Rapport de la preuve de concept : utilisation de Transformers pour la modélisation de données tabulées

Table des matières

I. Introduction 2
II. Les Tab Transformers 2
2.1 Principe 2
III. La base de données 3
3.1 Origine des données 3
3.2 Nettoyage et exploration 4
3.3 Création de nouvelles variables 7
3.4 Importance des features 8
IV. Modélisation 8
4.1 Les modèles « classiques » 9
4.2 Les réseaux de neurones 9
4.3 Les Transformers 10
4.4 Modèle averaging 10
4.5 Influence des features importances 10
V. Conclusion 11
VI. Références bibliographiques 11

Table des figures :

Figure 1 : Architecture d'un TabTransformer 3
Figure 2 : Figure caractérisant les mutations immobilières : 5
Figure 3 : Caractérisation de l'estimation du loyer : 6
Figure 4: Caractérisation des établissements scolaires : 6
Figure 5 : représentation de la répartition du type de transport en commun 7
Figure 6 : caractérisation des communes : 7
Figure 7 : Résultat des modélisations : 9

Table des tables :

Tableau 1 : Résultats de la modélisation avec l'ensemble des variables 8
Tableau 2 : Résultat de la modélisation avec les variables importantes. 9

I. Introduction

Depuis de nombreuses années, les techniques d'apprentissage machine sont en fort développement, notamment grâce à l'augmentation constante de la puissance de calcul disponible. Cela a notamment permis de développer des modèles de prédictions fiables sur des bases de données de tailles importantes, répondant à des problématiques complexes. Les modèles classiquement utilisés actuellement sont basés sur différentes méthodes : les méthodes linéaires et kernel trick, les arbres de décisions, les méthodes ensemblistes ou encore les réseaux de neurones et deeplearning. Aucune de ces méthodes ne peut être considérée meilleure qu'une autre, puisque les performances de chacune vont fortement dépendre du type et du nombre de données et de la complexité du problème. Concernant les données tabulées, qui représentent historiquement une part très importante du développement du machine learning, elles sont souvent plus adaptées à des modèles basés sur des arbres de décisions que des modèles de DeepLearning. En effet cette dernière, beaucoup plus adaptée au traitement d'images ou de textes, a des performances plus faibles que les arbres de décisions et a le désavantage de ne pas être facilement interprétable. Néanmoins les modèles basés sur les arbres de décisions ne conviennent pas à des données combinant des données tabulées et des images ou du texte par exemple et sont peu robustes aux données bruitées ou manquantes. Dans ce contexte, il a été démontré récemment par Huang et al [1] que les méthodes de Transformers appliquées couramment pour le traitement du langage naturel pouvaient tout à fait s'appliquer aux données tabulées et structurées. Selon les auteurs, ce traitement permet d'améliorer sensiblement les résultats des prédictions.

Le but de ce projet a été d'appliquer cette nouvelle approche de traitement des données appliquée aux données tabulées sur une base de données réelle et plus faiblement structurée afin de caractériser l'apport de cette méthode sur des données plus générales.

Dans une première partie le principe de fonctionnement des TabTransformers sera présenté ainsi que des exemples d'utilisation rapportés dans la littérature. Ensuite la construction de la base de données sera présentée, en détaillant l'origine de chaque information, le nettoyage et l'exploration effectués, ainsi que la création de nouvelles variables. Pour finir, les différents modèles testés seront présentés, afin de caractériser leurs performances et de conclure sur l'efficacité des Transformers sur ce type de données.

II. Les Tab Transformers

2.1 Principe

Les TabTransformers ont été développés par Huang et al, scientifique d'Amazon Web Service, dont les résultats ont été publiés en décembre 2020 [1]. Ces TabTansformers sont construits sur des transformateurs basés sur l'auto attention afin de transformer le plongement des variables catégorielles dans un plongement contextualisé permettant d'améliorer la performance du modèle. Historiquement le plongement de données a été étudié pour le traitement du langage naturel dans le but d'encoder les mots. L'ajout de la contextualisation apporté par BERT a permis de fortement améliorer les performances. Une explication complète du fonctionnement d'un transformateur peut être trouvée dans la référence [2]. L'architecture d'un TabTransfomer, représentée sur **Figure 1**, est composée d'une couche de plongement de données, d'un ou de plusieurs transformateurs et d'un perceptron multicouche. Chaque transformateur est composé d'une couche d'auto attention multi-têtes suivie par une couche d'anticipation. A la sortie des transfomateurs, les données catégorielles et continues sont concaténées avant d'être envoyées dans le perceptron multicouches. Dans ce projet nous utiliserons l'architecture optimale développée dans l'article de référence, composée de six couches, d'un plongement de dimension 32 et d'une couche d'attention à 8 têtes. Le perceptron final

est composé de deux couches de dimensions 4L et 2L respectivement avec L la dimension des données d'entrée.

