
```

```
             Actual      Predicted
11960  211509.855263  218981.954475
24284   93500.000000  103577.777778
23985   87363.142857  107805.416667
28140  191881.212500  262217.908422
16636  104800.000000   99683.333333
```

<u>Interprétation</u> :

Pour un échantillon de 11960 données

Valeur réelle : 211509.86 euros

Valeur prédite : 218981.95 euros

La prédiction est assez proche de la valeur réelle, avec une légère surestimation de 7472.10 euros

- FONCTION DE PREDICTION DU PRIX MOYEN SELON EN FRANCE

```python
# Fonction de prédiction
def knn_predict(new_data, k=3):
    # Normaliser les nouvelles données
    new_data_scaled = scaler.transform(new_data)

    # Créer le modèle K-NN
    knn = KNeighborsRegressor(n_neighbors=k)

    # Entraîner le modèle sur les données existantes
    knn.fit(X_train, y_train)

    # Prédire les valeurs pour les nouvelles données
    predictions = knn.predict(new_data_scaled)

    return predictions
```

```python
# Test de la fonction de prédiction
new_data = pd.DataFrame({
    'Nom_DEPT': [0, 1],  # codes des départements
    'Nb_mutations': [10, 50],
    'NbMaisons': [8.0, 45.0],
    'NbApparts': [2.0, 8.0],
    'propmaison': [80.0, 90.0],
    'propappart': [20.0, 10.0],
    'Prixm2Moyen': [2500.0, 3000.0],
    'SurfaceMoy': [100.0, 150.0]
})

predictions = knn_predict(new_data)
print(predictions)
```

```
[233381.26851852 342300.89590965]
```

<u>Interprétation</u> : Analyse des Prédictions

Première Prédiction :

