Cahier des Charges

Projet d'Analyse et de Prédiction de Données de Formule 1

Hajda Altin
Telley Cyril

23 Avril 2025

Période du projet : 11 Avril 2025 -

Table des matières

1 But général du projet

Développer un système complet et automatisé pour la collecte, l'analyse et la prédiction des données issues des courses de Formule 1. Ce système comprendra une interface utilisateur web (basée sur Dash ou Streamlit) pour la visualisation des analyses et des prédictions générées.

2 Contexte et objectifs

Contexte : La Formule 1 est un sport générant une quantité massive de données à chaque événement. L'analyse approfondie de ces données historiques peut révéler des tendances, des schémas de performance et fournir des informations précieuses pour comprendre les résultats passés et potentiellement anticiper les performances futures. Ce projet vise à exploiter ces données pour créer des outils d'analyse et de prédiction accessibles dans le cadre d'un projet limité dans le temps.

Objectifs spécifiques : Le projet se décompose en plusieurs objectifs clés :

1. Mettre en place un système de web scraping robuste pour extraire les résultats des sessions (Essais Libres, Qualifications, Courses, Qualifications Sprints, Courses Sprints) depuis le site officiel `formula1.com`.
2. Intégrer l'API `FastF1` (via sa bibliothèque Python) pour acquérir des données télémétriques et contextuelles détaillées complémentaires.
3. Concevoir et implémenter une méthode de stockage structurée et pérenne pour les données collectées, initialement sous forme de fichiers CSV locaux.
4. Développer un pipeline de nettoyage et de prétraitement des données, incluant la fusion cohérente des informations provenant des différentes sources (`formula1.com` et `FastF1`).
5. Élaborer et entraîner des modèles de Machine Learning capables de prédire divers aspects des courses futures (ex : classement final, composition du podium, détenteur du meilleur tour en course).
6. Mettre en place un processus opérationnel pour générer et actualiser ces prédictions avant chaque Grand Prix (dans la mesure du possible durant la période du projet).
7. Créer une interface utilisateur web interactive (avec Dash ou Streamlit) pour permettre la visualisation des données historiques, des analyses et des prédictions générées par les modèles.

3 Présentation des données à utiliser

3.1 Sources de données et droit d'utilisation

Le projet s'appuiera sur les sources de données suivantes :

Site Web Officiel (`formula1.com`) :

Droit d'utilisation : Le scraping sera effectué en conformité avec le fichier `robots.txt` du site et ses conditions générales d'utilisation. Des délais entre les requêtes et un user-agent approprié seront utilisés pour éviter toute surcharge du serveur. L'utilisation des données sera limitée à des fins d'analyse personnelle et non commerciale dans le cadre de ce projet.

API FastF1 (Bibliothèque Python `FastF1`) :

Droit d'utilisation : L'utilisation sera conforme à la licence de la bibliothèque `FastF1` et aux conditions d'utilisation des données sous-jacentes qu'elle expose (souvent issues de sources publiques ou d'API officielles).

3.2 Description des données (attributs, quantité)

Périmètre temporel : Les données couvriront a minima les 5 dernières saisons complètes de Formule 1, ainsi que la saison en cours, en fonction du temps disponible pour la collecte.

Attributs envisagés par session : (Liste non exhaustive, à affiner lors de l'exploration)

- *Informations générales :* Année, nom/identifiant du Grand Prix, nom du circuit, date(s), type de session (FP1, FP2, FP3, Q, SQ, R, SR).
- *Résultats (Classement) :* Position finale, numéro de voiture, nom complet du pilote, nom de l'écurie, temps final ou écart au leader, nombre de tours complétés, statut (ex : Fini, Abandon, +1 Lap), points marqués.
- *Données de Qualification / Sprint Qualification :* Temps réalisés dans chaque segment (Q1, Q2, Q3), classement final de la séance.
- *Temps au tour :* Meilleurs temps au tour par pilote pour chaque session, potentiellement l'historique complet des temps au tour (via `FastF1`).
- *Arrêts aux stands (Pit Stops) :* Nombre total d'arrêts, tour de chaque arrêt, durée de l'immobilisation (si disponible).
- *Données `FastF1` complémentaires :* Données de télémétrie (vitesse, RPM, rapport engagé, position sur la piste), informations sur les pneus (type de gomme utilisé, usure/tours effectués), temps par secteur, conditions météorologiques (température air/piste, pluie).

Quantité estimée : La volumétrie variera significativement. Les résultats de session représenteront quelques centaines de lignes par événement. L'inclusion des données de télémétrie et des temps au tour complets pourrait générer plusieurs milliers, voire dizaines de milliers, de lignes par course. Le stockage total représentera plusieurs centaines de Mo ou quelques Go sous forme de fichiers CSV.

3.3 Méthodes d'extraction et de stockage

`formula1.com` (Scraping) :

Utilisation de bibliothèques Python standards : `requests` pour les requêtes HTTP et `BeautifulSoup4` (ou `lxml`) pour l'analyse syntaxique du HTML.