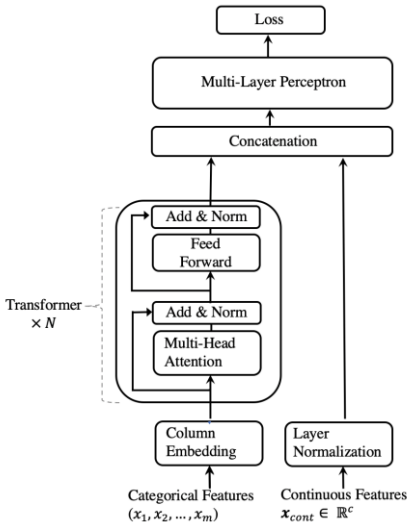

Figure 1 : *Architecture d'un TabTransformer*

Huang et al ont testé le modèle sur quinze bases de données différentes, toutes publiques, composées d'un nombre de variables catégorielles allant de 2 à 128. Dans tous les cas (ou presque) le modèle avec TabTransformer permet d'améliorer la performance du modèle comparé à un perceptron multicouche classique. Ces modèles sont capables d'accéder à des performances similaires à celles obtenues par des modèles basés sur des arbres de décisions. Dernièrement, plusieurs études ont été publiées en utilisant des modèles basés sur les TabTransformers en obtenant des bons résultats. On peut notamment citer des études sur la prédiction de maladies neurodégénératives [3] ou de la mortalité due à la COVID-19 [4], mais également pour étudier les attaques informatiques [5]. Très récemment, des auteurs ont proposé une méthode hybride combinant le modèle XGBoost et TabTransformer [6].

III. La base de données

3.1 Origine des données

La base de données utilisée dans ce projet a été créé à partir de la compilation de plusieurs bases de données disponible sur le site officiel des données publiques française (data.gouv.fr). Le notebook fourni avec les livrables (CHESNEAU_Erwan_2_notebook_creation_BDD_082022.ipynb). La base de données principales (téléchargeable ici) regroupe l'ensemble des mutations immobilières survenues en France depuis le 01 janvier 2017 jusqu'au 31 décembre 2021, séparées par départements. Dans ce projet il a été décidé arbitrairement de se focaliser sur les données de l'Essonne (91) afin de limiter la taille de la base de données et de pouvoir effectuer la preuve de concept sans utiliser une puissance de calcul (et de mémoire) importante. Cette base de données comporte environ 258k observations, reparties presque équitablement entre 2017 et 2021.

Afin de tenir compte de la localisation, en plus de la position géographique de chaque bien, une base de données contenant des informations sur les communes a été utilisée (téléchargeable ici). Elle comporte 36600 communes environ et va permettre d'apporter des informations comme le nombre d'habitants, l'accès aux soins, etc… (détaillé dans la partie 3.2). Afin de venir en complément des deux premières bases de données, une étude estimant le prix de loyer moyen par m² pour chaque commune a été téléchargée (lien ici pour les appartements et ici pour les maisons) afin d'incorporer ces informations.

Deux bases de données sont utilisées pour obtenir des informations sur l'accès à l'éducation : la première concerne les écoles (téléchargeable ici) et la seconde les universités et l'éducation supérieure (téléchargeable ici). Elles permettent de localiser chaque établissement en fonction du niveau. Pour finir, la liste avec la localisation de toutes les stations de tous les modes de transport en commun a été téléchargée (lien ici).

3.2 Nettoyage et exploration

Le nettoyage des bases de données permet de supprimer les valeurs aberrantes et d'harmoniser l'ensemble des données sélectionnées afin de faciliter la fusion de celles-ci. Les grandes étapes du nettoyage sont rappelées dans cette partie, pour plus de détails vous pouvez vous reporter au notebook suivant : CHESNEAU_Erwan_2_notebook_creation_BDD_082022.ipynb.