Pour une région (département codé 0) avec 10 mutations, 8 maisons vendus, 2 appartements vendus, 80% ventes de maison, 20% vente d'appartement, un prix moyen au mètre carré de 2500.0 et une surface moyenne de 100.0, le modèle prédit un prix moyen de 249032.45 €

Deuxième Prédiction :

Pour une région (département codé 1) avec 50 mutations, 45 maisons, 5 appartements, un prix moyen au mètre carré de 3000.0 et une surface moyenne de 150.0, le modèle prédit un prix moyen de 362600.28.

C- METHODE DE LA K-MEANS (APPRENTISSAGE NON SUPERVISE)

Le clustering consiste à diviser les données en groupes, appelés clusters ou classes, de manière que les individus d'un même groupe soient similaires entre eux, tandis que ceux de groupes différents présentent des différences.

Il procède en identifiant des centres de classes représentatifs de certaines zones de l'espace des données.

L'algorithme alterne entre deux étapes :

- attribuer chaque individu à la classe dont le centre est le plus proche, puis

- recalculer le centre comme la moyenne des points qui lui sont associés. L'algorithme s'arrête lorsque l'affectation des individus aux classes se stabilise.

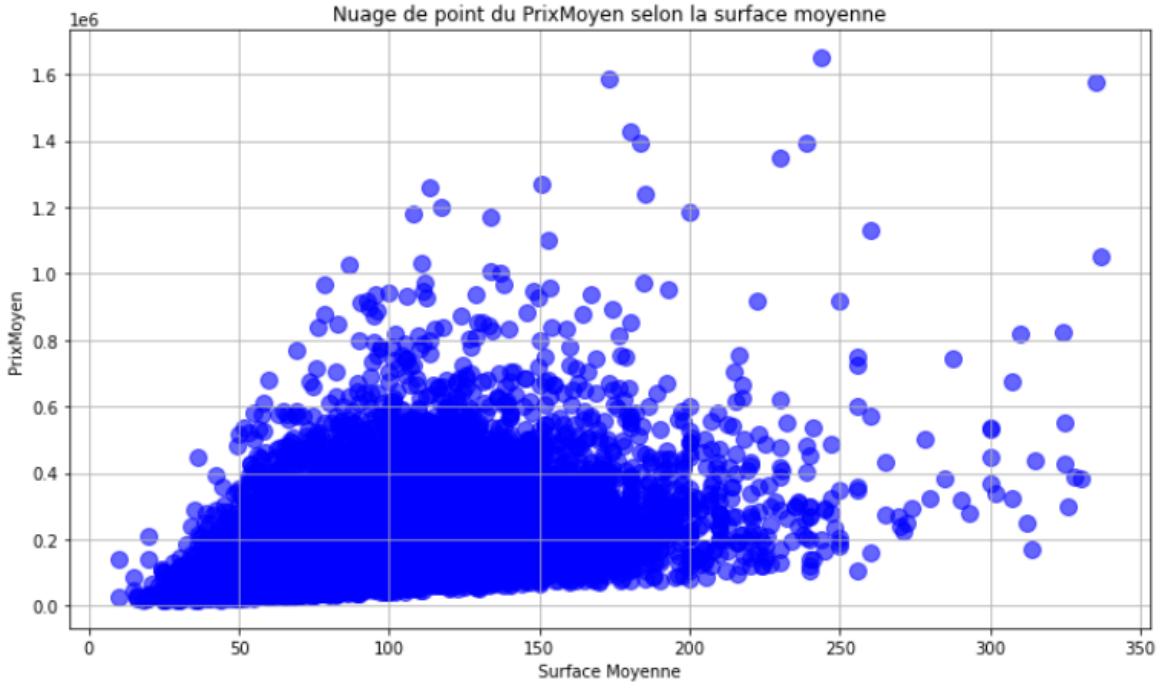
Le but est de faire ressortir les patterns cachés dans la donnée en regroupant les éléments qui se « ressemblent ».

- MISE EN PRATIQUE SELON NOTRE BASE DE DONNEE

Objectif : Etablir un cluster (3 groupes) issue de l'ensemble des départements afin de les regrouper selon leurs ressemblances.

- Visualisation de la distribution du prix moyen selon la surface en France.

