Année 2025 - 2026

ANALYSE DU MARCHE IMMOBILIER A PARTIR DE DONNEES WEB SCRAPING AVEC PYTHON

Merchan Diego
Sivayanama Perveena

Table des matières

INTRODUCTION

L'analyse du marché immobilier est essentielle pour comprendre les dynamiques socio-économiques contemporaines. Les fluctuations des prix, les disparités géographiques, ainsi que les comportements d'offre et de demande influencent directement les stratégies des ménages, des investisseurs et des acteurs publics. Dans un contexte de tension sur le marché immobilier dans de nombreuses zones urbaines et périurbaines, il devient crucial de disposer d'indicateurs fiables et à jour pour analyser les tendances, anticiper les évolutions et orienter les décisions. Cependant, les données nécessaires à ces analyses sont souvent massives, hétérogènes et difficiles à accéder dans un format exploitable, représentant ainsi un défi méthodologique et technique.

Dans ce contexte, l'émergence du web scraping et des technologies de traitement des données ouvre de nouvelles opportunités. L'extraction automatisée d'annonces immobilières sur des plateformes spécialisées permet de constituer un corpus large, granulaire et représentatif des réalités du marché. Une fois collectées, ces données brutes doivent être nettoyées, filtrées et enrichies pour être transformées en une base analytique solide. Ce processus implique notamment la standardisation des prix, des surfaces et des types de biens, ainsi que l'ajout de dimensions spatiales telles que la géolocalisation ou l'appartenance régionale.

Au-delà du nettoyage, l'objectif est aussi de réaliser une analyse statistique approfondie du marché. De plus, la construction d'un modèle prédictif des prix, reposant sur un pipeline d'apprentissage automatique intégrant des variables géographiques, structurelles et contextuelles, contribue à objectiver les écarts de prix et à fournir une estimation algorithmique cohérente avec les données observées.

Enfin, la restitution des résultats ne constitue pas seulement un enjeu scientifique, mais aussi un enjeu d'accessibilité. Dans cette optique, un tableau de bord interactif développé avec Streamlit permet de démocratiser l'analyse en offrant une exploration intuitive des données : filtres dynamiques, visualisations avancées, carte géographique des biens, outils de comparaison et indicateurs synthétiques. Cette interface joue un rôle clé dans la valorisation du travail accompli, en rendant l'information compréhensible et accessible à un public non spécialiste.

Ainsi, ce projet combine les quatre dimensions essentielles d'une analyse de données : **collecte**, **nettoyage**, **analyse** et **visualisation interactive**, tout en mobilisant des outils professionnels et des méthodes rigoureuses issues de la data science.

Dans cette perspective, une question centrale guide l'ensemble de ce travail :
Comment le prix au mètre carré varie-t-il en fonction de la localisation, de la surface et du type de bien immobilier en France ?

Architecture générale du projet

L'architecture du projet a été élaborée selon une approche modulaire et hiérarchisée, permettant une séparation nette des responsabilités à travers les différentes étapes du pipeline de data science : acquisition des données, transformation, analyse, modélisation et visualisation.
Cette structure assure une meilleure maintenabilité du code, la reproductibilité des traitements et la possibilité d'extension future du système.

L'arborescence du projet se divise en plusieurs ensembles fonctionnels, chacun ayant un rôle spécifique :

1. Le dossier config/
Ce répertoire regroupe les paramètres globaux du projet, centralisés dans settings.py. On y trouve les chemins des données, les options du scraper, les constantes pour l'API, ainsi que des variables partagées entre modules.
Le fichier README.md documente le fonctionnement de cette configuration et sert de référence technique.

2. Le dossier data/
Il contient toutes les données manipulées au cours du pipeline, organisées en deux sous-répertoires :

- **raw/** : comprend les données brutes issues du scraping, notamment annonces_raw.csv.
- **processed/** : regroupe les données nettoyées (annonces_clean.csv), les résultats d'analyses (analysis_results.json), les produits dérivés comme la carte Folium (map_listings.html) et le modèle d'apprentissage automatique final (price_model.pkl).

Cette structuration permet de dissocier clairement les données sources des données transformées, conformément aux bonnes pratiques ETL.