1. Mutations immobilières

La base de données contenant les mutations immobilières contient initialement 40 variables. Toutes ces informations ne seront pas utiles pour la caractérisation du bien et ne seront donc pas utilisées pour la prédiction du prix. Les variables conservées contiennent les informations sur la localisation du bien (latitude, longitude, code commune), la surface du bien et de son terrain, son type (maison, appartement…), le type de la mutation (vente, échange, expropriation…), le nombre de pièces du bien et bien évidemment la valeur foncière qui est la valeur à prédire. Certaines mutations sont séparées sur plusieurs lignes, notamment lorsqu'elle contient des biens sur différentes parcelles : toutes les mutations ont été regroupées sur une même ligne en sommant les surfaces et en concaténant les types de locaux notamment. Certaines mutations contiennent uniquement des dépendances ou contiennent des locaux commerciaux. Dans ce projet seuls les appartements ou les maisons sont étudiés : les lignes précédemment citées sont donc supprimées. Certaines mutations contiennent un nombre important de biens, ce qui doit correspondre à des opérations immobilières très particulières et qu'il est difficile à caractériser, il a donc été décidé de supprimer les mutations contenant plus de 3 biens. La représentation des dates des mutations met en évidence un nombre de ventes plus important l'été, et on observe également une diminution importante du nombre de vente durant le confinement lié à la COVID-19. La variable de la date est transformée pour créer le mois de la vente et l'année (**Figure 2 d**). Certaines mutations concernent uniquement une vente de terrain. Ce projet ne s'intéressant pas à ces observations il a été décidé de les supprimer. La représentation de la valeur foncière en fonction de la nature de la mutation (**Figure 2 e**) met en évidence que la valeur foncière moyenne n'est pas équivalente pour toutes les natures de mutation. La distribution de la valeur foncière (**Figure 2 a**) met en évidence des valeurs très importantes sur certaines mutations et d'autres avec une valeur très faible. Compte tenu de leur faible nombre, elles seront complexes à caractériser et il est donc décidé de supprimer les observations avec une valeur supérieure à 1M € et inférieure à 10k€. De même l'étude de la surface des biens met en évidence des biens d'une surface très faible (<10m²), ce qui est très improbable : on supprime donc ces observations. La représentation de la valeur foncière en fonction de la surface (**Figure 2 b**) met en évidence une corrélation importante (R=0.71), comme on peut s'y attendre. L'étude de la surface du terrain met quant à elle en évidence des mutations avec un terrain très grand, cela est difficilement caractérisable et elles sont donc supprimées lorsque le terrain est supérieur à 5000m². La surface du terrain est également corrélée à la valeur foncière (**Figure 2 c**) mais moins forte que la surface habitable. La distribution du nombre de pièce montre des biens avec aucune pièce principale et d'autres avec un nombre de pièces très important. Les observations sont conservées uniquement lorsque le nombre de pièces est compris entre 1 et 10. De plus, comme on peut s'y attendre, le nombre de pièce est corrélé à la valeur foncière (**Figure 2 f**). Pour finir le nettoyage, on vérifie que les latitudes et les longitudes sont correctement remplies et que les codes communes sont corrects en vérifiant leurs existences dans la base de données sur les communes. Cette base de données sera finalement composée de 79k observations et de 17 variables.

