Documentation technique

Structure dossiers:

assets:

Fichier css pour styliser certains élements

assets/data :

Fichiers de données généraux pour alimenter le tableau de bord + modèle sous format pickle

history :

Dossiers de fichier pour alimenter les dataTables du menu historique

map :

« communes-departement-region.txt » : Fichier avec détails géographiques des communes

assets/settings :

Fichier json afin de garder une trace des configurations choisies par l'utilisateur

callbacks :

« main.py » : Fichier regroupant l'ensemble des callbacks des différents controllers

controllers :

Un ensemble de controllers gérant l'application avec de nombreux callbacks.

Ils récupèrent les données des fichiers et les affichent sur la vue, puis quelques fois récupèrent les entrées utilisateurs sur la vue pour modifier les données des fichiers.

layouts :

« main.py » : Fichier composé de plusieurs vues afin de créer le « squelette » d'apparence de l'application.

C'est la première page affichée au build de l'application et au rechargement de la page

views :

Un dossier de vue par menu, ces fichiers contiendront le même nombre de fichiers que de sous menus rattachés au menu (chaque fichier représentera une vue)

includes :

Ces vues seront tout le temps affichée sur une partie de l'écran (ici la sidebar présente à gauche contenant les menus)

Architecture et fonctionnement de l'application :