3. Le dossier src/
C'est le cœur fonctionnel du projet, où chaque script a un rôle précis :

- **scraper.py** : collecte automatisée via l'API Bien'ici, gestion de la pagination, des doublons et stockage des annonces.
- **cleaner.py** : nettoyage avancé (prix, surface, pièces), enrichissement (géocodage, régions), calculs dérivés et validation des données.
- **analyser.py** : production d'indicateurs statistiques, segmentation, corrélations, détection d'outliers et clustering.
- **price_model.py** : entraînement du modèle de prédiction, prétraitement, validation et export du modèle.
- **analyse.py** : fonctions complémentaires d'analyse ou scripts secondaires aidant à l'exploration.

Cette organisation modulaire des fichiers favorise un développement agile et un débogage simplifié.

4. Le dossier dashboard/
Il renferme l'application Streamlit (app.py), qui constitue la couche de visualisation interactive du projet. Elle s'appuie directement sur les données traitées et le modèle

exporté pour offrir des filtres multi-critères, des cartes géographiques, des graphiques et des indicateurs de synthèse.

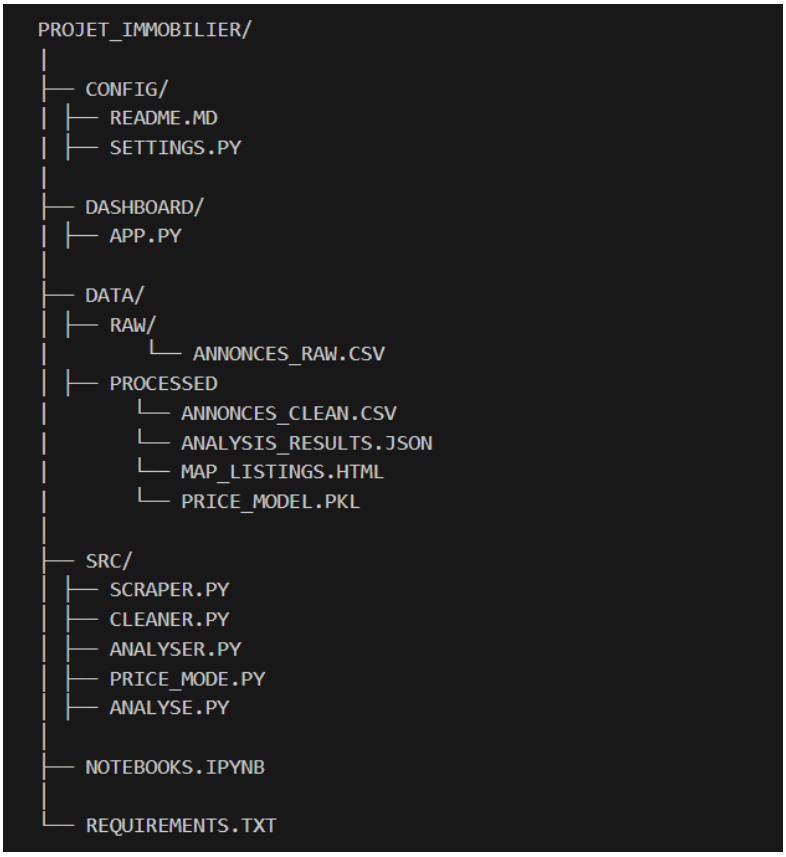
5. Le fichier notebooks.ipynb
Ce notebook sert d'espace exploratoire. Il permet :

- de tester rapidement du code,
- de visualiser des échantillons de données,
- de prototyper des analyses,
- de valider les fonctions avant leur intégration dans les scripts finaux.

Il joue un rôle d'interface entre l'exploration et la production.

6. Le fichier requirements.txt
Il liste toutes les dépendances du projet facilitant la reproductibilité et l'installation de l'environnement sur n'importe quelle machine.

Voici la structure du projet :



Méthodologie

La méthodologie mise en œuvre repose sur un pipeline complet de data science, allant de l'extraction automatisée des données à leur restitution sous forme de tableau de bord interactif.

1. Collecte des données par web scraping

1.1. Sources de données et choix techniques

Les données sont récupérées sur le site immobilier Bien'ici via son API, en utilisant le script scraper.py. Ce dernier emploie la bibliothèque « requests » pour interroger le point d'accès configuré dans « config/settings.py », avec des en-têtes HTTP spécifiques et un délai entre les requêtes pour minimiser les risques de blocage.

