DATA CAMP - M2 SEP

Version 2023/24

Table des Matières

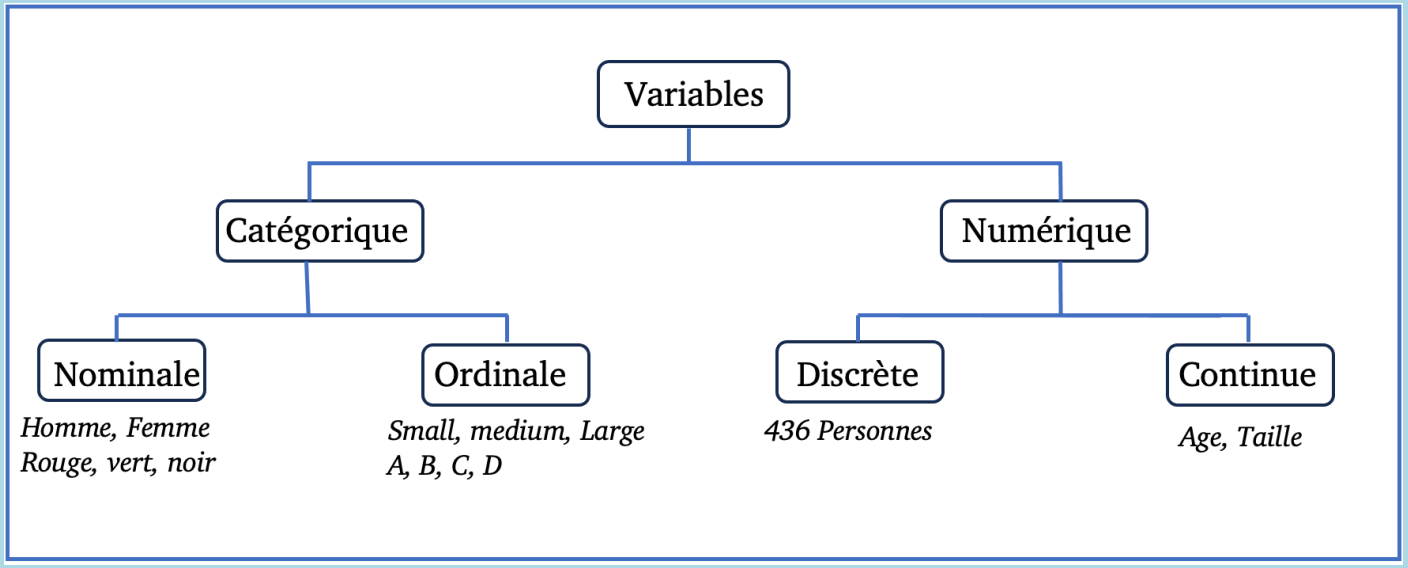
Partie 1 : Analyse de données

Rappels Fondamentaux

Qu'est ce qu'une variable ?

Une variable statistique est une caractéristique ou une mesure que l'on peut attribuer à chaque individu d'une population ou d'un échantillon.

Tableau du type de variable :



Vocabulaire autour de la variable :

Lors d'une analyse statistique, on distingue deux types principaux de variables :

- La **variable à expliquer** (à prédir, ou à estimer) : **Y**
- Les **variables explicatives** (prédictives ou estimatrices) : **X_i**

Tout l'exercice reside à établir la relation la plus pertinente entre les variables explicatives **X_i** et la variable à expliquer **Y**.

Tableau des synonymes :

Catégorique = Qualitative.
Numérique = Quantitative.

