

Encadrant : Peter Sander (sander@unice.fr)

Plan détaillé du rapport - Plan d'avancement pendant la période à plein temps

Remerciements

Abstract

I. Introduction

1. Contexte
2. Présentation globale du projet
3. Problématique

II. Data pipeline

1. Data visualisation:

Fait :

- Isolation des premières KPI (Key Performance Indicator) à afficher
- Généralisation de la data visualisation
- Première implémentation de dashboard automatique
 - Visualisation et représentation de différentes colonnes

A faire :

- Afficher plus de KPI :
 - Matrice de corrélation
 - Adapter les affichages en fonction du type de base, série temporelle
 - La description d'une base (min, max, mean, count,...)
 - Ajouter plus d'indicateurs au dashboard automatique

2. Data model:

Fait :

- Implémentation de 3 algorithmes de régression avec certains paramètres :
 - Gradient Boosting
 - Random Forest
 - Ridge
- Implémentation de 6 algorithmes de classification :
 - Linear SVC
 - AdaBoost
 - Gradient Boosting
 - Random Forest
 - Logistic Regression
- Affichage des résultats avec représentation graphique :
 - scatter plot
 - matrice de confusion
 - affichage des features importance (sous forme d'histogramme)

A faire :

- Ajouter des algorithmes de classification
- Afficher les r2_score et accuracy score
- Stabiliser les algorithmes

3. Sélection automatique des algorithmes:

Fait :

- Récupération de 30-40 bases pour créer les méta-data
- Création d'un script automatique pour récupérer les méta-features de chaque base et leur prédiction
- Création d'une base de méta-data
- Création d'un modèle de méta-learning afin de choisir le meilleur algorithme à utiliser sur une base pour réaliser une régression
- Implémentation d'un random search pour chaque algorithme et trouver les meilleurs hyperparamètres

A faire :

- Création d'un modèle de méta-learning afin de choisir le meilleur algorithme à utiliser sur une base pour réaliser une classification
- Création d'un modèle de méta-learning afin de choisir le meilleur algorithme à utiliser sur une base pour gérer les hyperparamètres

III. Web application

1. Structure app:

Fait :

- Une application Web comprenant différentes fonctionnalités: drag and drop de données (.csv, xlsx, ...), datavisualisation, prédictions, création et gestion de compte.

A faire :

- Ajouter de nouvelles fonctionnalités (cf : 2. client, 3. serveur)
- Ajouter des informations explicatives sur les paramètres à choisir et les résultats.
- Acheter un nom de domaine pour le site web
- Déployer le serveur sur une machine (en attente de nouvelle de notre encadrant)

2. Client : Framework/Component/Design:

Fait :

- Implémentation de React (framework)
- Utilisation de typescript en langage de programmation
- Création de components personnalisés (barre de navigation, cellules)
- Système de sign-up / log in
- Une page gérant l'upload des bases de données
- Une page de gestion des bases déjà upload
- Une page d'analyse-prédiction avec les différents algorithmes et paramètres configurables
- Une page de datavisualisation affichant un dashboard grâce à l'implémentation du module ChartJS
- Interface graphique codée avec Sass (Syntactically Awesome Style Sheets)
- Une connexion avec le serveur pour l'échange de données (requêtes,databases,...)
- Build du client permettant d'accéder au client via l'adresse du serveur, application prête au déploiement (un seul domaine sera nécessaire au déploiement)

A faire :

- Tests unitaires des différentes fonctions principales de l'application web
- Proposer plus de paramètres dans la datavisualisation
- Page permettant une démo de l'application pour les clients non authentifiés
- Gérer les matrices de corrélation
- Implémenter une pré-vérification avant l'envoi des bases de données (surveiller la taille, le type, etc...)

3. Serveur : Node/Mongo :

Fait :

- Un serveur Node, codé en Typescript
- Différentes routes pour gérer les requêtes utilisateurs (plus d'information dans la partie sécurité)
- Gestion des appels python afin d'utiliser les scripts développés par Christel et Nicolas.
- Gestion des comptes clients, hiérarchie de dossiers pour organiser et stocker les informations (bases/analyses) des clients.
- Une connexion avec une base MongoDB où les données (informations personnelles) des clients sont stockées.
- Des fonctions permettant l'analyse des requêtes des clients (vérifications des paramètres de prédictions,...)

