Document descriptif du projet ML

1. Contexte et objectif du projet

Ce projet vise à mettre en place un **pipeline Machine Learning complet et industrialisé** pour la **prédiction des médailles aux Jeux Olympiques d'été**, en s'appuyant sur un **dataset réel** couvrant plus de 125 ans d'historique des Jeux Olympiques.

L'objectif principal est de prédire si un athlète (ou une participation) remportera une médaille (**Gold, Silver, Bronze ou aucune médaille**) à partir de caractéristiques historiques et sportives.

Le projet suit une démarche **MLOps**, intégrant :

- le versioning du code et des données,
- l'entraînement et l'évaluation de modèles ML,
- le tracking des expériences,
- l'automatisation via CI/CD.

---

2. Dataset

2.1 Source des données

- **Nom du dataset** : *125 Years of Summer Olympics*
- **Source** : Kaggle
- **Lien** : https://www.kaggle.com/code/hamdallak/125-years-of-summer-olympics-analysis-visual/input

2.2 Description générale

Le dataset contient des informations détaillées sur :

- les athlètes,
- les pays (NOC),
- les sports et disciplines,
- les éditions des Jeux Olympiques,
- les médailles remportées.

Chaque ligne représente la participation d'un athlète à une épreuve donnée.

2.3 Variable cible

- **Target (y)** : `Medal`
- Valeurs possibles : `Gold`, `Silver`, `Bronze`, `None`

Selon la version du projet, cette variable pourra être :

- transformée en **classification binaire** (Médaille vs Pas de médaille),
- ou conservée en **classification multi-classes**.

---

3. Modèle de Machine Learning

3.1 Modèle choisi

- **Random Forest Classifier**

3.2 Justification du choix

- Robuste aux données bruitées
- Gère bien les features catégorielles après encodage
- Peu sensible au surapprentissage comparé à d'autres modèles
- Très utilisé comme baseline fiable en ML

3.3 Hyperparamètres initiaux

- `n_estimators = 100`
- `max_depth = 5`
- `random_state = 42`

---

4. Métriques d'évaluation

Les performances du modèle seront évaluées à l'aide de :

- **Accuracy** : taux global de prédictions correctes
- **F1-score** : compromis entre précision et rappel, particulièrement adapté aux classes déséquilibrées

---

5. Flux de travail (Workflow MLOps)

5.1 Initialisation du projet

- Initialisation d'un dépôt **Git**
- Mise en place de **DVC (Data Version Control)** pour :
- versionner les datasets,
- suivre les changements de données au fil du projet.

5.2 Gestion des versions du dataset

Le projet comportera **au moins trois versions du dataset** :

Version 1 – Dataset brut

- Fichier : `olympics_raw.csv`
- Données originales sans modification

Version 2 – Dataset nettoyé et enrichi

- Fichier : `olympics_cleaned.csv`
- Nettoyage :
- gestion des valeurs manquantes,
- suppression des doublons,
- normalisation des formats
- Feature engineering de base :
- âge de l'athlète,
- nombre de participations précédentes

Version 3 – Dataset avancé

- Fichier : `olympics_featured.csv`
- Ajout de features avancées
- 6. Entraînement et preprocessing

6.1 Script principal

- Script Python : `train.py`

Fonctions principales :

- chargement des données via DVC,
- preprocessing (encodage, normalisation),
- split train/test,
- entraînement du modèle Random Forest,
- évaluation des performances,
- sauvegarde du modèle.

---

7. Tracking des expériences avec MLflow

MLflow sera utilisé pour :

- suivre les hyperparamètres,
- enregistrer les métriques (Accuracy, F1-score),
- comparer les résultats entre versions de datasets,
- sauvegarder les modèles entraînés.

---

8. Automatisation avec GitHub Actions

Un pipeline **CI/CD** sera mis en place avec **GitHub Actions**.

8.1 Déclenchement

- À chaque `push` sur le dépôt Git.

8.2 Étapes du pipeline

1. Installation des dépendances Python
2. Restauration des données via DVC
3. Exécution du pipeline d'entraînement
4. Évaluation du modèle
5. Enregistrement automatique des résultats dans MLflow

---

9. Fonctionnalité avancée

Une fonctionnalité avancée sera ajoutée, par exemple :

- **Comparaison automatique des performances entre versions de datasets**
- Ou **détection du déséquilibre des classes et pondération automatique**
- Ou **visualisation des performances dans MLflow (courbes, matrices de confusion)**

Cette fonctionnalité renforcera l'aspect MLOps et expérimental du projet.

---

10. Analyse et comparaison des résultats

Les résultats permettront de :

- comparer l'impact du nettoyage et du feature engineering,
- analyser l'évolution des métriques entre les versions,
- justifier les choix de features et de modèle.

Une explication détaillée des différences de performance sera fournie pour chaque version du dataset.