```
# Distribution de PrixMoyen
plt.figure(figsize=(10, 6))
plt.scatter(dt_departement['SurfaceMoy'], dt_departement['PrixMoyen'], c='bl
plt.title('Nuage de point du PrixMoyen selon la surface moyenne')
plt.xlabel('Surface Moyenne')
plt.ylabel('PrixMoyen')
plt.grid(True)
plt.tight_layout()
plt.show()
```



- Préparons les Clusters

```
# Définir le nombre de clusters (par exemple, 3)

kmeans = KMeans(n_clusters=3, random_state=0)

kmeans.fit(X_scaled)

# Ajouter les étiquettes de clusters au DataFrame
dt_departement['Cluster'] = kmeans.labels_
```

- Transformons la base de données dt_département en DataFrame panda pour faciliter la lecture.

Les DataFrames de pandas sont un outil puissant et polyvalent pour travailler avec des données tabulaires en Python, offrant une large gamme de fonctionnalités pour l'analyse, la manipulation et la visualisation des données.

```
dt_departement = pd.DataFrame(dt_departement)
```

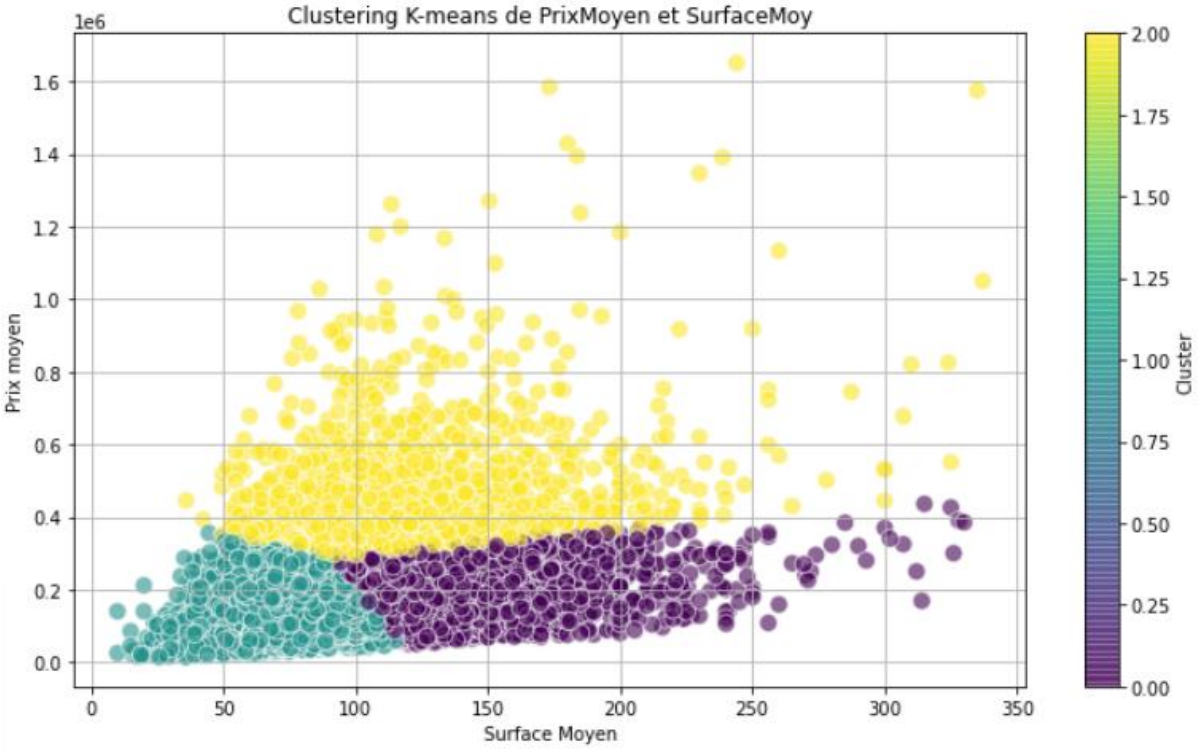
- Création des Cluster

```
# Sélection des colonnes pertinentes
X = dt_departement[['PrixMoyen', 'SurfaceMoy']]

# Standardisation des données (optionnel mais recommandé)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Application de K-means avec 3 clusters
kmeans = KMeans(n_clusters=3, random_state=0)
kmeans.fit(X_scaled)

# Ajouter les étiquettes de clusters au DataFrame
dt_departement['Cluster'] = kmeans.labels_

# Visualisation des clusters
plt.figure(figsize=(10, 6))
plt.scatter(dt_departement['SurfaceMoy'],dt_departement['PrixMoyen'], c=dt_
plt.title('Clustering K-means de PrixMoyen et SurfaceMoy')
plt.xlabel('Surface Moyen')
plt.ylabel('Prix moyen')
plt.colorbar(label='Cluster')
plt.grid(True)
plt.tight_layout()
plt.show()
```

X_scaled contiendra les données de X standardisées, ce qui est souvent une étape préalable importante dans de nombreuses analyses de données et modélisations statistiques. La standardisation permet de mettre toutes les caractéristiques sur la même échelle, ce qui est souvent nécessaire pour certaines techniques d'apprentissage automatique et d'analyse statistique.

La standardisation des données permet de ramener les données sur la même échelle afin de faciliter la représenter dans la dimension du graphique.

- Représentation graphique



Interprétation : la méthode K-means établis 3 familles / groupes / cluster de départements qui présentent sensiblement le même prix moyen des biens vendus selon la surface moyenne.

- IDENTIFIONS LES CENTRES DES CLUSTERS LES PLUS REPRESENTATIFS