A faire :

- Vérifications des requêtes restantes afin d'éviter tout "crash" involontaire du serveur
- Implémentation de potentielles nouvelles routes en fonction des nouveautés ajoutées
- Tester la totalité des fonctionnalités de l'application une fois déployée
 - Test d'envoie de fichiers lourds
 - Calcul du temps d'exécution pour les requêtes complexes
 - Détermination des restrictions (taille de fichier, nombre de requêtes par client) pour un serveur fluide et fonctionnel.

IV. Sécurité

1. Authentification

Fait :

- Authentification par Json Web Token (JWT)
 - Token JWT dans un cookie HttpOnly (pour empêcher les XSS)
 - Token CSRF dans localStorage (pour empêcher les failles CSRF)
- Mot de passe de l'utilisateur doit correspondre à des critères de sécurité.
- Mot de passe de l'utilisateur hasher avec l'algorithme Argon2.
- Confirmation obligatoire de l'e-mail sinon pas d'accès aux fonctionnalités de l'application.

A faire :

- Implémentation de l'authentification à double facteur (code envoyé par e-mail) si l'utilisateur active cette option.

2. Chiffrement

Fait :

- Chiffrement de toutes les données de l'utilisateur (database, analyse) avec AES-256-CBC.
- Mise en place de HTTPS avec redirection HTTP vers HTTPS.

A faire :

- Création d'un certificat HTTPS avec Let's Encrypt une fois le nom de domaine acheté et la machine pour le déploiement obtenue.

3. Sécurité supplémentaire

Fait :

- Mise en place d'un captcha.
- Mise en place d'une limite d'upload par utilisateur côté serveur.
- Vérification de toutes les entrées de l'utilisateur avec un middleware pour éviter les XSS.

A faire :

- Mise en place d'un blocage temporaire si l'utilisateur se trompe à plusieurs reprises au moment du login.
- Effectuer des tests unitaire et d'intégration.

Conclusion

Bibliographie

ChartJS : <https://www.chartjs.org/>

React+Typescript : <https://www.sitepoint.com/react-with-typescript-best-practices/>

React : <https://openclassrooms.com/fr/courses/7008001-debutez-avec-react>

Pour tous les packages Node : <https://www.npmjs.com/>

Authentification+API REST :

<https://openclassrooms.com/fr/courses/6390246-passez-au-full-stack-avec-node-js-express-et-mongodb>

<https://www.codeheroes.fr/2018/03/23/securiser-une-api-rest/>

Chiffrement : <https://lollyrock.com/posts/nodejs-encryption/>

Test unitaire et d'intégration : <https://jestjs.io/fr/docs/getting-started>

Meta-Learning :

https://www.automl.org/wp-content/uploads/2019/05/AutoML_Book_Chapter2.pdf?fbclid=IwAR3etbHkil7zYoMtOnNxxxy19iT3idxX4tGRd9a2grareL05kK9NXFCZRthY

https://www.sciencedirect.com/science/article/abs/pii/S0020025521011130?fbclid=IwAR3XSrevvuJD9fQkkOF6KpAWDwGJJn2E5h31c_cx3mCgL3Lb9SxG1caVkJA

<https://pypi.org/project/pymfe/?fbclid=IwAR0REZr9CBt7soRI73iXjcJaNLtYLktrxDHC8bFBGfdPE5VT5mL4I72HA9c>

https://cran.r-project.org/web/packages/mfe/vignettes/mfe-vignette.html?fbclid=IwAR2BpVm_pDBB-rAvSujVf_quF5Gy1sPHtpHkrnpAIRZtOxf4NbVzJs1IXs

Pour tous les algorithmes de Machine Learning : <https://scikit-learn.org/>

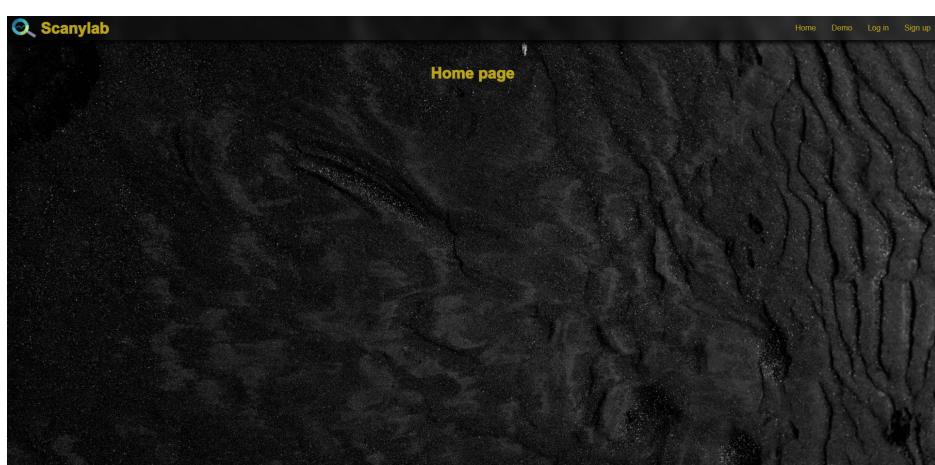
Features importance :