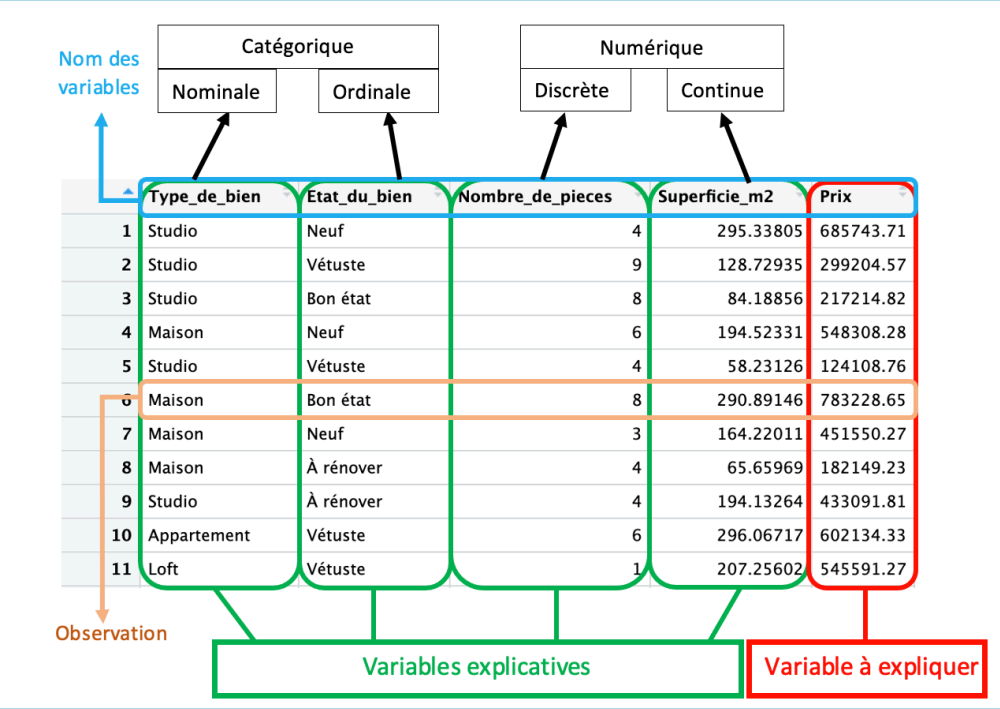
Tableau autour de la variable	
Xi	Y
Explicative	à expliquer
Exogène	Endogène
Indépendante	Dépendante
	Réponse (Machine Learning)

Exemple concret :

Je cherche à estimer le prix d'une maison en fonction de 4 caractéristiques : le type de bien, l'état du bien, le nombre de pièces et la superficie en m2.

Pour ce faire, j'ai à ma disposition une base de données intitulée "data_immobilier" (générée aléatoirement) que je traite en R.

Avant tout, je dois avoir ces éléments en tête :



	Type_de_bien	Etat_du_bien	Nombre_de_pieces	Superficie_m2	Prix
1	Studio	Neuf	4	295.33805	685743.71
2	Studio	Vétuste	9	128.72935	299204.57
3	Studio	Bon état	8	84.18856	217214.82
4	Maison	Neuf	6	194.52331	548308.28
5	Studio	Vétuste	4	58.23126	124108.76
6	Maison	Bon état	8	290.89146	783228.65
7	Maison	Neuf	3	164.22011	451550.27
8	Maison	À rénover	4	65.65969	182149.23
9	Studio	À rénover	4	194.13264	433091.81
10	Appartement	Vétuste	6	296.06717	602134.33
11	Loft	Vétuste	1	207.25602	545591.27

Ensuite, je peux passer à l'étape suivante.

Traitement des données

Peu importe leur origine, les bases de données nécessitent presque toujours un traitement avant d'être exploitables. De la collecte à l'enregistrement, des irrégularités s'introduisent, causant des erreurs lors de l'exploitation. Le traitement vise à corriger ou à gérer ces irrégularités. Bien que chaque base de données présente ses propres spécificités et problèmes, les quatre points que nous aborderons par la suite sont courants et doivent être maîtrisés

Gestion des valeurs manquantes

Les valeurs manquantes, souvent représentées par les `NaN`, sont parmi les anomalies les plus couramment rencontrées dans les bases de données. Plusieurs stratégies peuvent être adoptées face à ces valeurs :

Conversion : On surpasse le `NaN` en le convertissant en une autre valeur, comme un flottant.

