TP 2

Démarrer un projet de zéro : importer, visualiser, modéliser

TUELEAU Tom

5 septembre 2025

Table des matières

1 Objectifs pédagogiques

— Savoir créer l'ossature d'un projet (dossiers, `venv`, `requirements.txt`, notebook).
— Importer un jeu de données depuis différentes sources : URL publique, fichier local, jeu intégré.
— Explorer et **visualiser** un dataset de façon systématique.
— Entraîner un *premier* modèle de machine learning (classification) pour introduire les concepts.

2 Prérequis

— Python 3.11+ installé (avec `pip`).
— VS Code (extensions Python et Jupyter) ou équivalent.
— Connaissances de base sur les répertoires/fichiers sous votre OS.

3 Étape 1 — Créer l'ossature du projet

Listing 1 Initialiser le projet (Windows PowerShell / Linux / macOS)

```
# 1) Cr er un dossier de travail
mkdir tp2_projet_de_zero && cd tp2_projet_de_zero

# 2) Environnement virtuel
python -m venv .venv
# Activer :  .\.venv\Scripts\activate      # Windows
#            source .venv/bin/activate       # Linux/macOS

# 3) D pendances minimales
pip install --upgrade pip
pip install pandas matplotlib scikit-learn jupyter

# 4) Figer l'environnement (optionnel)
pip freeze > requirements.txt

# 5) Arborescence conseill e
# tp2_projet_de_zero/
#         data/
#         notebooks/
#         src/
```

4 Étape 2 — Importer des jeux de données

Nous proposons trois approches complémentaires. Choisissez **Option A** si vous avez Internet, sinon **B**. L'**Option C** couvre des fichiers locaux courants.

Option A — Depuis une URL publique (Penguins)

Listing 2 Importer via URL (pandas)

```
import pandas as pd

URL = "https://raw.githubusercontent.com/mwaskom/seaborn-data/master/penguins.csv"
```

```
df = pd.read_csv(URL) # ncessite un accs Internet
print(df.shape)
df.head()
```

Astuce : pour travailler hors-ligne, téléchargez le CSV une fois et placez-le dans `data/`, puis :

```
df = pd.read_csv("data/penguins.csv")
```

Option B — Jeu intégré (sans Internet)

Listing 3 Importer un dataset scikit-learn (Wine)

```
from sklearn.datasets import load_wine
import pandas as pd

wine = load_wine(as_frame=True)
df = wine.frame.copy() # features + target
target_name = "target"
features = wine.feature_names
df.head()
```

Option C — Fichiers locaux (CSV, Excel, JSON, Parquet)

Listing 4 Lectures courantes avec pandas

```
import pandas as pd

df_csv = pd.read_csv("data/mon_fichier.csv")
df_xlsx = pd.read_excel("data/mon_fichier.xlsx", sheet_name=0)
df_json = pd.read_json("data/mon_fichier.json")
# Parquet (colonnes types, rapide)  ncessite pyarrow ou fastparquet
# pip install pyarrow
# df_parq = pd.read_parquet("data/mon_fichier.parquet")
```

5 Étape 3 — Nettoyage & préparation (exemple Penguins)

Colonnes utiles pour débuter : `bill_length_mm`, `bill_depth_mm`, `flipper_length_mm`, `body_mass_g`, `species`.

Listing 5 Filtrer colonnes, gérer les NA, fabriquer X/y

```
import numpy as np

num_cols = ["bill_length_mm","bill_depth_mm","flipper_length_mm","body_mass_g"]
keep_cols = num_cols + ["species"]

df_clean = df[keep_cols].dropna().copy()
df_clean["species"] = df_clean["species"].astype("category") # cible catgorielle
X = df_clean[num_cols]
y = df_clean["species"]
print(X.shape, y.shape, y.unique())
```