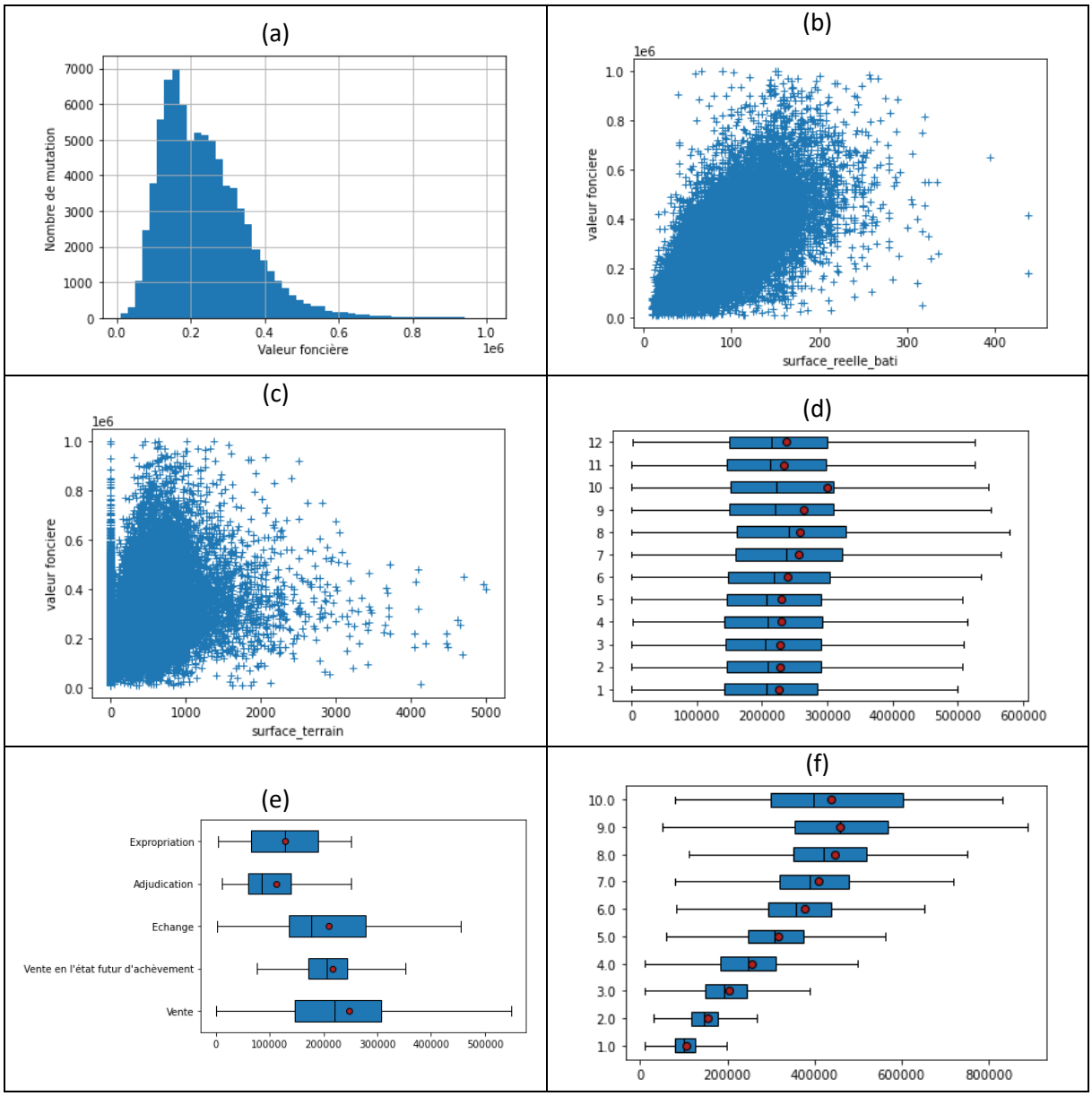

***Figure 2** : Figure caractérisant les mutations immobilières :*
(a) distribution de la valeur foncière ; (b) corrélation entre la surface et la valeur foncière ; (c) corrélation entre la surface du terrain et la valeur foncière ; (d) valeur foncière en fonction du mois de la mutation ; (e) valeur fonction en fonction du type de la mutation ; (f) valeur foncière en fonction du nombre de pièce

2. Données sur les loyers

Les données sur les estimations des loyers pour les maisons et les appartements sont traitées de la même façon. On conserve le code de la commune nommé ici INSEE, le nom de la commune, le département et le loyer moyen, inférieur et supérieur par m². Dans les villes contenant des arrondissements, chaque arrondissement est distinct l'un de l'autre, ils sont alors tous réunis dans une seule observation en prenant la moyenne de chaque variable. Les distributions des loyers minimums maximums et moyens sont représentées sur la **Figure 3** pour les maisons et pour les appartements. Ces données peuvent être utiles pour estimer la valeur foncière d'un bien.

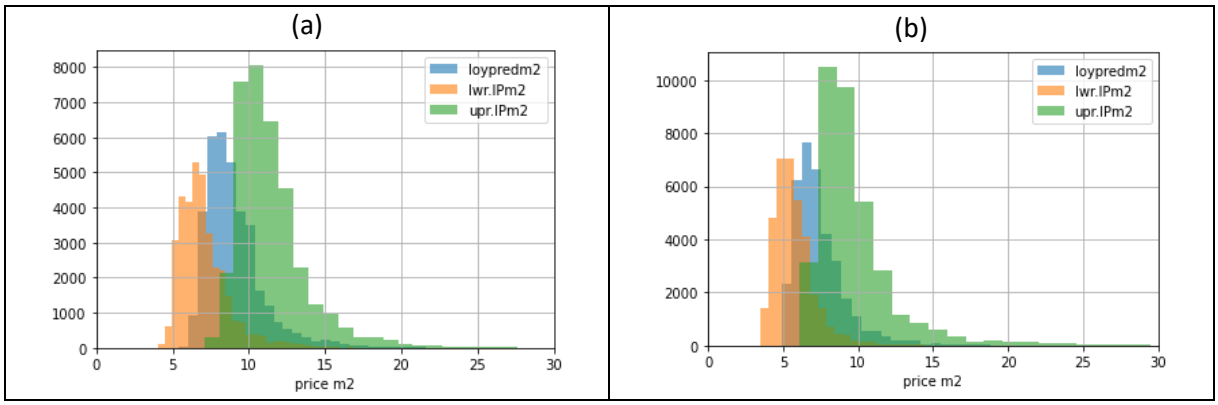

***Figure 3** : Caractérisation de l'estimation du loyer :*
(a) pour les appartements ; (b) pour les maisons

3. Education

La base de données comportant la liste de toutes les écoles de France contient beaucoup d'informations les caractérisant. Parmi celles-ci, seules quelques-unes sont conservées car pouvant être utiles : le secteur d'activité (public / privé), la localisation (coordonnées, commune, code postal), l'état de l'établissement (ouvert ou non) et le niveau d'étude. Pour réduire le nombre d'observations les écoles en dehors du département étudié sont supprimées ainsi que celles qui sont fermées. Dans un but de simplification il est décidé de regrouper les niveaux d'études en quatre niveaux : école maternelle, élémentaire, collège et lycée. La répartition entre les différents niveaux est représentée sur la **Figure 4**. Pour finir, les valeurs des coordonnées sont vérifiées afin de ne pas contenir de valeurs aberrantes. Cette base de données permettra de calculer la distance de chaque logement à un établissement scolaire, ce qui peut avoir un impact sur la valeur foncière.

Le traitement de la base de données contenant les informations sur les universités est similaire. Les variables conservées caractérisent le secteur d'activité, le type d'établissement, la localisation et le nombre d'inscrits sur les dernières années. Cette information pourra être utile pour connaitre le nombre d'étudiants cherchant un logement à proximité. On supprime les établissements dont la localisation n'est pas renseignée. Le nombre d'inscrits moyen entre 2010 et 2017 est calculé afin de remplacer la valeur pour chaque année et donc simplifier la base de données. La répartition du type d'établissement est représentée sur la **Figure 4**.