<https://machinelearningmastery.com/calculate-feature-importance-with-python/>

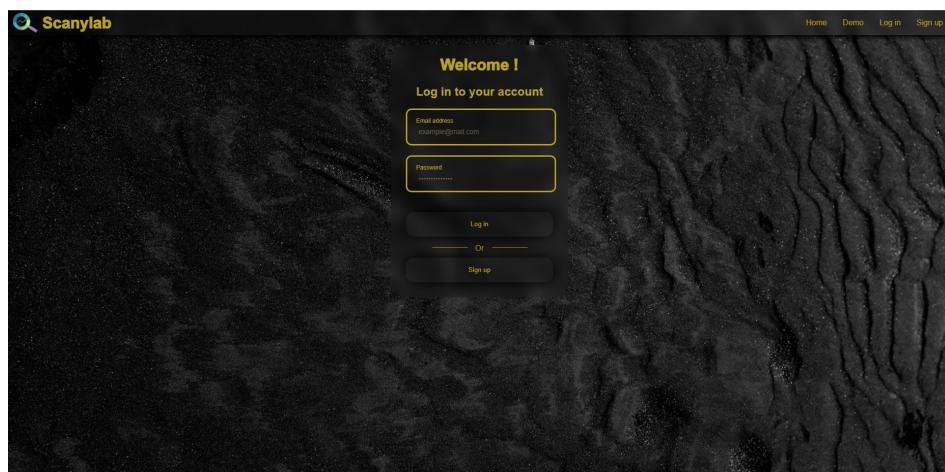
Annexe

Lien github : <https://github.com/iBananos/just-drag-and-drop>

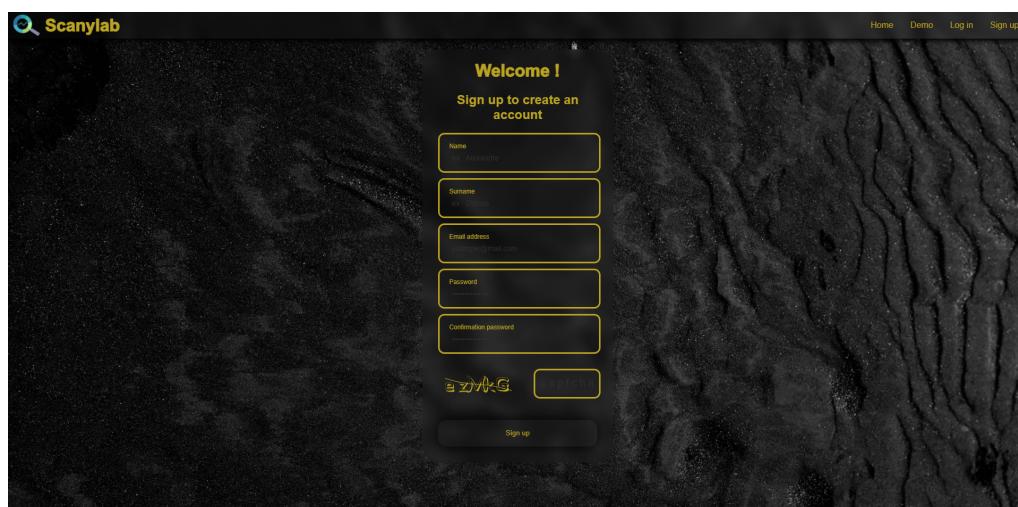
Home page, à compléter une fois l'application aboutie :