6 Étape 4 — Visualiser sous toutes les coutures

6.1 Distributions

Listing 6 Histogrammes (une figure par variable)

```
import matplotlib.pyplot as plt

for col in num_cols:
    plt.figure()
    X[col].hist(bins=20)
    plt.title(f"Distribution de {col}")
    plt.xlabel(col); plt.ylabel("Frquence")
    plt.tight_layout(); plt.show()
```

6.2 Variabilité par classe

Listing 7 Boîtes à moustaches par espèce

```
for col in num_cols:
    plt.figure()
    data_by_species = [X.loc[y==sp, col] for sp in y.unique()]
    plt.boxplot(data_by_species, labels=list(y.unique()))
    plt.title(f"{col} par espce")
    plt.xlabel("Espce"); plt.ylabel(col)
    plt.tight_layout(); plt.show()
```

6.3 Relations entre variables

Listing 8 Nuages de points (paires simples)

```
pairs = [("bill_length_mm","bill_depth_mm"),
         ("flipper_length_mm","body_mass_g")]

codes = y.cat.codes # 0,1,2
for a,b in pairs:
    plt.figure()
    plt.scatter(X[a], X[b], c=codes)
    plt.title(f"{a} vs {b} (couleur = espce)")
    plt.xlabel(a); plt.ylabel(b)
    cbar = plt.colorbar(ticks=[0,1,2]); cbar.ax.set_yticklabels(list(y.cat.categories))
    plt.tight_layout(); plt.show()
```

6.4 Corrélations

Listing 9 Matrice de corrélations

```
corr = X.corr(numeric_only=True)
import numpy as np

fig, ax = plt.subplots()
```

```
cax = ax.matshow(corr.values)
fig.colorbar(cax)
ax.set_xticks(range(len(corr.columns)))
ax.set_yticks(range(len(corr.columns)))
ax.set_xticklabels(corr.columns, rotation=45, ha="left")
ax.set_yticklabels(corr.columns)
plt.title("Corrlations entre variables")
plt.tight_layout(); plt.show()
```

7 Étape 5 — Premier modèle (introduction)

Objectif : illustrer le *workflow* de base (découpage, standardisation, apprentissage, évaluation) sans entrer dans la théorie.

Listing 10 Pipeline simple : StandardScaler + LogisticRegression

```
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y)

pipe = make_pipeline(StandardScaler(), LogisticRegression(max_iter=1000))
pipe.fit(X_train, y_train)

pred = pipe.predict(X_test)
print("Accuracy:", accuracy_score(y_test, pred))
print(classification_report(y_test, pred))
```

Listing 11 Matrice de confusion (visualisation)

```
import numpy as np
import matplotlib.pyplot as plt

cm = confusion_matrix(y_test, pred, labels=list(y.cat.categories))
fig, ax = plt.subplots()
im = ax.imshow(cm)
ax.set_xticks(range(len(y.cat.categories))); ax.set_yticks(range(len(y.cat.categories)))
ax.set_xticklabels(y.cat.categories, rotation=45, ha="right")
ax.set_yticklabels(y.cat.categories)
plt.title("Matrice de confusion")
for i in range(cm.shape[0]):
    for j in range(cm.shape[1]):
        ax.text(j, i, cm[i, j], ha="center", va="center")
plt.tight_layout(); plt.show()
```

8 À rendre / Questions guidées

1. **Import** : testez Option A (URL) puis Option B (jeu intégré). Avantages/inconvénients ?

2. **Nettoyage** : listez les colonnes avec des valeurs manquantes et justifiez votre stratégie (suppression vs imputation).
3. **Visualisation** : pour chaque graphique, écrivez une phrase d'interprétation (tendance, dispersion, séparabilité des classes).
4. **Modèle** : remplacez la régression logistique par `KNeighborsClassifier` puis `DecisionTreeClassifier`. Comparez l'accuracy.
5. **Reproductibilité** : exportez votre environnement (`requirements.txt`) et décrivez comment relancer le projet sur une autre machine.

Variante rapide (sans Internet). Refaire tout le TP avec `load_wine(as_frame=true)` : adapter la sélection des colonnes numériques, les graphiques et le modèle.