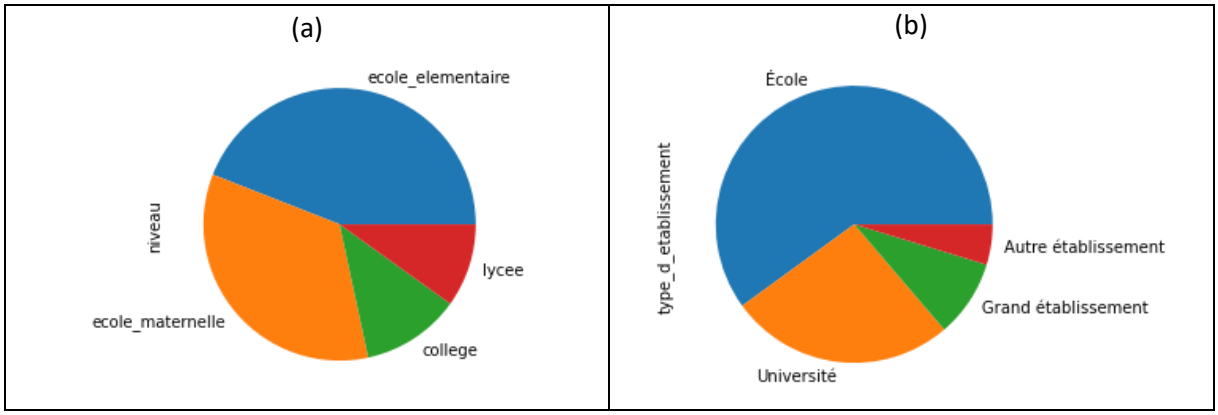

***Figure 4**: Caractérisation des établissements scolaires :*
(a) répartition du nombre d'école en fonction du niveau ; (b) répartition des établissements d'enseignement supérieur en fonction du type d'établissement

4. Transports en communs

La base de données sur les transports caractérise l'ensemble des stations de transports en communs en Ile de France. Ces informations seront utilisées pour calculer la distance aux transports du logement,

et pour savoir de quel type de transport il est le plus prêt. Parmi la grande quantité d'informations, seules les variables contenant le nom de la station, le mode de transport et la localisation sont conservés. On vérifie que chaque observation est correctement renseignée en ne contenant pas de valeur nulle, et que la localisation ne soit pas aberrante. La répartition du type de transport en commun est représentée sur la **Figure 5**.

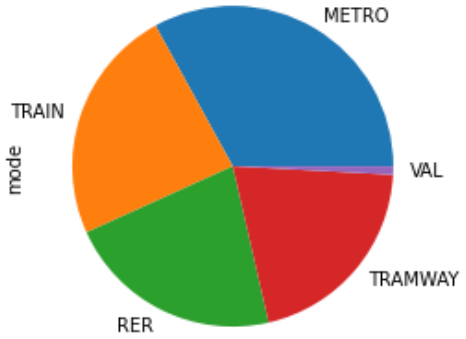

***Figure 5** : représentation de la répartition du type de transport en commun*

5. Données sur les communes

La base de données sur les communes contient environ 36k observations et 101 variables. Néanmoins certaines sont trop peu renseignées et sont donc supprimées pour conserver uniquement des informations utiles à notre étude. Cela permettra de caractériser les communes, ce qui peut être intéressant pour estimer la valeur foncière d'un bien. On conserve donc le code de la commune, le nom de la commune, le département, les informations sur l'accès aux soins, les informations démographiques ainsi que le taux de propriété. Pour réduire la taille de la base de données, les villes en dehors de l'Essonne sont supprimées. Parmi les variables conservées, le nombre de pharmacies semble erroné puisqu'il semble peu probable qu'il n'y ai jamais plus de deux pharmacies par communes : elle est donc supprimée. De même le taux d'étudiant ne semble pas être discriminant, et est donc supprimé également. La **Figure 6** représente des exemples de distributions caractérisant les communes.

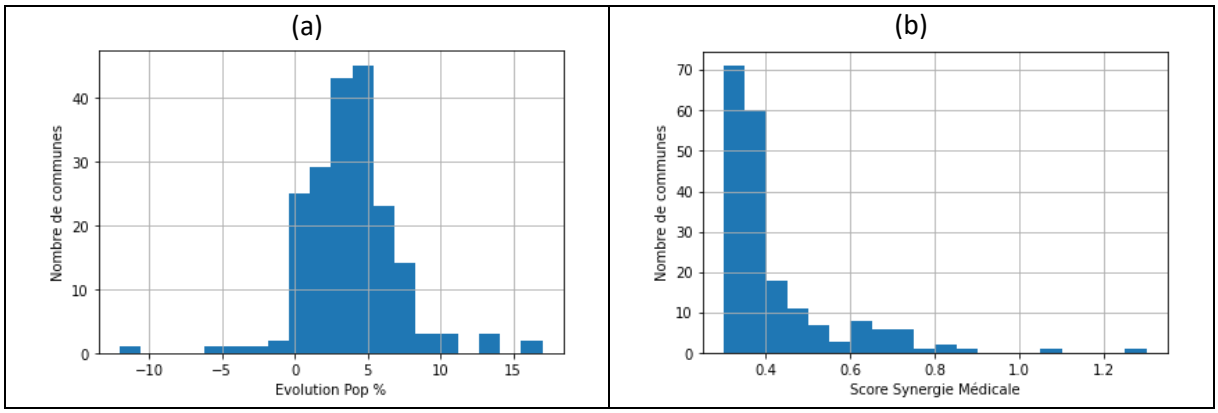

***Figure 6** : caractérisation des communes :*
(a) évolution de la population ; (b) score de synergie médicale

3.3 Création de nouvelles variables

1. Combinaison des bases de données

L'ensemble des bases de données est combiné pour former un jeu d'apprentissage. Pour cela les données sur les mutations sont combinées avec celles sur les communes et les loyers grâce aux codes

communes. Grâce à cela il est possible de calculer le loyer estimé pour chaque mutation. Les loyers normalisés par m² sont ensuite supprimés.