Développement d'une logique de navigation automatisée pour parcourir les archives par année, Grand Prix et type de session.

`FastF1` (API Client) :

Utilisation directe de la bibliothèque Python `FastF1` et de ses fonctions pour requêter et récupérer les données structurées (télémétrie, pneus, météo, etc.).

Stockage Initial :

Sauvegarde des données brutes et traitées dans des fichiers au format CSV (*Comma-Separated Values*).

Organisation dans des dossiers locaux, suivant une convention de nommage claire et une structure de colonnes prédéfinie et documentée. (Ex : `data/raw/YYYY/YYYY_XX_GPName_SessionType.csv`, `data/processed/master_results.csv`)

4 Architecture globale et technologies envisagées

Architecture Modulaire : Le système sera conçu autour de modules distincts et faiblement couplés pour faciliter le développement, les tests et la maintenance :

- Module d'Extraction : Scraper `formula1.com` + Client API `FastF1`.

- Module de Gestion des Données : Lecture, écriture, validation et fusion des fichiers CSV.
- Module de Traitement : Nettoyage, prétraitement, ingénierie de caractéristiques (feature engineering).
- Module de Modélisation : Entraînement, évaluation et sauvegarde des modèles de Machine Learning.
- Module de Prédiction : Chargement des modèles et génération des prédictions pour les nouvelles courses.
- Module d'Interface Utilisateur : Application Web (Dash/Streamlit) pour la visualisation.

Flux de Données Principal : (a) Extraction (Scraper/API) → Fichiers CSV bruts.

(b) Lecture CSV bruts → Traitement (Nettoyage, Fusion, Feature Engineering) → Fichiers CSV traités / Datasets pour ML.

(c) Lecture Datasets ML → Entraînement Modèles → Modèles sérialisés (fichiers).

(d) Chargement Modèles + Données nouvelles courses → Script de Prédiction → Fichier/Base de Prédictions.

(e) Lecture Données Traitées + Prédictions → Interface Utilisateur.

Stockage : Utilisation du système de fichiers local pour les fichiers de données (CSV) et les modèles de ML sauvegardés (ex : via `pickle` ou `joblib`).

Technologies Principales : • *Langage de programmation :* Python (version 3.9+ recommandée).

- *Manipulation de données :* Pandas, NumPy.
- *Web Scraping :* Requests, BeautifulSoup4 / lxml.
- *API F1 :* FastF1.
- *Machine Learning :* Scikit-learn (pour les algorithmes de base, pipelines, métriques), potentiellement XGBoost, LightGBM ou CatBoost pour des modèles de gradient boosting plus performants.
- *Interface Utilisateur (Web App) :* Dash (Plotly) ou Streamlit. Le choix final dépendra de la complexité souhaitée et de la facilité d'intégration.
- *Gestion de l'environnement :* `venv` (standard Python) ou `conda` (Anaconda/Miniconda) pour isoler les dépendances du projet. Fichier `requirements.txt` ou `environment.yml`.

Exécution : Les différents modules seront implémentés sous forme de scripts Python exécutables depuis la ligne de commande. L'exécution sera initialement manuelle. Une planification simple pourrait être envisagée mais n'est pas un objectif prioritaire vu le délai court.

5 Techniques, méthodes et algorithmes envisagés

Analyse Exploratoire des Données (EDA) :

- Analyse statistique descriptive : Moyennes, médianes, écarts-types, quantiles des temps au tour, écarts, nombre d'arrêts, etc.
- Statistiques agrégées par pilote, équipe, circuit, saison pour identifier des tendances.
- Visualisations (via notebooks Jupyter ou directement dans l'UI) : Histogrammes, box plots, scatter plots, time series pour suivre l'évolution des classements, comparer les performances, étudier les corrélations.

Nettoyage et Prétraitement :

- Traitement des valeurs manquantes (NaN) : Suppression ou imputation simple (moyenne, médiane, mode).

- Standardisation des formats : Conversion des temps (en secondes ou millisecondes), unification des formats de date/heure.
- Encodage des variables catégorielles : Transformation des noms de pilotes, équipes, circuits en représentations numériques (ex : One-Hot Encoding, Label Encoding).
- Normalisation/Standardisation des variables numériques si requis par les algorithmes ML.

Ingénierie de Caractéristiques (Feature Engineering) :

- Création de variables basées sur l'historique : Performance récente du pilote/équipe, performance historique sur le circuit, résultats des qualifications, etc.
- Intégration des caractéristiques intrinsèques du circuit.
- Agrégation de données fines (si temps et données permettent) : Moyenne/médiane des temps au tour, nombre d'arrêts.
- Variables relatives : Écart de performance par rapport au coéquipier, position sur la grille.

Modélisation (Machine Learning) :

- *Priorité : Problèmes de Classification :*

  Prédiction du Podium (Top 3) : Approche binaire (pilote X dans le top 3 ?) ou multi-classe (prédiction directe des 3 premiers). Algorithmes : Régression Logistique, Random Forest Classifier, Gradient Boosting (LightGBM/XGBoost).

