Projet Scraping

Collecter automatiquement les données d’internet

Étude de cas

Table des matières

1. Contexte et objectifs

Dans le cadre du développement d'un modèle de prédiction de résultats sportifs, l'accès à des données historiques fiables est indispensable. Le modèle nécessite :

- Les scores de matchs passées,
- Les côtes d'ouvertures et de fermetures,
- Les matchs à venir,
- Ainsi qu'un suivi saisonnier sur plusieurs compétitions et ou équipes.

Pour obtenir ces données, j'ai dans un premier temps opter pour l'option la plus simple, c'est-à-dire explorer les API de données sportives, notamment *The Odss API* et d'autres services similaires. Cependant, une limite majeure est rapidement apparue : Pour collecter 10 saisons complètes sur plusieurs compétitions majeures, le budget API estimé devenait :

- Trop important pour un projet individuel,
- Proportionnel au volume de requêtes,
- Et difficilement justifiable pour un dataset en constante évolution.

De plus, certaines API ne fournissent pas l'historique complet, ou limitent l'accès aux variations de cotes (notamment opening vs closing)

Ainsi, afin de contourner ces limites, la stratégie adoptée a été de développer un scraper personnalisé basé sur Playwright, ciblant le site *OddsPortal.com*.

2. Le scraping de données

Le scraping de données est une méthode qui consiste à récupérer automatiquement des informations depuis un site web. Cette méthode présente plusieurs avantages :

- Aucun coût par requête,
- Possibilité de récupérer autant de saisons, compétitions et équipes que nécessaires,
- Grande flexibilité et contrôle total sur la structure des données.

Dès le départ, j'avais deux objectifs principaux pour mon scraper :

1. Modularité : pouvoir choisir exactement les compétitions, saisons, sports et bookmakers.
2. Reproductibilité : obtenir toujours les mêmes résultats à partir des mêmes paramètres.

La première version récupérait les données selon trois critères principaux :

- La compétition,
- La saison,
- Le bookmaker

Ce choix me permettait non seulement de sélectionner précisément la compétition à analyser, mais aussi le bookmaker offrant le plus de données.
J'ai choisi d'utiliser uniquement les côtes de Betclic, pour une meilleures cohérence dans le modèle de ML.

L'architecture initiale reposait sur :

- Une fonction principale,
- Un passage de paramètres par arguments dans le terminal.

Mais, un problème majeur est vite apparu : je devais attendre la fin complète du script pour en relancer un nouveau, ce qui entraînait une perte de temps considérable.

Pour pallier cette limite, j'ai implémenté :

- Un fichier de configuration .json, pour facilité l'automatisation et assurer la reproductibilité,
- Une fonction orchestrateur, chargée d'appeler la fonction principale autant de fois que nécessaire jusqu'à avoir collecté toutes les données.

Comme le scraping restait lent, j'ai ajouté une double parallélisation :

- Niveau match : 3 matchs scrappés simultanément
- Niveau global : 3 scripts de scrapping exécutés en parrallèle

Cela permet de réduire le temps d'exécution par un facteur allant jusqu'à 9. Cependant, je n'ai pas poussé davantage la parralélisation, car ma machine imposait des limites, ce qui aurait entraîné :

- Un ralentissement global
- Un risque accru de perte de données,
- Des pages non chargées ou incomplètes.

3. Problèmes rencontrés

Le scraping présente plusieurs contraintes :

- Données ignorées à cause :
    - De particularité du site,
    - De chargements incomplèts,
    - Ou d’erreurs ponctuelles.
- Limitation des données, en effet, oddsPortal propose :
    - Les côtes,
    - les noms des équipes,
    - Les compétitions
    - Les dates de matchs et de publications des côtes

  Mais cela ne couvre pas certains types de données avancées comme excepted goals, les statistiques de matchs, etc.

4. Perspectives d’évolution

À moyen terme, l’objectif est de :

- Diversifier les sources de données, afin de réduire la dépendance à un seul site,
- Récupérer d’autres types de données plus riches,
- Tester le scraper sur d’autres sports disponible sur OddsPortal (fonctionnel théroqiement, mais pas encore validé en pratique).

Une des pistes à explorer pour les atteindre et l’implémentation de L’IA. Certaines solutions open-source utilise déjà l’intelligence artificielle, comme *ScrapeGraphAI*, qui peut extraire des données complexes à partir d’un schéma cible. Cependant :

- Ces solutions reposent sur des API de LLM (ex : GPT),
- Elles entraînent donc un coût supplémentaire non négligeable.

Elles restent toutefois intéressantes pour une futures version du projet.