Cahier des Charges - Compétition de Data Science :

Credit Scoring

1. Contexte de la Problématique

Le projet consiste à analyser les données de crédit (German Credit Data) afin de développer un système de scoring crédit capable de prédire si un individu est éligible ou non pour un crédit. Les résultats doivent être présentés sous forme de tableau de bord interactif tout en respectant les contraintes d'industrialisation et de déploiement.

2. Objectifs et Découpage des Tâches

Tâche 1 : Exploration des Données (EDA)

1. Charger les données et examiner leur structure (types de variables, dimensions, valeurs manquantes).
   - Bibliothèques : pandas, numpy.
- Actions : `.info()`, `.describe()`, identification des valeurs manquantes.

2. Analyser les distributions des variables et identifier les valeurs aberrantes.
- Visualisations : Histogrammes, boxplots (matplotlib, seaborn).

3. Analyser les corrélations entre les variables et l'étiquette cible (*target* : octroi de crédit).
   - Actions : Calcul de la matrice de corrélation (variables numériques).
- Visualisation : Heatmap, test de chi2 pour variables qualitatives.

4. Documenter les observations clés :
   - Variables avec forte corrélation avec l'octroi/non octroi de crédit.
- Valeurs manquantes et stratégies de traitement (moyenne, médiane, catégorie "inconnu").

Tâche 2 : Développement du Modèle de Prédiction

1. Préparation des Données :

   - Encodage des variables catégoriques (Label Encoding ou One-Hot Encoding).
   - Normalisation ou standardisation des variables numériques.
   - Division des données : Ensemble d'entraînement (80%) et de test (20%).

2. Sélection des Modèles :

- Modèles à tester : Régression logistique, Random Forest, XGBoost, LightGBM.
- Bibliothèques : scikit-learn, xgboost, lightgbm.

3. Entraînement et Validation :

- Validation croisée (K-Fold).
- Métriques d'évaluation : Précision, rappel, F1-score, courbe ROC-AUC.

4. Optimisation :

- Ajustement des hyperparamètres (évaluation GridSearchCV ou RandomizedSearchCV).
- Sélection du modèle avec les meilleures performances.

Tâche 3 : Dockerisation

1. Préparation des fichiers :

- Script Python pour prédire les scores de crédit.
- API REST (Flask ou FastAPI) pour exposer le modèle.

2. Création du Dockerfile :

- Inclure Python, les dépendances (requirements.txt).
- Commandes pour exécuter le script et l'API.

3. Tests locaux :

- Démarrage du conteneur et test de l'API avec des requêtes.

Tâche 4 : Hébergement sur GitHub/GitLab

1. Organiser le repository :

- Structurer le code (dossier `src/`, `data/`, `models/`, etc.).
- Ajouter un README avec explications claires (installation, exécution, déploiement).

2. Versionner le projet :

- Utiliser Git pour suivre les modifications.

3. Publier le repository sur GitHub ou GitLab.

Tâche 5 : Tableau de Bord Interactif

1. Conception des Visualisations :

- Distribution des scores de crédit.
- Importance des variables dans le modèle (Feature Importance).
- Comparaison entre scores prédits et résultats réels.

2. Outil choisi (Power BI ou Looker Studio) :

- Importation des résultats (CSV ou base de données).
- Création de graphiques interactifs.

3. Publication et partage :

- Publier le tableau de bord pour le rendre accessible.

4. Livrables

1. Rapport d'analyse exploratoire.
2. Modèle de scoring prédictif.
3. Dockerfile fonctionnel.
4. Repository GitHub/GitLab documenté.
5. Tableau de bord interactif publié.

5. Contraintes et Risques

- Qualité des données: Gestion des valeurs manquantes et outliers essentielle.
- Temps limité : Prioriser les tâches selon leur impact sur les résultats.
- Compatibilité technique : Assurer que les outils choisis (Docker, Power BI) sont fonctionnels sur l'environnement de travail.

6. Conclusion

Le projet vise à fournir une solution complète et industrialisable pour le scoring de crédit en respectant les meilleures pratiques de la science des données et du développement logiciel. Les livrables finaux répondront aux objectifs de la compétition tout en garantissant une présentation claire et impactante des résultats.