- **Avantage** : Conservation de données.
- **Désavantage** : Introduction d'un biais potentiellement significatif.

Code :

```r
# Convertir les NaN en 0 (ou une autre valeur) en R.
df[is.na(df)] <- 0
```

```python
# Convertir les NaN en 0 (ou une autre valeur) en python.
df.fillna(0, inplace=True)
```

imputation : remplacer les valeurs manquantes par des estimations (la moyenne ou la médiane des autres valeurs).

- **Avantage** : Conservation de données.
- **Désavantage** : Introduction d'un biais potentiellement significatif.

Code :

```r
## Utiliser la moyenne pour imputer en R
df$column_name[is.na(df$column_name)] <- mean(df$column_name, na.rm = TRUE)
```

```python
# Utiliser la moyenne pour imputer en python
df['column_name'].fillna(df['column_name'].mean(), inplace=True)
```

suppression : supprimer les lignes avec des NaN.

- **Avantage** : On limite le biais .
- **Désavantage** : perte d'information générale.

Code :

```r
# Supprimer les lignes contenant des NaN en R
df <- df[complete.cases(df), ]
```

```python
# Supprimer les lignes contenant des NaN
df.dropna(inplace=True)
```

Il n'y a pas de solution parfaite, certains cas vont favoriser certains choix, mais la suppression reste quand même la plus sur en terme de qualité de l'information, à privilègier des que possible.

Group_by

La fonction `group_by` est utilisé pour regrouper des données en fonction de certaines variables catégorielles. Cela permet d'effectuer des opérations et des analyses spécifiques à chaque groupe, et comprendre les comportements ou les anomalies spécifiques à chaque groupe.
Concretement, si l'on reprend la base `data_immobilier`, faire une group_by sur la variable `Type_de_bien` nous permettra de faire une etude sur le groupe `maison`, `loft`, `appartement` et `studio` séparemment.

R :

```r
# Groupement des données par Type_de_bien et calcul de la moyenne des autres variables
data_grouped <- data_immobilier %>%                                    #Nouveau DataFrame "data_grouped"
  group_by(Type_de_bien) %>%                                           #Selection par groupe "Type de bien"
  summarise(                                                           #Affiche les infomations
    Prix_moyen = mean(Prix, na.rm = TRUE),                             #La moyenne de prix pour chaque groupe
    Superficie_moyenne = mean(Superficie_m2, na.rm = TRUE),            #La moyenne de superficie pour chaque groupe
    Nombre_moyen_de_pieces = mean(Nombre_de_pieces, na.rm = TRUE)      #La moyenne de Nombre de piece pour chaque groupe
  )
print(data_grouped)                                                    #Affichage
```

Python :

```python
# Groupement des données par Type_de_bien et calcul de la moyenne des autres variables
data_grouped = data_immobilier.groupby('Type_de_bien').agg({
    'Prix': 'mean',
    'Superficie_m2': 'mean',
    'Nombre_de_pieces': 'mean'
}).reset_index()
print(data_grouped)
```

