1. Analyse globale du DataSet

- Utiliser la bibliothèque pandas pour récupérer les données *airbnb.csv*
- Supprimer les features suivantes :
  'id','description', 'first_review', 'host_has_profile_pic', 'host_identity_verified', 'host_response_rate', 'host_since', 'last_review', 'latitude', 'longitude', 'name','number_of_reviews', 'review_scores_rating', 'thumbnail_url', 'zipcode'
- Afficher la taille du dataSet, le nombre et le type des features.
- Afficher les doublons, puis supprimez-les.
- Combien y-a-t-il de valeurs manquantes dans chaque features. Afficher sous forme de dataFrame uniquement les features qui contiennent des valeurs manquantes, sous forme triées.

2. Analyse de la target

- Affichez sous forme d'histogramme les prix des locations. Vous utiliserez 30 bacs.
- Affichez maintenant toujours sous la forme d'un histogramme les vrais prix avec le même nombre de bacs. Commentez les deux courbes.
- Transformer la colonne *log_price* en une colonne *price*.
- Combien y a-t-il de location avec un prix de moins de 10$, puis supprimez-les.
- Combien de locations coutent plus de 1000$.
- Affichez les valeurs moyennes sur les features qui ne sont pas de type 'objet' pour les locations de plus de 1000$ et celles de moins de 200$. Que constate-t-on, que peut-on supposer.

3. Analyse du type de propriété

- Combien y-a-t-il de types de propriétés différentes.
- Affichez le nombre de location et le prix moyen de ces locations en fonction de leur type *property_type*.
- On souhaite pouvoir réduire le nombre de type de propriétés, en particulier ceux pour lesquelles il y a moins de 50 samples. Récupérer

la liste des types de locations avec moins de 50 samples (*Lmoins50*) et celle de plus de 50 samples (*Lplus50*).

- Pour cela on peut soit supprimer toutes ces samples, soit transformer les locations peut fréquente en d'autres types de locations plus fréquentes. On cherche donc location Lmoins50 quel est le type de location de Lplus50 le plus proche. On utilisera la bibliothèque *spacy* et le modèle pré-entrainé *en_core_web_md*:

```
!python -m spacy download en_core_web_md      #Récupération du modèle
nlp = spacy.load("en_core_web_md")            #Chargement du modèle pré-entrainé
nlp(w1).similarity(nlp(w2))                   #Calcul de la similarité entre w1 et w2
```

- Calculer la meilleure similarité entre les mots de *Lmoins50* et celle de *Lplus50*.
- Transformer les locations de par celles de si la similarité est supérieure à 0.5, supprimez les autres.
- On considère qu'une valeur est un outlier si sa valeur est superieure à Q3+1.5(IQR) soit Q3+1.5(Q3-Q1). Affichez le nombre d'outliers pour chaque type de location.
- Les types de location ne peuvent pas être comparés entre eux il est nécessaire de les coder en utilisant une colonne binaire pour chacun. Vous effectuerez cette opération (on pourra reporter cette opération en à la toute fin de programme).

4. Analyse des types de chambre

- Afficher le nombre et les différentes moyennes des différents types de chambre, vous n'afficherez que les locations à moins de 500 dollars. Que constatez-vous ?
- Afficher avec un *boxplot* les prix en fonction du type des chambres.
- Au regard de ces premières informations est-il nécessaire de créer une *feature* pour chaque type de chambre, ou une seule colonne suffit.
- Affichez la variation des prix des locations avec la fonction *histplot*.
- Si l'on souhaite conserver une seule colonne, peut-on les numéroter dans un ordre aléatoire.
- Sachant que la plupart des algorithmes d'apprentissage utilisent la norme euclidienne pour comparer les samples entre eux, quel serait la meilleure façon de représenter ces différents types.

- Affichez via un *hisplot* l'évolution des prix en fonction des chambres.
- Transformez les données par des colonnes binaires. Vous supprimerez explicitement la colonne *Shared room*. Les types de location ne peuvent pas être comparés entre eux il est nécessaire de les coder en utilisant une colonne binaire pour chacun. Vous effectuerez cette opération (on pourra reporter cette opération en à la toute fin de programme).

5. Analyse de la feature accomodates

- Quelles sont les différentes valeurs de la feature *accommodate,* quel est le nombre de samples correspondant et la valeur moyenne des locations.
- Afficherez ces valeurs triées sur la valeur moyenne. Que constate-t-on.
- La plupart des valeurs sont des valeurs binaires. Il peut être utile de normaliser les données pour des modèles d'apprentissage. Utiliser le *MinMaxScaler* pour transformer les données.

6. Analyse de la feature bathrooms

- Affichez le prix moyen des locations en fonction du nombre de chambe.
- Que constate-t-on en particulier pour le nombre de *bathrooms* égal à 8.
- Affichez pour les locations avec 8 salles de bains le nombre de location et les prix moyen en fonction du type de chambre. Attention la colonne a été converties en colonnes binaires et la colonne *Shared room* a été supprimée. (on pourra reporter cette opération en à la toute fin de programme).
- Quelle est le prix moyen des valeurs manquantes. A quelle moyenne ressemble-t-il le plus.
- Remplacer ces valeurs manquantes par le nombre de salle de bains qui ont la moyenne la plus proche.

