Machine Learning in Python

Mahmoud Elsawy and Jean Luc Bouchot

Centre Inria d'Université Côte d'Azur

June 10, 2025

Outline

Les Étapes Réalisées Jusqu'à Présent

Votre premier script devrait vous avoir donné un compréension de l'exploration et la visualisation des données, incluant

- Collection/chargement des images des chiffres 0 à 9.
- Normalisation des données.
- Exploration et visualisation de la base de données
- Calculs de trois *features*:
  - une réduction de dimensions par ACP (combien de composantes?)
  - des composantes local de mesure d'intensités
  - une mesure de l'intensité globale des gradients
- Création d'une nouvelle matrice des caractéristiques F par concaténation des features.

Feature 0 Feature 1 Feature 2 Feature 3 Feature 4

Division des données pour l'apprentissage

- **Pourquoi diviser les données ?**
  - Évite le sur-apprentissage (*overfitting*).
  - Permet de tester le modèle sur des données inédites.
  - Améliore la généralisation du modèle.
- **Types de division des données :**
  - **Entraînement/Test** : - 80% pour l'entraînement, 20% pour le test.
  - **Validation croisée (Cross-Validation)** : découpage en plusieurs sous-ensembles pour tester plusieurs configurations.
  - **Stratification** : assure une distribution équilibrée des classes dans les ensembles d'apprentissage et de test.
- **Impact sur l'apprentissage :**
  - Un mauvais découpage peut fausser les résultats.
  - Une proportion trop faible d'exemples de test peut limiter la fiabilité de l'évaluation.
- Trois ensembles:
  - Ensemble d'apprentissage (training): optimization des paramètres
  - Ensemble de validation: hyperparameter tuning
  - Ensemble de test: évaluation d'un modèle

Méthodes de classification avec SVM

- **Support Vector Machine (SVC)**
  - Trouve une frontière optimale pour séparer les classes.
  - Utilise des **kernels** pour gérer des distributions complexes.
- **Validation croisée (Cross-Validation)**
  - Utilisation de **K-Fold** : divise les données en K sous-ensembles et entraîne/teste plusieurs fois.
  - Permet de mieux évaluer la performance réelle du modèle.
- **Stratification**
  - Assure une **répartition équilibrée des classes**.
  - Utile pour éviter les biais dus à un déséquilibre des données.
- **One-vs-One vs One-vs-Rest**
  - **One-vs-One (OvO)** : entraîne un classificateur pour chaque paire de classes.
  - **One-vs-Rest (OvR)** : entraîne un modèle pour chaque classe contre toutes les autres.
  - Le choix dépend de la taille et de la complexité du jeu de données.

Implémentation

Pour le second script, vous devez:

- Charger les données bruts,
- Implémenter une première division train/test,
- Visualiser les distributions des classes dans chacun des ensembles, les comparer
- Mettre en place des fonctions utiles à une `Pipeline` de traitement des données
- Mettre en place un simple SVClassifier avec un noyau linéaire, optimiser les paramètres, et rapporter l'erreur de test
- Consider un SVC plus complexe avec divers (hyper)paramètres à optimiser et mettre en place un validation croisée.
- Commenter sur les temps de calculs et fiabilité des diverses stratégies d'apprentissage (OvO, OvR)
- Si le temps permet, tester d'autres modèles d'apprentissage. Lesquels? Pourquoi? Quels paramètres optimiser?