Les distances entre des lieux d'intérêts et les biens peuvent être calculées grâce à la librairie sklearn haversine_distances (en multipliant le résultat par le rayon terrestre). Grâce à cette fonction les distances entre chaque bien et chaque niveau scolaire, ainsi qu'entre chaque bien et le transport en commun le plus proche sont calculées. On récupère également le mode de transport le plus proche et le nombre d'établissement d'enseignement supérieur et le nombre d'étudiants dans un rayon de 10km.

3.4 Importance des features

La base de données finale contient 23 variables, mais toutes les variables ne sont pas forcément utiles pour la prédiction. Il peut être intéressant pour améliorer la performance de ne conserver que des variables importantes. Une première méthode est de calculer les coefficients de corrélation entre chaque variable. Par cette méthode, la valeur foncière est corrélée à la surface habitable et du terrain, aux nombres de pièces, aux différentes informations sur la commune et au type de logement. On peut également utiliser d'autres méthodes que le f_score, la régularisation Lasso (L1), les arbres de décisions ou les méthodes ensemblistes. Des conditions sont fixées arbitrairement de manière à identifier une dizaine de variable par méthode (F_score et lasso > 1000 et random forest et XGBOOST > 0.01). Les variables conservées sont quant à elles présentes dans la sélection d'au moins 3 des 4 méthodes en y ajoutant le nombre de dépendance et le nombre de pièces principales. Les dix variables conservées sont : latitude, longitude, loyer_moyen, n_etab_sup_10km, nature_mutation_enc, surface_reelle_bati, surface_terrain, type_local_enc, Dépendance, nombre_pieces_principales.

IV. Modélisation

La modélisation a été effectuée à partir de la base de données précédemment créée, après un encodage de chaque variable catégorielle. Pour obtenir uniquement des variables numériques. Différents types de modèles ont été testés, soit avec toutes les variables, soit uniquement avec les variables déterminées comme importantes. Afin d'essayer d'améliorer la performance des modèles, une méthode de moyennage des prédictions a également été utilisée. L'ensemble des résultats est résumé dans les **Tableau 1** et **Tableau 2** et représenté sur la **Figure 7**. Chaque méthode sera détaillée dans les parties suivantes.

Model	MAE train set	MAE test set	Adjusted R^2 train set	Adjusted R^2 test set
XGBoost	28113	32393	0.875	0.809
XGboost opt	22506	31864	0.923	0.808
Cateboost	24972	30711	0.872	0.817
NN_l-10_n-128	37254	36755	0.777	0.772
NN_l-15_n-256	37138	36896	0.782	0.774
TabMLP	39778	39247	0.751	0.748
TabResNet	36623	36304	0.784	0.780
TabTransformers	35808	35480	0.787	0.783
Model Averaging	29824	32126	0.851	0.811

***Tableau 1** : Résultats de la modélisation avec l'ensemble des variables.*

Model	MAE train set	MAE test set	Adjusted R^2 train set	Adjusted R^2 test set
XGBoost	31160	34403	0.849	0.791
XGboost opt	26217	33863	0.896	0.791
Cateboost	28611	33199	0.848	0.797
NN_l-10_n-128	35589	35626	0.790	0.782
NN_l-15_n-256	46929	47806	0.682	0.646
TabResNet	39691	38997	0.755	0.755
TabTransformers	40303	39798	0.751	0.748
Model Averaging	32083	34199	0.835	0.795