```
# Afficher les paramètres du modèle
print("Nombre de clusters :", kmeans.n_clusters)
print("Centres des clusters :", kmeans.cluster_centers_)
print("Inertie :", kmeans.inertia_)
print("Labels des clusters :", kmeans.labels_)
```

```
Nombre de clusters : 3
Centres des clusters : [[ 0.13011798  0.82054461]
 [-0.46202614 -0.57722453]
 [ 2.10314431  0.45113584]]
Inertie : 29653.737555518514
```

On identifie les coordonnées des centres des clusters

Premier cluster [0.13011798 0.82054461]

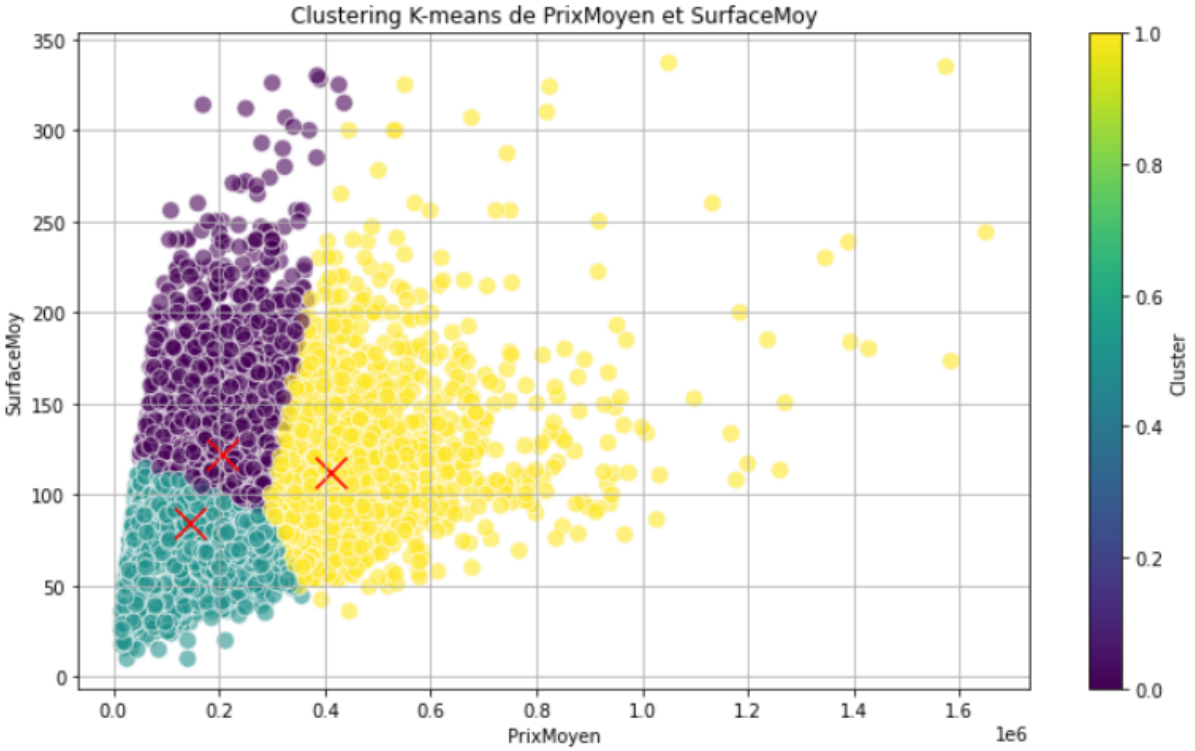
Deuxième Cluster [-0.46202614 -0.57722453]

Troisième cluster [2.10314431 0.45113584]]

Inertie : 29653.737 indique la somme des distances au carré des échantillons à leur centre de cluster le plus proche. Une inertie plus faible indique une meilleure cohésion des clusters.

- Représentation des centres de cluster :