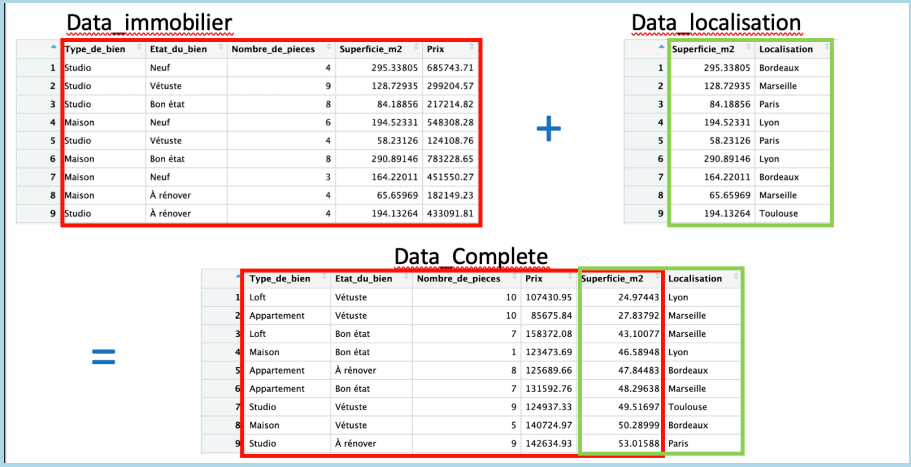
Merge

La fonction merge est utilisée pour fusionner deux dataframes sur la base d'au moins une colonne commune. Cette fonction est extremement utile si l'on souhaite combiner des données provennant de differentes bases pour une même analyse.

Application concrète avec la base de données `data_immobilier`:
On a une autre base de données, `data_Localisation`, avec les variables `Superficie_m2` et `Localisation`. En utilisant la fonction merge, on va fusionner les deux bases de données en utilisant la colonne commune, `Superficie_m2` , pour avoir une base de données plus riche.

```r
# Exemple de code R utilisant merge
data_complete <- merge(data_immobilier, data_Localisation, by = "Superficie_m2", all = TRUE)
```

```python
# Exemple de code Python utilisant merge
data_complete = pd.merge(data_immobilier, data_Localisation, on='Superficie_m2', how='outer')
```



	Type_de_bien	Etat_du_bien	Nombre_de_pieces	Superficie_m2	Prix
1	Studio	Neuf	4	295.33805	685743.71
2	Studio	Vétuste	9	128.72935	299204.57
3	Studio	Bon état	8	84.18856	217214.82
4	Maison	Neuf	6	194.52331	548308.28
5	Studio	Vétuste	4	58.23126	124108.76
6	Maison	Bon état	8	290.89146	783228.65
7	Maison	Neuf	3	164.22011	451550.27
8	Maison	À rénover	4	65.65969	182149.23
9	Studio	À rénover	4	194.13264	433091.81

	Superficie_m2	Localisation
1	295.33805	Bordeaux
2	128.72935	Marseille
3	84.18856	Paris
4	194.52331	Lyon
5	58.23126	Paris
6	290.89146	Lyon
7	164.22011	Bordeaux
8	65.65969	Marseille
9	194.13264	Toulouse

	Type_de_bien	Etat_du_bien	Nombre_de_pieces	Prix	Superficie_m2	Localisation
1	Loft	Vétuste	10	107430.95	24.97443	Lyon
2	Appartement	Vétuste	10	85675.84	27.83792	Marseille
3	Loft	Bon état	7	158372.08	43.10077	Marseille
4	Maison	Bon état	1	123473.69	46.58948	Lyon
5	Appartement	À rénover	8	125689.66	47.84483	Bordeaux
6	Appartement	Bon état	7	131592.76	48.29638	Marseille
7	Studio	Vétuste	9	124937.33	49.51697	Toulouse
8	Maison	Vétuste	5	140724.97	50.28999	Bordeaux
9	Studio	À rénover	9	142634.93	53.01588	Paris

variables dummys

Les variables dummy sont utilisées pour convertir des variables catégorielles en variables numériques.
Objectifs : Utiliser ces variables dans des analyses statistiques et des modèles de machine learning qui requièrent des entrées numériques. Comment créer des variables dummy ? Exemple concret :

On reprend le dataframe qui a une desormais colonne catégorielle nommée `Localisation` avec des noms de villes. On crée les variables dummy, soit pour chaque catégorie unique dans la colonne `Localisation` deviendra une nouvelle colonne dans le dataframe.

```r
# En R
data_with_dummies <- model.matrix(~ Localisation - 1, data = data_complete) %>%
                     as.data.frame()
```

```python
# En python
data_with_dummies = pd.get_dummies(data_complete, columns=['Localisation'], prefix='', prefix_sep='')
```

Dans le nouveau dataframe, `data_with_dummies`, chaque ville unique de la colonne `Localisation` originale a été transformée en une nouvelle colonne. Si une observation dans la colonne `Localisation` originale était “Paris”, alors la colonne `LocalisationParis` serait marquée avec un `1`, et toutes les autres colonnes de ville seraient marquées avec des `0`.