***Tableau 2** : Résultat de la modélisation avec les variables importantes.*

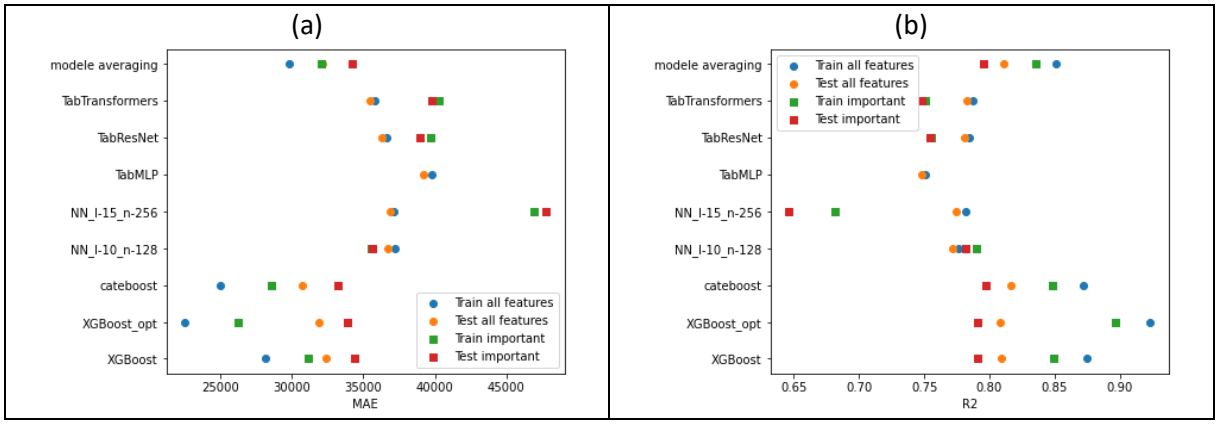

***Figure 7** : Résultat des modélisations :*
(a) Erreur moyenne absolue ; (b) R2 ajusté

4.1 Les modèles « classiques »

Des modèles utilisés régulièrement en ML et très bien connu ont été utilisés comme références. Quatre modèles ont été sélectionnés parmi quatre méthodes différentes : ElasticNet [7] pour les méthodes linéaires, KNearest Neighbors [8] pour les méthodes basées sur la proximité entre observations, Kernel Support Vector Machine [9] pour les méthodes non linéaires et XGBoost [10] et CateBoost [11] pour les méthodes par Gradient Boosting.

Sur ces données complexes, comprenant des données catégorielles et continues, les méthodes de Gradient Boosting permettent d'obtenir les meilleurs résultats avec une erreur moyenne absolue (MAE) de 32,8k et 30,7k pour XGBoost et CateBoost respectivement. Le modèle KNN permet d'obtenir une MAE de 41k, et ElasticNet 47k. Le modèle le moins performant est le Kernel SVM avec une MAE de 91k. Ce modèle apprend à peine mieux qu'un modèle retournant la moyenne de 92k. Les modèles XGBoost et Cateboost forment une bonne référence pour étudier les améliorations qu'il est possible d'apporter.

4.2 Les réseaux de neurones

Des réseaux de neurones ont également été utilisés pour créer une référence puisque les méthodes de TabTransformers sont couplées à un réseau de neurones. Cela permet donc de conclure directement sur les améliorations des transformers sur la performance du réseau du neurones. Même si la bibliothèque des TabTransformers est basée sur Pytorch, les références ont été créées avec Tensorflow [12] par convenance personnelle. Pour ces tests une architecture classique a été utilisée, composée de « l » couches cachées de « n » neurones avec une fonction d'activation ReLU, avec une couche d'entrée correspondant au nombre de variables et une sortie de taille 1 pour la prédiction. Plusieurs couples de valeurs de « n » et » « l » ont été testés pour trouver la meilleure architecture. La meilleure performance est obtenue pour les couples (n,l) = (128, 10) et (256,15). Les MAE obtenues

pour ces modèles sont d'environ 37k€ en utilisant toutes les variables. La performance des réseaux de neurones est inférieure à celle obtenue par les méthodes de boosting, mais les modèles ne présentent pas de surapprentissage.

4.3 Les Transformers

La librairie pytorch-widedeep contenant les transformers de données tabulées [1] a été utilisée dans ce projet. En plus de l'architecture proposée par Huang et al [1], la librairie contient d'autres modèles comme le TabMLP (qui combine un plongement des données catégorielles avec des données continues qui sont ensuite passées dans une série de couches fully-connected), ou le TabResNet (le plongement de données est passé dans une série de bloc ResNet initialement conçu pour le traitement de langage). Ces trois modèles conduisent à des performances légèrement différentes. Le TabMLP est le modèle avec la performance la plus médiocre (MAE de 39k) ce qui est moins bon qu'un réseau de neurones simple. Le TabResNet et le TabTransformers sont quant à eux légèrement plus performants qu'un réseau de neurones simple, avec 36k et 35k de MAE respectivement. Cependant ces performances sont inférieures à ce qui est accessible grâce à des méthodes de boosting (MAE entre 31 et 32k). Cela signifie que les transformers ne permettent pas toujours d'améliorer la précision des modèles et que cela va dépendre du type de données à disposition. Dans notre cas il est probable que les données ne soient pas assez structurées pour espérer un gain significatif, surtout comparé à des méthodes de gradient boosting très efficaces sur ce type de données.

4.4 Modèle averaging

Compte tenu des performances relativement faibles de nos modèles, il peut être intéressant d'appliquer un modele averaging. Cette méthode consiste à moyenner les résultats obtenus à travers différents modèles, de manière similaire aux méthodes ensemblistes, sauf qu'il s'agit d'assembler des modèles forts au lieu de modèles faibles. L'objectif est de réduire l'erreur totale en la moyennant sur plusieurs modèles. Cela est donc basé sur l'hypothèse que l'erreur de chaque modèle est aléatoire et que donc qu'elle peut s'annuler (du moins en partie) en moyennant plusieurs prédictions.