```
# Visualisation des clusters avec les centres de clusters
plt.figure(figsize=(10, 6))
plt.scatter(dt_departement['PrixMoyen'], dt_departement['SurfaceMoy'], c=d
plt.scatter(scaler.inverse_transform(kmeans.cluster_centers_)[:, 0], scale
plt.title('Clustering K-means de PrixMoyen et SurfaceMoy')
plt.xlabel('PrixMoyen')
plt.ylabel('SurfaceMoy')
plt.colorbar(label='Cluster')
plt.tight_layout()
plt.grid(True)
plt.show()
```



Remarque : scaler.inverse_transform (kmeans.cluster_centers_) : Cette partie de code inverse la transformation appliquée aux centres des clusters lors de la standardisation. Les centres des clusters sont généralement calculés dans un espace standardisé (c'est-à-dire avec une moyenne de 0 et un écart-type de 1) pour l'algorithme K-means. En utilisant inverse_transform, ces centres sont ramenés à l'échelle originale des données.

```python
# Affichage du nombre de départements par cluster le plus proche
print("Nombre de départements par cluster le plus proche :")
print(df_departement_closest_cluster['ClosestCluster'].value_counts())
```

```
Nombre de départements par cluster le plus proche :
1    16789
0    10144
2     3077
Name: ClosestCluster, dtype: int64
```

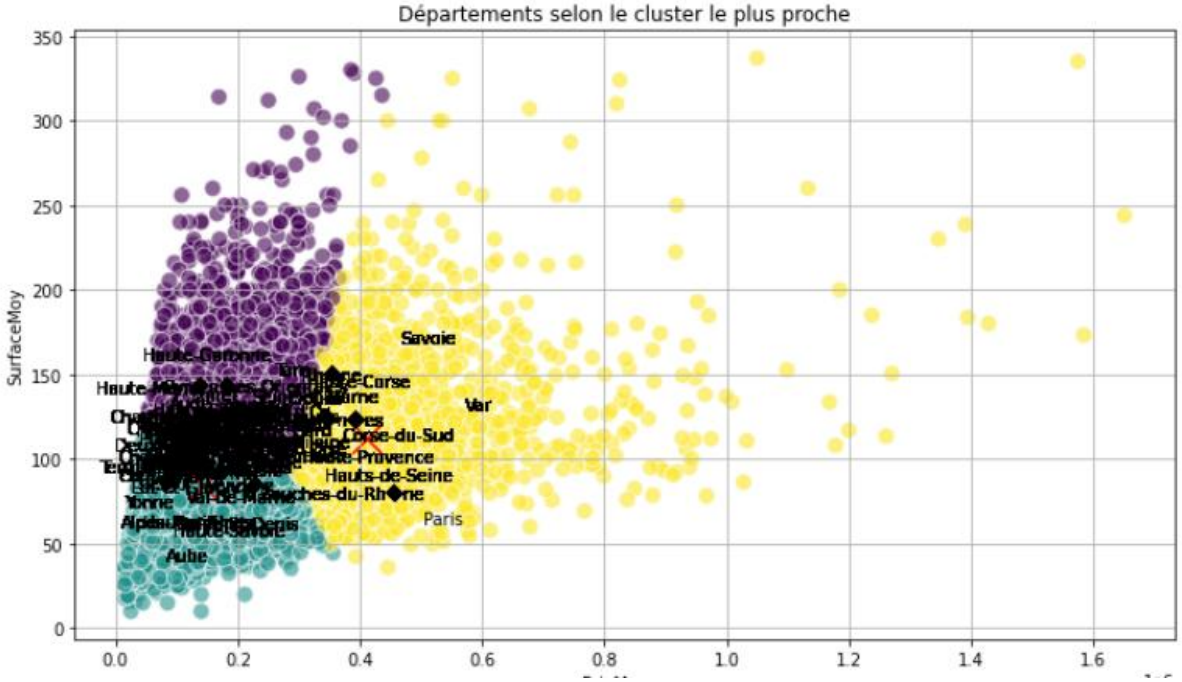
Interprétation :

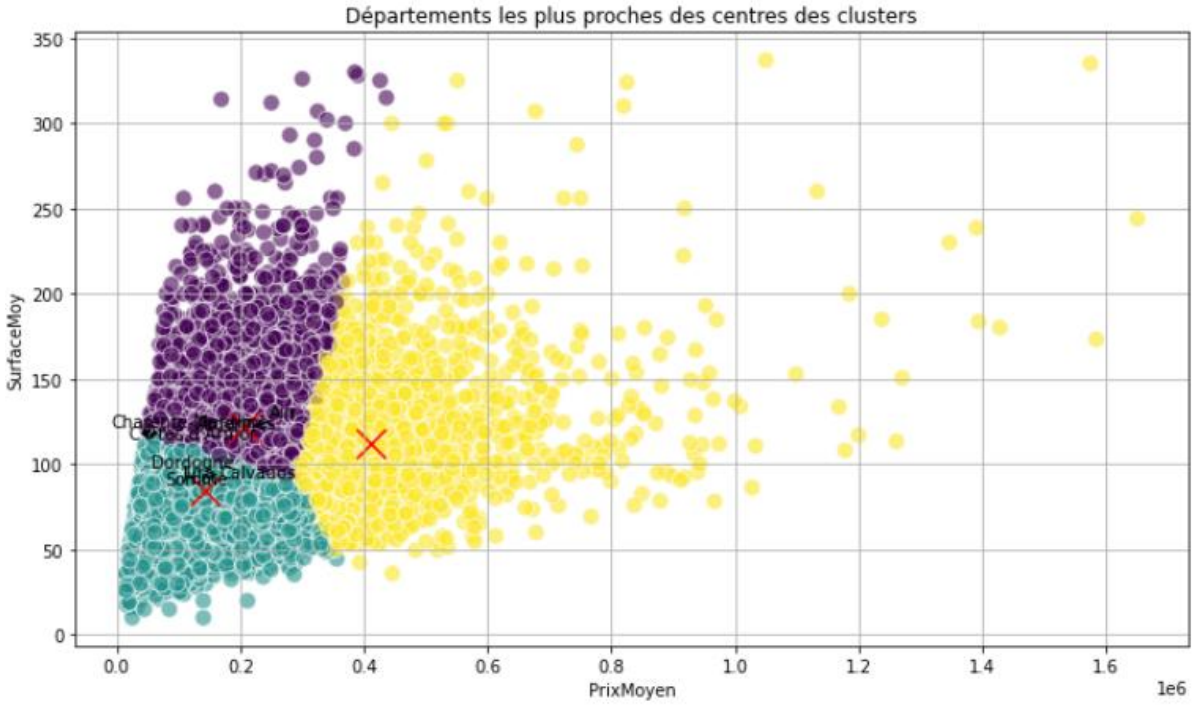
Le cluster labélisé 1 (vert) contient dans son groupe 16789 département très proche de son centre.

Le cluster labélisé 0 (Violet) contient dans son groupe 10144 département très proche de son centre.

Le cluster labélisé 2 (Jaune) contient dans son groupe 3077 département très proche de son centre.

- IDENTIFIONS LES DEPARTEMENTS (2) QUI PARTAGENT SENSIBLEMENT LES MÊMES CRITERES





Interprétation :

- les départements Dordonnes, Aube proposent sensiblement un prix Moyen presque identique selon la surface moyenne