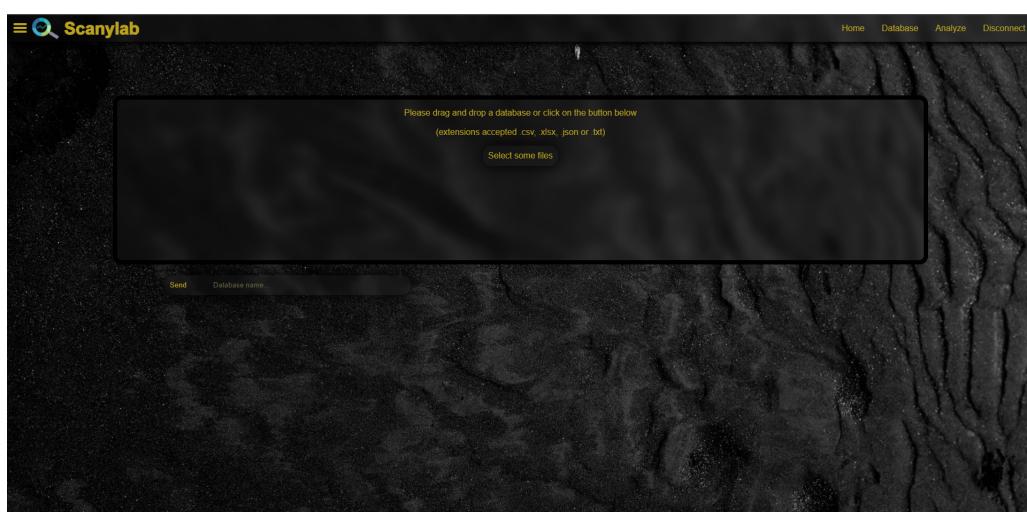
Page de Log in :



Page de Sign up :



Page de drag and drop pour upload une nouvelle base :



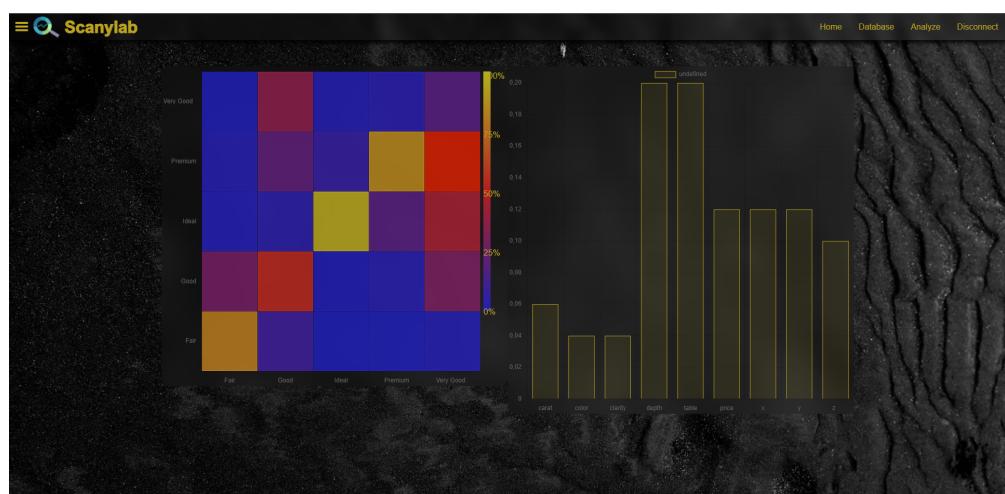
Page "My database" où on peut retrouver ses bases déjà upload et les télécharger et/ou les supprimer :

The screenshot shows the 'My database' section of the Scanylab interface. It displays two uploaded CSV files: 'oui' and 'test'. Both files were uploaded on Thursday, December 16, 2021, at 22:26:54 GMT+0100 (heure normale d'Europe centrale). The size of each file is 3192560 octets and the type is csv. There are small icons for download and delete next to each file entry.

