Téléchargement et Préparation du Dataset Covertype

Présentation de la Base de Données Covertype

La base de données **Covertype** provient de l'UCI Machine Learning Repository. Elle est utilisée pour **classer les types de couvert forestier** à partir de mesures cartographiques (sol, altitude, pente, distance aux points d'eau, etc.).

Caractéristiques du Dataset

- **Nombre d'observations** : 581 012
- **Nombre de variables** : 54 (features continues et binaires)
- **Nombre de classes** : 7 types de couverture forestière (1 à 7)
- **Problème à résoudre** : Classification supervisée

Les classes ne sont **pas équilibrées**, ce qui peut influencer la performance des modèles de classification. Les proportions des classes dans l'ensemble original sont les suivantes :

Classe	Type de forêt	Effectif	Proportion (%)
1	Épicéa	211 840	36.5
2	Pin	283 301	48.8
3	Peuplier	35 754	6.2
4	Bouleau	2 747	0.5
5	Érable	9 493	1.6
6	Hêtre	17 367	3.0
7	Mélèze	18 510	3.2

Objectif de l'Échantillonnage

Pourquoi réduire la taille du dataset ?

Le dataset **Covertype est très grand** (581 012 individus). En raison du **temps de calcul important**, nous avons décidé d'utiliser un **échantillon plus petit**, tout en conservant la distribution des classes.

Comment gérer le déséquilibre des classes ?

Certaines classes sont **très majoritaires** (ex : **Pin et Épicéa** représentent à eux seuls **85% des données**), tandis que d'autres sont **très minoritaires** (ex : **Bouleau** à seulement **0.5%**). Nous avons appliqué un **échantillonnage différencié** :

- **Sous-échantillonnage des classes majoritaires** (Épicéa et Pin) → **5% de leurs effectifs d'origine**
- **Sur-échantillonnage relatif des classes minoritaires** (Peuplier, Bouleau, Érable, Hêtre, Mélèze) → **20% de leurs effectifs d'origine**

Nous ne supprimons pas totalement le déséquilibre, car nous souhaitons **tester nos modèles dans des conditions réalistes**, où certaines classes restent plus rares que d'autres.

L'échantillon obtenu comptera un peu plus de 40 000 individus, dont 60% seront réservés à l'entrainement des modèles, 20% à la validation des hyperparamètres, et 20% aux tests.

Téléchargement et Préparation des Données

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Téléchargement direct des données depuis l'URL
url =
 ↪  "https://archive.ics.uci.edu/ml/machine-learning-databases/covtype/covtype.data.gz"
column_names = [f'Feature_{i}' for i in range(1, 55)] + ['Cover_Type']
data = pd.read_csv(url, header=None, names=column_names)

# Définition des taux d'échantillonnage (différent selon les classes)
sampling_rates = {1: 0.05, 2: 0.05, 3: 0.2, 4: 0.2, 5: 0.2, 6: 0.2, 7: 0.2}
```

```python
# Échantillonnage différencié par classe
sampled_data = pd.concat([
    data[data['Cover_Type'] == cls].sample(frac=sampling_rates[cls],
 ↪  random_state=42)
    for cls in data['Cover_Type'].unique()
])

# Réinitialisation des index après échantillonnage
sampled_data = sampled_data.reset_index(drop=True)

# Affichage des effectifs par classe après échantillonnage
print("Effectifs par classe après échantillonnage différencié :")
print(sampled_data['Cover_Type'].value_counts().sort_index())

# Division des données en ensembles d'entraînement, validation et test
train_data, temp_data = train_test_split(sampled_data, test_size=0.4,
 ↪  random_state=42, stratify=sampled_data['Cover_Type'])
val_data, test_data = train_test_split(temp_data, test_size=0.5,
 ↪  random_state=42, stratify=temp_data['Cover_Type'])

# Affichage des tailles des ensembles
print(f"\nTaille des ensembles :")
print(f"  - Entraînement : {len(train_data)} lignes")
print(f"  - Validation : {len(val_data)} lignes")
print(f"  - Test : {len(test_data)} lignes")

# Affichage des effectifs par classe dans chaque ensemble
print("\nEffectifs par classe dans l'ensemble d'entraînement :")
print(train_data['Cover_Type'].value_counts().sort_index())

print("\nEffectifs par classe dans l'ensemble de validation :")
print(val_data['Cover_Type'].value_counts().sort_index())

print("\nEffectifs par classe dans l'ensemble de test :")
print(test_data['Cover_Type'].value_counts().sort_index())

# Sauvegarder les ensembles en fichiers CSV
train_data.to_csv('covertype_train.csv', index=False)
val_data.to_csv('covertype_val.csv', index=False)
test_data.to_csv('covertype_test.csv', index=False)
```

```
Effectifs par classe après échantillonnage différencié :
Cover_Type
```

```
1    10592
2    14165
3     7151
4      549
5     1899
6     3473
7     4102
Name: count, dtype: int64

Taille des ensembles :
  - Entraînement : 25158 lignes
  - Validation : 8386 lignes
  - Test : 8387 lignes

Effectifs par classe dans l'ensemble d'entraînement :
Cover_Type
1    6355
2    8499
3    4291
4     329
5    1139
6    2084
7    2461
Name: count, dtype: int64

Effectifs par classe dans l'ensemble de validation :
Cover_Type
1    2118
2    2833
3    1430
4     110
5     380
6     695
7     820
Name: count, dtype: int64

Effectifs par classe dans l'ensemble de test :
Cover_Type
1    2119
2    2833
3    1430
4     110
5     380
```

```
6     694
7     821
Name: count, dtype: int64
```