LocalisationBordeaux	LocalisationLyon	LocalisationMarseille	LocalisationParis	LocalisationToulouse
0	1	0	0	0
0	0	1	0	0
0	0	1	0	0
0	1	0	0	0
1	0	0	0	0
0	0	1	0	0
0	0	0	0	1
1	0	0	0	0
0	0	0	1	0

Attention aux problémes de multicolinéarité, supprimer une colonne pour une utilisation dans certains modèles.

Statistiques Descriptives

Les statistiques descriptives permettent de résumer, décrire et comprendre les données.

Pour les variables `quantitatives`, on utilise :

`Moyenne` : La valeur moyenne.
`Médiane` : La valeur centrale.
`Mode` : La valeur la plus fréquente.
`Min,Max`= valeur minimal et maximum. `Écart Type` : Mesure de la dispersion des valeurs.

```python
# En python
describe(data_quantitative)
```

```r
# En R
summary(data_quantitative)
```

Pour les variables `qualitatives`, on utilise:

`Fréquences` : Nombre de fois qu'une catégorie apparaît.
`Pourcentages` : Proportion d'une catégorie par rapport au total.

```python
# En python
# Calculer les fréquences
freq = data_qualitative.value_counts()
# Calculer les pourcentages
perce = frequencies / len(data_qualitative) * 100
```

```r
# En R
# Calculer les fréquences
frequencies <- table(data_qualitative)
# Calculer les pourcentages
percentages <- prop.table(frequencies) * 100
```

On illustre également par des **graphiques** descriptifs:

Histogrammes et Diagrammes en Barres : Pour montrer la distribution des données.
Boîtes à Moustaches (Boxplots) : Pour montrer la médiane, quartiles et valeurs aberrantes.

```python
# En python
# Créer un histogramme pour les données quantitatives
plt.hist(data_quantitative, bins=5, edgecolor='k')
plt.xlabel("Valeurs")
plt.ylabel("Fréquence")
plt.title("Histogramme des données quantitatives")
plt.show()
```

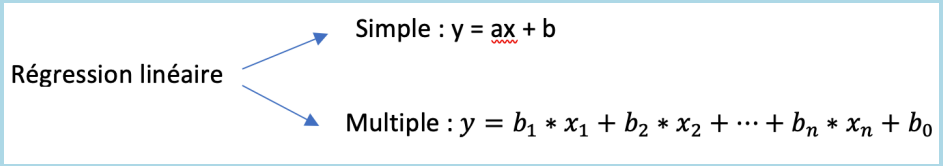
```r
# En R
# Créer un histogramme pour les données quantitatives
ggplot(data = data.frame(x = data_quantitative)) +
  geom_histogram(aes(x = x), binwidth = 5, fill = "blue", color = "black") +
  labs(x = "Valeurs", y = "Fréquence", title = "Histogramme des données quantitatives")
```

La statistique descriptive offre une première compréhension des données, indispensable avant toute analyse plus approfondie. Elle permet de déceler des tendances, anomalies, ou relations à explorer davantage. C'est une étape indispensable à ne surtout pas negliger.

Premiers modèles

Modèles linéaires

Def regression : Un ensemble de méthodes pour analyser la relation d'une variable à expliquer par une ou plusieurs autres variables explicatives.



Méthode des moindres carrés ordinaires :

Objectif : Regarder à quel point la droite obtenue est meilleure que la droite de la moyenne des observations. Donc, $R^2 = \min\left(\sum(y - \hat{y})^2\right)$

Moindres carrés ajustés :

Le R^2 carré ordinaire peut seulement augmenter ou rester constant par l'ajout d'une nouvelle variable. Concretement, ne réduira jamais la capacité explicative du modèle. Le adjusted - R^2 est une methode par pénalisation qui compense ce défaut. Il faut donc le privilègier à l'etude.