7. Analyse de la feature bed_type

- Affichez le prix moyen des locations en fonction du type de lit.
- On souhaite comparer les prix moyens en fonction du prix moyen des locations avec de vrais lits *Real Bed*. Que constatez-vous sur les prix et sur le nombre de types de lits au regard du nombre et du prix des locations avec de vrais lits.

- Regrouper toutes les locations en deux catégories les *Real Bed* et les autres.

8. Analyse de la cancelation_policy

- Affichez le prix moyen des locations en fonction du type d'annulation.
- Pensez-vous que l'on puisse regrouper les données comme pour le cas précédent. Est-il nécessaire de créer pour chaque type d'annulation une colonne spécifique.
- Transformer les données.
- Normalisé cette feature.

9. Analyse de la cleaning_fee

- Afficher la via un histogramme la répartition des prix de chacune de ces catégories. Que constatez-vous, est-il judicieux de conserver cette *feature* ?
- Supprimez cette feature.
- Effectuer le même travail sur la feature *instant_bookable*.

10. Analyse de la variable bedrooms

- Afficher pour chaque nombre de lits le prix moyen des locations.
- Rechercher et afficher le nombre de valeurs manquantes ainsi que le prix moyen de ces locations.
- Que constatez-vous pour les locations avec 0 lits. Pensez-vous que cette valeur peut être modifiée. Modifier ces valeurs.
- Quel est le prix moyen des locations lorsque les valeurs sont manquantes.
- Transformer les valeurs manquantes.
- Normaliser les valeurs.

11. Analyse de la variable beds

- Effectuer le même travail que pour la variable beedrooms.

12. Analyse de la variable city

- Quelle est la répartition des locations et des prix moyens par villes.

- Afficher visa un histplot la répartition des prix des locations en fonction des villes. Que constatez-vous.
- Utilisez une colonne binaire pour chaque ville. (on pourra reporter cette opération en à la toute fin de programme).

13. Influence des features

- Effectuer les one-hot encoding.
- Sur une copie de la base supprimer les features non numérique et calculer les corrélations.
- Que constatez-vous, quelles sont les features qui semblent avoir le plus d'importance dans le prix des locations.

14. Traitement des quartiers

- Combien y a-t-il de type de quartiers différents.
- Quelle est la répartition des prix sur les valeurs manquantes.
- Quelle est la répartition des prix sur les autres valeurs.
- Comparer ces deux courbes en vous restreignant aux prix inférieurs à 500$. Que constatez-vous.
- Supprimez toutes les valeurs manquantes.
- Calculer le prix moyen de chaque quartier est affichez les répartitions avec la fonction histplot.
- Compte tenu du nombre important de quartiers il n'est pas possible de créer une colonne pour chaque quartier. L'idée est donc de regrouper les quartiers en bacs en fonction des prix moyens. Créer un DataFrame qui contient le nombre de locations dans un quartier et le prix moyen.
- Ajouter une colonne correspondant à un découpage en 50 bacs. Puis affichez le nombre par bacs. Que constatez-vous.
- Transformer tous les bacs supérieurs à 10 en un seul bac (10).
- Transformez les quartiers en fonction de ce découpage. Puis normalisez les valeurs.

15. Traitement de la feature amenities

L'encodage de la variable amenities nécessite plusieurs étapes.

- Supprimer tous les caractères inutiles tels que ", {, }.

  - Identifier les différents aménagements disponibles.
  - Corriger les erreurs et modifier les aménagements mal orthographiés
  - Créer des variables pour chaque aménagement, et les compléter avec les données de la feature amenities
  - Supprimer les aménagements trop peu présents dans les locations.
  - Enfin, il peut être nécessaire de réduire sa dimensionnalité.

- Les données contenues dans la feature amenities se présentent sous la forme :

  '{"Wireless Internet","Air conditioning",Kitchen,Heating,"Family/kid friendly",Essentials,"Hair dryer",Iron,"translation missing: en.hosting_amenity_50"}

- Supprimer tous les caractères { " } de amenities.
- Identifier toutes les commodités sous la forme d'un ensemble.
- Transformer toutes les commodités mal orthographiées. Pour cela on utilisera les deux listes suivantes

```
to_replace_ = list(['Doorman Entry', 'Elevator in building' , 'Internet' , 'Firm matress',
      'Other pet(s)', 'Smart lock', 'Wide clearance to bed', 'Wide clearance to shower & toilet',
            'Wide clearance to shower and toilet',  'translation missing: en.hosting_amenity_49',
       'translation missing: en.hosting_amenity_50', 'Waterfront'])
replace_ = list(['Doorman', 'Elevator', 'Ethernet connection' , 'Firm mattress', 'Pets allowed',
        'Smartlock' , 'Wide clearance', 'Wide clearance', 'Wide clearance', 'translation missing',
        'translation missing', 'Beachfront'])
```

- Créer maintenant une colonne pour chaque commodité.