- les départements Haute Garonnes, Ain proposent sensiblement un prix Moyen presque identique selon la surface moyenne

- les départements Haute-de-Seine, Savoie proposent sensiblement un prix Moyen presque identique selon la surface moyenne.

3e PARTIE : AVANTAGES ET INCONVENIENTS DES METHODES

- **REGRESSION LINEAIRE**

Avantages :

- Implémentation facile et rapide
- Facile à comprendre et interpréter

Limites :

- Limité à l'estimation de modèle linéaire, ce qui restreint les applications possibles.
- L'estimation que nous obtenons est influencée par les fluctuations ou les erreurs de mesure présentes dans nos données d'apprentissage. Par conséquent, si une donnée aberrante, c'est-à-dire une observation très éloignée du modèle que nous essayons d'estimer, est présente, elle aura un impact significatif sur nos estimations des paramètres du modèle.

- **K-NN (K PLUS PROCHES VOISINS)**

Avantages :

- Facile à mettre en œuvre : En raison de sa simplicité et de son efficacité, cet algorithme est souvent l'un des premiers que les nouveaux spécialistes des données apprennent.
- Facilement adaptable : L'algorithme peut être ajusté pour intégrer de nouveaux échantillons d'apprentissage, car toutes les données d'apprentissage sont conservées en mémoire.
- Peu d'hyperparamètres : Le KNN ne nécessite que la sélection d'une valeur pour k et une mesure de distance, ce qui est moins complexe que de nombreux autres algorithmes d'apprentissage automatique.

Limites :

- Le choix de la valeur K peut exercer une influence sur la prédiction de l'algorithme.
- Difficile de déterminer la méthode de calcul de la distance ainsi que le nombre de voisins K approprié à utiliser dans un algorithme de type k-plus proches voisins.
- La vitesse de l'algorithme diminue considérablement à mesure que le nombre d'observations et de variables indépendantes augmente.

- **K-MEANS (K-MOYENNES)**

Avantages :