P-value :

C'est un indicateur statistique utilisé pour évaluer la significativité d'un résultat. On pose que l'hypothèse nulle (généralement l'absence d'effet ou de relation) est vraie. Une p-value faible suggère que les observations sont peu probables sous l'hypothèse nulle, indiquant ainsi une évidence forte contre l'hypothèse nulle et en faveur de l'hypothèse alternative, donc qu'il y a probablement une relation.

Une p-value inférieure à un seuil défini (souvent 0,05) est généralement interprétée comme statistiquement significative, suggérant que le modèle ou la variable examinée a un effet réel et non dû au hasard.

Concrètement,

- p-value $< 5\%$, la variable est statistiquement significative, donc probablement explicative, on la garde dans le modèle.
- p-value $> 5\%$, la variable n'est pas statistiquement significative, donc probablement pas explicative, on l'exclut du modèle.

Gestion des outliers :

Certaines observations extrêmes, ou outliers, peuvent fausser les résultats en raison de leur décalage significatif par rapport au reste des données. Il faut les identifier et, si nécessaire, les exclure en amont de l'analyse. Pour ce faire, on utilise souvent des méthodes telles que DFFITS et DFBETAS. Ces techniques aident à déterminer si une observation individuelle est particulièrement influente et si elle devrait être considérée comme un outlier devant être exclu de l'analyse pour obtenir des résultats plus fiables.

Multicollinéarité :

La multicollinéarité dans les modèles de régression linéaire est un phénomène où deux ou plusieurs variables explicatives sont fortement corrélées entre elles. Cette forte corrélation peut causer des problèmes dans l'estimation des coefficients de régression, rendant les résultats moins fiables. Pour mesurer le degré de multicollinéarité dans un modèle, on utilise souvent l'indice VIF.

- VIF < 5, peu de risque de multicollinéarité .
- VIF > 5, risque de multicollinéarité.

Méthode Backward élimination pour affiner la précision du modèle :

Initialement, on a une variable à expliquer et plusieurs variables explicatives.

On cherche la meilleure combinaison possible entre ces variables pour definir ce modèle, du point de vue de la significativité à travers la p-value, et de l'explicativité à travers le adjusted- R^2.

Une idée serait de tester toutes les combinaisons une à une, en théorie cela fonctionnerait, mais cela n'est pas réaliste au vu de la quantité de calcul nécessaire.

la méthode Backward élimination consiste alors à évaluer le modèle par itération suivant le schéma suivant :

- 1 - Je fais mon modèle avec l'ensemble des variables.
- 2 - Je supprime LA variable qui à la p-value $> 5\%$, la plus élevé.
- 3 - Je réévalue le modèle
- 4 - Je regarde comment le adjusted-R^2 à évoluer :
  - Il augmente, c'est bien le modèle est plus explicatif
  - Il diminue, c'est mauvais, la variable avait quand même une importance
    - Je refléchis à la conserver ou non si la p-value est pas trop haute.
- 5 - Je répète les étapes 2,3 et 4 autant de fois que nécessaire.

Codes :

Régression linéaire simple :

```
#var1 -> a expliquer
#var2 -> explicative

Reg1<-lm(var1~var2, data=df)
summary(Reg1) # Le détail
abline(Reg1) #Representation de la regression linéaire
```

Régression linéaire multiple :

```
#var1 -> a expliquer

Reg1<-lm(var1~var2+var3+var4, data=df)
```

```
summary(Reg1) # Le détail
abline(Reg1)  #Representation de la regression linéaire
```

Regression logistique

Test d'independance

Récapitulons

Comment faire une analyse rigoureuse ?

1 - Identifier les variables leurs types et leurs interet dans l'étude.
2 - traitement de la base pour la rendre exploitable.
3 - Les statistiques descriptives pour une premieres perceptions.
4 - Etude à proprement parler.

Partie 2 : Complément de cours

Mathématiques

Economie