  Prédiction du Vainqueur : Multi-classe.
- *Si temps permet : Classement ou Régression :*

  Prédiction du Classement Complet (approche simplifiée via régression de la position ou classification).

  Prédiction Nombre d'Arrêts (régression).

Évaluation des Modèles :

- Métriques adaptées :

  Classification : Exactitude, Précision, Rappel, F1-Score (pondéré/macro), Log Loss.

  Régression : MAE, RMSE.
- Techniques de validation : Validation croisée temporelle (ex : entraînement sur saisons N-2, validation sur saison N-1, test sur saison N) pour simuler des prédictions futures. Mise de côté d'un jeu de test final (ex : dernière saison complète disponible).

6 Résultats attendus

À l'issue du projet (échéance), les livrables suivants sont attendus :

- Un ensemble de scripts Python fonctionnels pour l'extraction (scraping/API), le traitement de base, l'entraînement d'au moins un modèle prédictif (ex : podium) et la génération de prédictions.
- Une collection organisée de fichiers CSV contenant les données F1 structurées pour les saisons ciblées.
- Au moins un modèle de Machine Learning entraîné et sauvegardé pour la tâche de prédiction principale.
- Des fichiers de sortie (ou affichage UI) contenant les prédictions pour les courses à venir (ou un exemple basé sur des données historiques).
- Une application web locale (`Dash`/`Streamlit`) fonctionnelle permettant à minima :
  - La consultation simple des données historiques (ex : résultats par GP/saison).

  - L'affichage des prédictions générées.
- Une documentation technique concise (README) décrivant l'architecture simplifiée, l'installation et l'utilisation des scripts/UI.

7 Risques, points critiques et problèmes potentiels

Risques Identifiés (principaux pour un projet court) :

- *Maintenance/Blocage du Scraper :* Changements sur `formula1.com` ou mesures anti-scraping pouvant fortement ralentir/bloquer la collecte.
- *Qualité/Disponibilité des Données :* Incohérences ou délais dans la disponibilité des données pouvant impacter le prétraitement et la pertinence des prédictions.
- *Complexité Imprévue :* Sous-estimation du temps nécessaire pour le nettoyage des données, le feature engineering ou le développement de l'UI.
- *Précision Limitée :* Attentes réalistes concernant la précision des modèles vu la complexité de la F1 et le temps limité pour l'optimisation.
- *Gestion du Temps :* Le délai court (env. 9 semaines) est le risque principal, nécessitant une priorisation stricte des fonctionnalités.

Points Critiques pour le Succès : • Obtenir rapidement un scraper fonctionnel et stable pour les données essentielles (résultats).

- Mettre en place rapidement un pipeline de données minimal (Collecte -> Nettoyage -> Stockage).
- Se concentrer sur un ou deux objectifs de prédiction clairs (ex : podium).
- Choisir des technologies maîtrisées pour l'UI (Dash/Streamlit) pour un développement rapide.

Problèmes Rencontrés à ce Stade (23 Avril 2025) :

- (Section à remplir si des difficultés spécifiques ont déjà été identifiées lors des deux premières semaines, sinon indiquer "Exploration initiale des sources et des outils en cours.")

8 Planification prévisionnelle des étapes du projet

Le projet a débuté le 11 Avril 2025 et se terminera le . Le planning prévisionnel suivant, réparti sur environ 9 semaines, est proposé :

Phase 1 : *Initialisation & Exploration (env. 1.5 semaines : 11 Avril 2025 - 25 Avril 2025)*

- Analyse détaillée des sources (`formula1.com`, `FastF1`).
- Définition structure CSV cible.
- Développement prototype scraper (core fonctionnalités).
- Finalisation Cahier des Charges (ce document).
- Configuration environnement de développement.

Phase 2 : *Développement Core Data (env. 2.5 semaines : 28 Avril - 16 Mai 2025)*

- Développement scraper complet et intégration API `FastF1`.
- Implémentation scripts gestion CSV (lecture/écriture/fusion).
- Collecte données historiques (focus sur les 5 dernières saisons).
- Implémentation nettoyage et prétraitement de base.

Phase 3 : *Modélisation & Prédiction (env. 2.5 semaines : 19 Mai - 30 Mai 2025)*

- Feature engineering (variables clés).
- Entraînement/évaluation modèles prioritaires (ex : podium).
- Sélection et sauvegarde du (des) meilleur(s) modèle(s).
- Développement script de génération de prédictions.

Phase 4 : *Développement Interface Utilisateur (env. 1.5 semaines : 2 Juin - 13 Juin 2025)*

- Choix final Dash/Streamlit et mise en place structure App.
- Développement composants UI (visualisation données, affichage prédictions).
- Intégration avec modules données/prédictions.

Phase 5 : *Tests, Raffinement & Finalisation (env. 0.5 semaine : 13 Juin -)*

- Tests d'intégration finaux.
- Corrections de bugs mineurs.
- Finalisation documentation (README).
- Validation finale et préparation démonstration.
- Livraison finale du projet.

Note : Ce planning est indicatif et pourra être ajusté en fonction de l'avancement réel et des difficultés rencontrées.