1.2. Paramétrage des requêtes

Le scraper génère dynamiquement les paramètres de l'API. Les résultats sont paginés : pour chaque page, un « offset » et un « nombre de résultats » sont calculés, permettant de naviguer progressivement dans le catalogue d'annonces sans surcharge pour le serveur.

1.3. Gestion des doublons et structure de sortie

Chaque annonce est identifiée par un identifiant unique dérivé de l'ID Bien'ici (préfixe bienici-…). Avant d'ajouter une annonce, le scraper vérifie qu'elle n'est pas déjà présente dans les données brutes, en se basant sur un ensemble d'IDs chargés depuis « annonces_raw.csv ». Cette approche prévient les doublons en cas de scraping répété. Les annonces sont ensuite stockées dans un fichier CSV unique dans « data/raw/annonces_raw.csv ».

1.4. Champs collectés

Pour chaque annonce, le scraper extrait et normalise :

- Informations financières : prix, charges
- Caractéristiques du bien : surface, nombre de pièces, nombre de chambres, type de bien (appartement/maison), étage, ascenseur, meublé, parking
- La localisation : ville, code postal
- Les métadonnées : date de publication, URL de l'annonce, date de scraping, source

Cette étape constitue la phase d'extraction du pipeline.

2. Nettoyage et enrichissement des données

Le traitement des données brutes est géré par le script « cleaner.py ». Il correspond à la phase transformation et vise à obtenir un jeu de données cohérent, propre et exploitable.

2.1. Chargement et suppression des doublons

Les données brutes sont chargées depuis « data/raw/annonces_raw.csv ». Les doublons sont supprimés en se basant sur la colonne « id », ne conservant que la première occurrence de chaque annonce.

2.2. Filtrage des types de biens

Pour se concentrer sur le résidentiel courant, seules les annonces de type appartement et maison sont retenues. Les autres types (locaux commerciaux, parkings indépendants, etc.) sont exclus, ce qui homogénéise le périmètre d'étude.

2.3. Nettoyage des prix

Les valeurs manquantes ou nulles concernant le prix sont éliminées. Un filtre est appliqué pour exclure les valeurs extrêmes jugées non réalistes (prix < 200 € ou > 10 000 €). Cette étape réduit l'impact des erreurs de scraping ou des annonces atypiques sur les analyses statistiques ultérieures.

2.4. Nettoyage des surfaces

De la même manière, les surfaces nulles ou manquantes sont supprimées. Un intervalle plausible est retenu (10 m² à 500 m²), ce qui exclut les micro-surfaces irréalistes et les très grands biens rares susceptibles de biaiser les agrégats.

2.5. Calcul du prix au mètre carré

Une nouvelle variable « price_per_m2 » est calculée comme le ratio prix / surface. Ce ratio est un indicateur central de l'analyse, permettant de comparer des biens de tailles différentes sur une base commune.

2.6. Traitement des pièces et chambres

Les colonnes « rooms » et « bedrooms » sont complétées :

- Les valeurs manquantes de « rooms » sont remplacées par 1 (studio).
- Les valeurs manquantes de « bedrooms » sont remplacées par 0.

- Une contrainte « bedrooms ≤ rooms – 1 » est imposée pour garantir la cohérence entre pièces et chambres.

2.7. Nettoyage des dates

La date de publication est convertie au format ISO YYYY-MM-DD. Les dates invalides ou héritées (comme 1970-01-01) sont remplacées par des valeurs manquantes pour éviter les interprétations erronées lors des analyses temporelles.

2.8. Enrichissement géographique : régions et géocodage

À partir du code postal, une fonction de mapping attribue à chaque bien une région administrative (Île-de-France, Auvergne-Rhône-Alpes, PACA, etc.). Cela permet des analyses par régions sans dépendre uniquement de l'information de la ville.

Pour les biens dont la ville, le code postal ou les coordonnées géographiques sont manquants, un géocodage automatique est réalisé via Nominatim. Un cache mémoire et un mécanisme de rate limiting limitent le nombre de requêtes et évitent les répétitions inutiles. Les colonnes latitude et longitude sont ainsi complétées, ce qui est essentiel pour la cartographie.