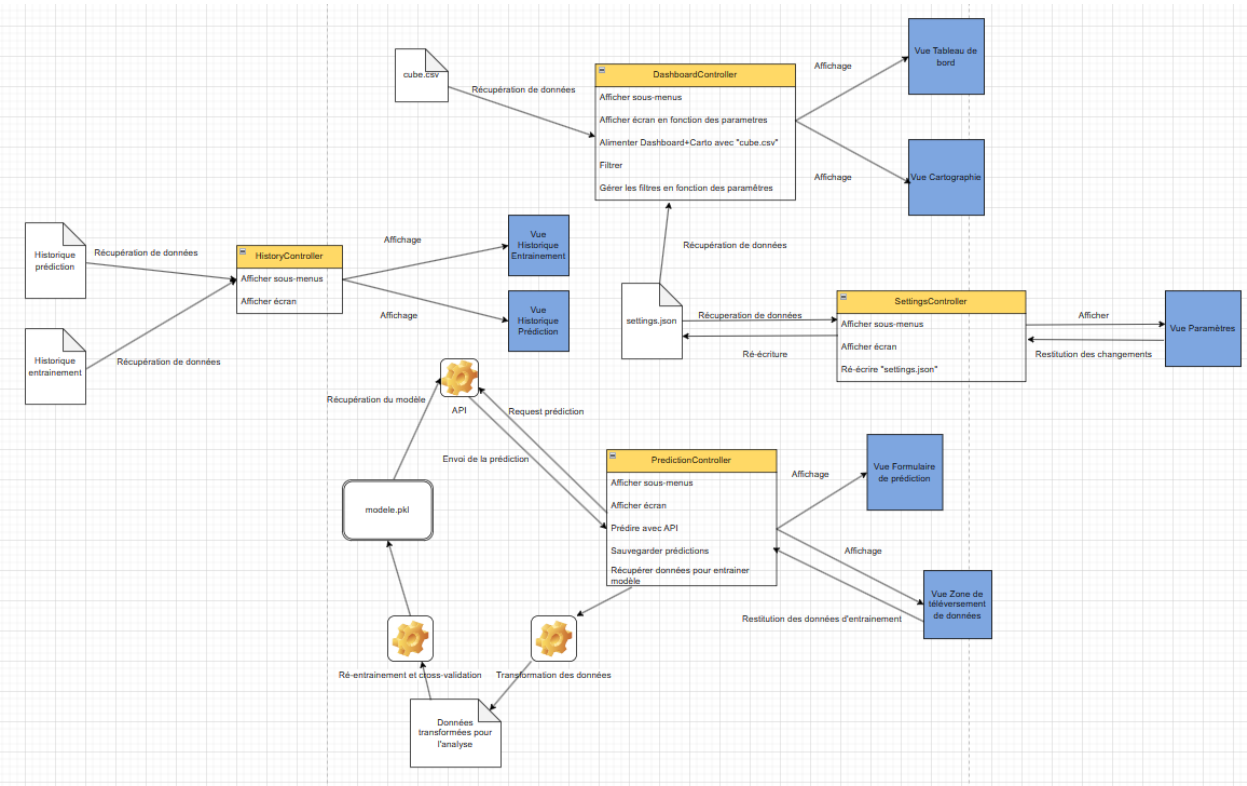

Schéma de l'architecture de l'application

Fonctionnement général :

Les controllers gèrent toutes la logique et ont tous les callbacks. Chaque controller est rattaché à un menu et gère les fonctionnalités des sous-menu rattaché à celui-ci.

Fonctionnement menu :

La sidebar sera toujours affichée, il y a un div principal à droite.

Au moment du click sur un menu, un div est ajouté en dessous de ce menu, décrivant les sous menus du menu cliqué.

Au click d'un sous menu, un div enfant est rajouté dans le div principal avec le contenu du sous menu.

Fonctionnement Tableau de bord :

Récupération des données transformées pour alimenter les graphiques

Groupement des données et décompte des nombre de ventes pour chaque rapport

On récupère le fichier json avec les paramètres.

Si le paramètre ordonner les graphiques est à true, on met les graphiques dans un div en display flex

Fonctionnement filtres :

Une fonction donne le jeu de données filtrés par la combinaison de tous les filtres, puis on met à jour les graphiques

Fonctionnement filtres intelligents :

On récupère le fichier json avec les paramètres.

Si le paramètre filtre intelligents est à true, quand l'utilisateur sélectionne une valeur de filtre, on filtre les options des autres filtres

Fonctionnement Drill-down/Roll-up « temporalité » :

On garde une trace du niveau de temporalité actuel et on groupe par le niveau de temporalité suivant le décompte des ventes pour la valeur supérieure choisie.

Fonctionnement Carte :

On récupère le fichier json avec les paramètres.

On groupe les points du scatter geo par ce niveau de détail et l'on met à jour la carte.

Fonctionnement Prédiction :

On vérifie que les données explicatives renseignées par l'utilisateur sont valides.

On récupère le modèle de prédiction.

On prédit sur les données renseignées par l'utilisateur.

Si l'utilisateur clique sur sauvegarder on récupère le fichier savedPrediction.csv et on rajoute une ligne avec les nouvelles prédictions

On ré-écrit le fichier

Fonctionnement Entrainement :

On vérifie que l'extension du jeu de donnée est conforme, si oui on affiche l'aperçu

On vérifie que les variables importantes ne sont pas nulles

On vérifie que les communes sont valides.

On vérifie le type des données.

Si tout le jeu de données à insérer est intègre, on fait tourner les transformations pour pouvoir entrainer le modèle et on intègre ces données aux autres données pour entrainer le modèle.

On ré-entraine sur plusieurs modèles. On stock le meilleur modèle comme notre modèle de prédiction.

On récupère le fichier csv qui garde une trace des données insérées et on rajoute une ligne historisant l'insertion du fichier puis on le ré-écrit.

Dans train-details on créé un csv ayant pour nom du dernier ID du fichier maitre et on y stock les données insérés

Fonctionnement Historique Prediction :

On affiche le csv traquant l'historique de prédiction sous forme de DataTable

Fonctionnement Historique Entrainement :

On affiche le CSV historisant les insertions sous forme de DataTable. Si l'utilisateur clique sur une ligne de la DataTable on récupère le csv du fichier concerné (grâce à son ID) puis on affiche le détail du fichier sous forme de DataTable

Fonctionnement Settings :

On récupère le fichier json concernant les paramètres en montrant les paramètres choisis précédemment.

Si l'utilisateur fais des modifications et clique sur Sauvegarder, on ré-écrit le fichier json pour prendre en compte les modifications de paramètres