- Implémentation facile et rapide
- Adapté aux gros data sets
- Complexité temporelle : la segmentation k-means est linéaire en nombre d'objet données, ce qui augmente le temps d'exécution.

Limites :

- Le choix du nombre de classe K peut exercer une influence sur le regroupement des classes.
- Cet algorithme ne prend pas en compte les éléments qui sont significativement éloignés de la moyenne de leur classe respective. En conséquence, cela peut entraîner une classification incorrecte de certains éléments dans des classes erronées.
- L'algorithme ne peut être exécutée que dans des données numériques.

BIBILOGRAPHIE

- Andreas C. Müller and Sarah Guido (2017) Introduction to Machine Learning with Python, Editor: Dawn Schanafelt. P 168.
- Aurélien Géron(2017), Machine Learning with Scikit-Learn and TensorFlow, ISBN 9781491962299, Edition DUNOD, traduit de l'anglais par Anne Bohy. P14.
- Clhoé-Agathe Azencott(2022) bis.
- Clhoé-Agathe Azencott(2022), Introduction au Machine Learning, 2e Edition DUNOD
- **https://www.data.gouv.fr/fr/datasets/indicateurs-immobiliers-par-commune-et-par-annee-prix-et-volumes-sur-la-periode-2014-2023/**
- https://www.oracle.com/fr/artificial-intelligence/machine-learning/what-is-machine-learning/
- Laurent Gimazane (), **Les différents algorithmes de l'IA Suite de "Les différents types d'IA, Enseigner l'intellligence artificielle,**
- Ludovic DE MATTEIS Steeven JANNY - Solal NATHAN – Wenqi SHU-QUARTIER (2022), Introduction à l'apprentissage automatique, in CULTURE SCIENCE DE L'INGENIEUR Edition Ecole Normale Supérieure Paris-Saclay.
- Ludovic DE MATTEIS Steeven JANNY - Solal NATHAN – Wenqi SHU-QUARTIER (2022), Introduction à l'apprentissage automatique, in CULTURE SCIENCE DE L'INGENIEUR Edition Ecole Normale Supérieure Paris-Saclay.
- Virginie MATHIVET (2024), Machine Learning Implémentation en Python avec Scikit-learn (2e édition),