2.9. Sauvegarde des données nettoyées

Les données finales sont réorganisées et sauvegardées dans « data/processed/annonces_clean.csv », constituant le dataset de référence pour les analyses et la modélisation.

3. Analyse statistique et exploration

L'analyse statistique est gérée par le module « analyser.py », qui calcule des indicateurs de synthèse et génère un ensemble structuré de résultats dans « analysis_results.json ».

3.1. Statistiques descriptives

Pour les variables clés (prix, prix/m², surface), sont calculés :

- Moyenne, médiane, minimum, maximum
- Écart-type, coefficient de variation
- Percentiles (25 %, 50 %, 75 %, 90 %, 95 %, 99 %)

Ces résultats permettent d'identifier la dispersion, la présence de valeurs extrêmes et l'asymétrie du marché.

3.2. Corrélations

Un sous-ensemble de variables numériques (prix, surface, pièces, chambres, prix/m², étage) est utilisé pour établir une matrice de corrélation. Les coefficients (notamment prix–surface, prix–pièces et surface–pièces) servent à quantifier les relations linéaires entre les caractéristiques structurelles du logement.

3.3. Analyses géographiques

Les données sont agrégées par « ville » et par « région » :

- Pour chaque ville/région : prix moyen, prix médian, prix/m² moyen, surface moyenne et volume d'annonces
- Construction d'un top 10 (villes les plus chères, villes au prix/m² le plus élevé, villes avec le plus grand nombre d'annonces)

3.4. Segmentation et typologies

Deux formes de segmentation sont mises en place :

Par gamme de prix : les biens sont répartis en classes (Économique, Moyen, Premium, Luxe) selon des intervalles de prix définis dans le code.
Par nombre de pièces : pour chaque catégorie de pièces, des statistiques spécifiques sont calculées (prix moyen, prix/m², surface moyenne).

3.5. Impact des équipements

L'impact d'attributs tels que « ascenseur », « meublé » ou « parking » est mesuré en comparant les prix moyens selon la présence ou non de ces équipements, et en calculant un "premium" relatif (en pourcentage).

3.6. Outliers et score de « bonne affaire »

Des biens atypiques sont identifiés :

- Les plus chers en valeur absolue
- Ceux au prix/m² le plus élevé
- Ceux jugés "suspiciously cheap", en dessous de deux écarts-types sous la moyenne

Un « score de valeur » (value_score) est également calculé pour repérer les « bonnes affaires » en tenant compte de la surface, du nombre de pièces et du contexte de prix moyen de la ville.

4. Modélisation prédictive du prix

La modélisation est réalisée par le script price_mode.py (ou train_price_model.py dans la version générique), qui construit un pipeline scikit-learn complet.

4.1. Sélection et préparation des variables

Les données nettoyées sont filtrées pour ne conserver que les lignes avec price > 0 et surface > 0, sans valeurs manquantes pour les variables critiques (prix, surface, ville, région).

4.2. Sauvegarde du modèle

Le modèle final est enregistré sous forme de fichier « price_model.pkl » dans « data/processed », via la bibliothèque joblib, afin d'être directement réutilisable par l'application Streamlit.

5. Visualisation, cartographie et tableau de bord

La dernière étape vise à rendre les résultats accessibles à un utilisateur final, grâce à des outils de visualisation statique et interactive.

5.1. Carte interactive Folium

Le script « visualizer.py » charge les données nettoyées, filtre les biens possédant des coordonnées valides, puis génère une carte Folium centrée sur la moyenne des latitudes/longitudes. Les biens sont représentés par des cercles avec « marker clustering », et chaque pop-up affiche ville, prix, surface, nombre de pièces et lien vers l'annonce originale. La carte est exportée en HTML (map_listings.html).

5.2. Tableau de bord Streamlit

L'application « dashboard/app.py » constitue l'interface principale d'exploration des données. Elle :

- Charge le fichier « « annoncs_clean.csv » et le fichier d'analyses « analysis_results.json »
- Met en place des filtres latéraux (région, ville, surface, prix, nombre de pièces)
- Calcule des indicateurs de synthèse (nombre de biens, prix moyen, surface moyenne, prix/m²)
- Affiche des graphiques (barres, histogrammes, scatter plots) et une carte Folium intégrée
- Propose une page « Insights avancés » exploitant les résultats du module d'analyse (segmentation, clusters, comparaison appartements/maisons)

Le modèle de prix peut également être chargé (price_model.pkl) pour fournir des estimations directes au sein du tableau de bord.