Pour réaliser cette moyenne, seuls les modèles performants sont sélectionnés : soit le XGboost et le XGboost optimisé, le CateBoost, les réseaux de neurones, le TabResNet et le TabTransformers. Cela conduit à une MAE de 30k et 32k sur le jeu d'apprentissage et de test respectivement (en utilisant toutes les variables). Ce résultat est donc meilleur que ceux des modèles de réseaux de neurones ou de transformers, mais est dans le même ordre de grandeur de ce qui est accessible par les méthodes de boosting. Bien que la performance ne soit pas améliorée, cette méthode permet d'affirmer que les erreurs des modèles ne proviennent pas d'erreurs aléatoire, mais que les mêmes erreurs se retrouvent entre toutes les méthodes. Cela signifie vraisemblablement que notre base de données n'est pas assez précise pour espérer de meilleures performances.

4.5 Influence des features importances

La sélection de features a pour but de simplifier le modèle, voire d'espérer d'améliorer les performances. Dans notre cas la sélection de features est basée sur différentes méthodes expliquées dans la partie 3.4. La **Figure 7** permet de comparer les résultats des modèles avec toutes les variables avec les modèles contenant uniquement les variables importantes. Il est indéniable que cette méthode détériore la performance des différents modèles. La détérioration est plus notable sur les méthodes de boosting alors que l'effet négatif est moins important sur les transformers. Il faut donc être vigilant avant d'utiliser cette méthode.

V. Conclusion

La performance des transformers appliquée aux données tabulées a été étudiée dans ce projet en utilisant une base de données réelle issue de données réelles collectées sur le site gouvernemental data.gouv.fr dans le but de prédire les valeurs immobilières en Essonne. La première partie du projet a consisté à créer la base de données en combinant plusieurs et en filtrant les variables qui semble les plus pertinente. De nouvelles variables ont été créées, notamment pour caractériser la distance entre certains lieux d'intérêts et chaque bien. A partir de cette base de données, plusieurs modèles ont été entrainés pour servir de référence. Les meilleures performances sont obtenues à partir d'un modèle CatBoost. Les modèles basés sur les transformers ne permettent pas d'accéder à une précision aussi forte, même s'ils sont plus précis que des réseaux de neurones classiques. Cela est probablement lié au fait que nos données sont faiblement structurées et que les méthodes de gradient boosting sont plus adaptées à ces données que les réseaux de neurones. Les transformers ne permettent donc pas toujours d'améliorer les performances, cela dépendra fortement du type de données à disposition. Pour finir, il est à noter que la performance globale de tous les modèles reste assez faible et une méthode de modèle averaging ne permet de l'améliorer, ce qui semble signifier que les données à disposition ne sont pas assez précises pour obtenir une précision acceptable pour le développement d'une application.

VI. Références bibliographiques

[1] X. Huang, A. Khetan, M. Cvitkovic, et Z. Karnin, « Tabtransformer: Tabular data modeling using contextual embeddings », *ArXiv Prepr. ArXiv201206678*, 2020.

[2] A. Vaswani *et al.*, « Attention is all you need », *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[3] G. A. Aguayo *et al.*, « Comparison of Deep Neural Networks and Regularised Cox Regression Models in the Prediction of Neurodegenerative Diseases in the General Older Population », *Available SSRN 4026085*.

[4] H. Aboutalebi, M. Pavlova, M. J. Shafiee, A. Florea, A. Hryniowski, et A. Wong, « COVID-Net Biochem: An Explainability-driven Framework to Building Machine Learning Models for Predicting Survival and Kidney Injury of COVID-19 Patients from Clinical and Biochemistry Data », *ArXiv Prepr. ArXiv220411210*, 2022.

[5] M. A. Sheikh, G. Z. Khan, et F. K. Hussain, « Systematic Analysis of DDoS Attacks in Blockchain », in *2022 24th International Conference on Advanced Communication Technology (ICACT)*, 2022, p. 132-137.

[6] X. Xu et X. Zheng, « Hybrid Model for Network Anomaly Detection with Gradient Boosting Decision Trees and Tabtransformer », in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, p. 8538-8542. doi: 10.1109/ICASSP39728.2021.9414766.

[7] J. H. Friedman, T. Hastie, et R. Tibshirani, « Regularization Paths for Generalized Linear Models via Coordinate Descent », *J. Stat. Softw.*, vol. 33, n° 1, p. 1-22, févr. 2010, doi: 10.18637/jss.v033.i01.

[8] N. S. Altman, « An introduction to kernel and nearest-neighbor nonparametric regression », *Am. Stat.*, vol. 46, n° 3, p. 175-185, 1992.

[9] K. Crammer et Y. Singer, « On the algorithmic implementation of multiclass kernel-based vector machines », *J. Mach. Learn. Res.*, vol. 2, n° Dec, p. 265-292, 2001.

[10] T. Chen et C. Guestrin, « Xgboost: A scalable tree boosting system », in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, p. 785-794.

[11] A. V. Dorogush, V. Ershov, et A. Gulin, « CatBoost: gradient boosting with categorical features support », *ArXiv Prepr. ArXiv181011363*, 2018.

[12] T. Developers, « TensorFlow ». Zenodo, mai 2022. doi: 10.5281/zenodo.6574269.