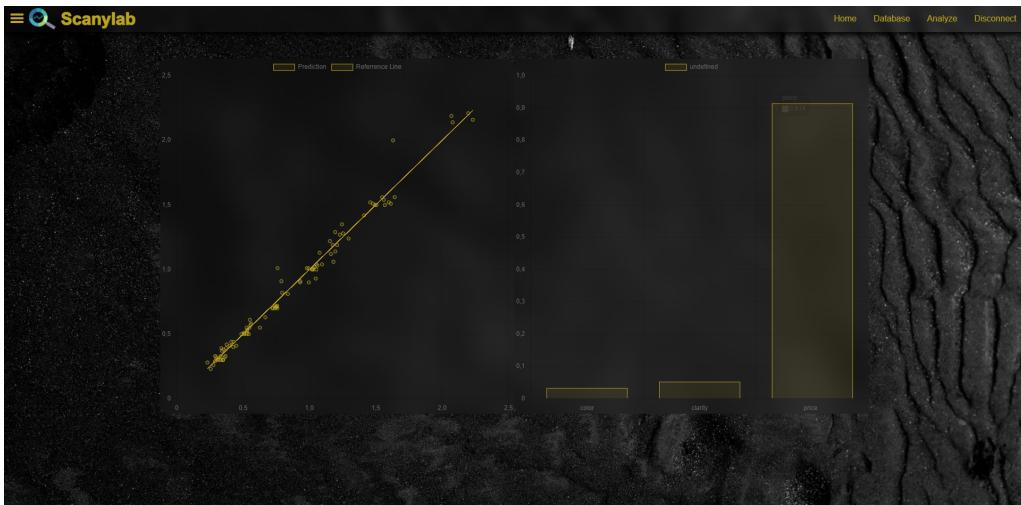
Page d'analyse avec plusieurs select pour choisir la base, les colonnes, l'algorithme puis enfin les paramètres :

The screenshot shows the 'Analyze page' of Scanylab. It includes a dropdown for 'Choose other parameters' containing columns like 'carat', 'color', 'clarity', 'depth', 'table', 'price', 'x', 'y', and 'z'. Below this are several algorithm selection buttons: AUTOMATIC, LINEARSVC, ADABOOST (which is selected), GRADIENT BOOSTING, RANDOM FOREST, and LOGISTIC REGRESSION. There are also input fields for 'n_estimators' (set to 50) and 'learning_rate' (set to 1). A 'Check the new tab for the result !' button and an 'Analyze' button are also present.

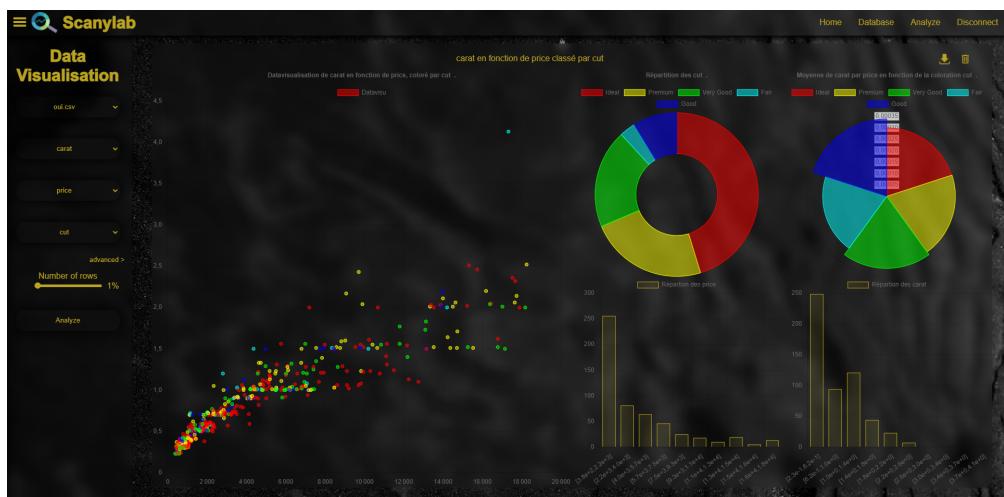
Page d'affichage d'une prédition-classification :



Page d'affichage d'une prédition-régression :



Page de datavisualisation, ici une datavisualisation sur 3 colonnes dont une utilisée pour la coloration :



Résultat une fois la datavisualisation enregistrée:



Page d'historique des analyses déjà réalisées :

The screenshot shows the Scanylab History page. At the top, there are navigation links: Home, Database, Analyze, and Disconnect. Below the navigation, there is a "Load analysis" button and a detailed analysis card. The card displays the following information:

- Title: analyseGradientBoosting
- Date: 2021-12-17T10:55:59.175Z
- Database: test.csv
- Type: prediction
- Algorithm: GradientBoosting
- Parameters:
 - learning_rate: 0.1
 - n_estimators: 100
 - max_depth: 3
 - min_samples_split: 2

On the left side of the main area, there is a vertical list of analysis names: analyseGradientBoosting, analyse(1), analyse(2), analyse(3), and analyse. To the right of this list is a large, dark, textured image, likely a preview of the scanned document.