En somme, cette méthodologie articule de manière cohérente la collecte, le nettoyage, l'analyse, la modélisation et la visualisation, conformément aux meilleures pratiques professionnelles en data science appliquée au marché immobilier.

Résultats

L'analyse des données révèle un marché immobilier particulièrement hétérogène, marqué par une forte dispersion des prix et une distribution asymétrique. Les moyennes sont influencées à la hausse par un nombre limité de biens très onéreux, tandis que la majorité des annonces se situe dans des tranches de prix plus modérées. L'étude du prix au mètre carré met en lumière une tendance claire : les petites surfaces, telles que les studios et T1, sont proportionnellement les plus coûteuses. Une corrélation négative significative entre la superficie et le prix/m² confirme ce phénomène, typique des zones urbaines où la demande pour des logements compacts est particulièrement forte.

La dimension géographique est cruciale dans la formation des prix. Les régions Île-de-France et Provence-Alpes-Côte d'Azur se distinguent largement, tant par le prix moyen que par le prix/m², tandis que d'autres régions affichent une variabilité plus limitée. Certaines villes se démarquent par un positionnement premium, lié à leur attractivité économique ou touristique. La comparaison entre appartements et maisons révèle deux dynamiques distinctes : les appartements, principalement en centre-ville, affichent un prix/m² élevé, alors que les maisons, bien que plus spacieuses, montrent un prix total supérieur mais une valorisation unitaire plus faible.

La segmentation en classes de prix permet une structuration claire du marché, le segment Medium représentant la majorité des biens analysés. Les équipements tels que l'ascenseur, le parking ou le fait d'être meublé contribuent à l'augmentation du prix moyen, surtout en milieu urbain où ces caractéristiques sont prisées. L'analyse des outliers met en lumière des biens excessivement chers ou anormalement bon marché, enrichissant ainsi la compréhension des extrêmes du marché et facilitant l'identification d'opportunités ou d'anomalies.

Dans l'ensemble, les résultats témoignent d'un marché structuré par la localisation, la superficie et les caractéristiques intrinsèques des biens, avec des disparités notables selon les zones et les usages.

Limites du projet

Bien que le projet dispose d'un pipeline complet et fonctionnel, plusieurs limites méritent d'être mentionnées pour bien cerner la portée des résultats.

- Dépendance au scraping: L'ensemble du système repose sur l'API publique de Bien'ici. Toute modification de son fonctionnement pourrait compromettre l'extraction, nécessitant une maintenance régulière.

- Géocodage via Nominatim : Bien que ce service soit fiable, il est soumis à des quotas stricts et à des délais d'attente, ce qui allonge le traitement et peut entraîner des valeurs manquantes pour certaines localisations.

- Limitation de source : Les analyses s'appuient uniquement sur les données d'un seul site, introduisant des biais potentiels liés à sa position ou à son type d'audience. Une étude plus représentative nécessiterait l'intégration de plusieurs plateformes immobilières.

- Absence de dynamique temporelle : Même si les dates de publication sont disponibles, la collecte n'est pas effectuée de manière récurrente, ce qui empêche l'analyse des tendances et des variations saisonnières.

Ces limites ouvrent des perspectives d'amélioration futures, tant au niveau des données que des méthodes d'analyse et de modélisation.

CONCLUSION

Ce projet a permis de développer un pipeline complet d'analyse immobilière, depuis la collecte automatisée jusqu'à la visualisation interactive des résultats. Les analyses ont mis en évidence l'hétérogénéité du marché, l'impact majeur de la localisation et la survalorisation des petites surfaces. Les segmentations, outliers et clusters ont permis de mieux comprendre la diversité des biens, tandis que le modèle prédictif fournit une estimation cohérente des prix.

Malgré certaines limites, dépendance au scraping, source unique, absence de dimension temporelle, le travail réalisé constitue une base solide pour des analyses plus avancées et ouvre la voie à des améliorations futures, notamment l'intégration de nouvelles sources et de modèles spatio-temporels plus précis.