Construisez vos pipelines de données avec Databricks

Déployez une application avec Databricks Apps et Streamlit

Correction

La modification et son effet. La ligne attendue est `SEUIL_BAISSE = 0.3`, en haut du fichier, et rien d'autre. Pourquoi cette ligne et pas le curseur ? Parce que la constante est la valeur par défaut : c'est ce que Claire verra à l'ouverture, avant de toucher à quoi que ce soit. Modifier les autres arguments de `st.slider` ne changerait que les bornes ou le pas du réglage. L'effet attendu : la liste au seuil par défaut s'allonge, puisqu'une baisse de trente pour cent inclut toutes les baisses de cinquante. Un bon rendu contient aussi une vérification croisée : la même requête lancée dans l'éditeur SQL avec le seuil à 0,3 donne le même nombre de lignes que l'application.

Résultat attendu : 216 abonnés au seuil 0,5, 290 au seuil 0,3, dans l'application comme dans l'éditeur SQL.

Le redéploiement qui ne donne rien. La cause la plus probable : avoir créé une seconde application au lieu de redéployer la première, puis ouvert l'URL de l'ancienne. Supprimez la seconde, redéployez `frequence-decrochage`. Deuxième cause : l'application s'est arrêtée (24 heures après son dernier démarrage) et attend un **démarrage**, pas un redéploiement. Troisième : une modification involontaire ailleurs dans le fichier, visible dans le journal par une erreur de syntaxe Python à la ligne indiquée.

Le nouveau besoin : un suivi des écoutes par région, chaque lundi. Déroulons les quatre critères. **Type de besoin** : suivre. Les responsables veulent regarder un chiffre connu, chaque semaine, pas déclencher une action ni régler une logique. **Public** : plusieurs responsables, peu techniques, qui liront un écran sans écrire ni relire une requête. **Enjeu de fiabilité** : moyen ; un suivi d'activité, pas un chiffre de comité, mais récurrent, donc à définir une fois pour toutes. **Coût de maintenance** : un dashboard se met à jour seul, une application demande un redéploiement à chaque évolution.

Brique recommandée : **le dashboard**, avec un filtre de région ajouté à ceux d'offre et de période, et un envoi planifié le lundi matin (**Schedule** et abonnés par e-mail des dashboards AI/BI). Attention : la région vient de `utilisateurs`, la table restreinte. Ce nouveau graphique change donc la réponse du chapitre P2C1 sur le mode de publication : relisez-la avant de publier.

> En Free Edition, le bouton **Schedule** du dashboard publié est disponible : il planifie le rafraîchissement (chaque jour, chaque semaine…) et propose aux abonnés de recevoir le dashboard par e-mail.

Pourquoi pas l'application ? Elle est tentante parce que la liste de décrochage vient d'être livrée et que « par région » ressemble à un paramètre. Mais aucun besoin d'action n'apparaît dans la demande, et le filtre de région est exactement ce qu'un dashboard fait bien. Vous ajouteriez du code à maintenir pour un bénéfice nul. Pourquoi pas Genie ? Les questions ne changent pas d'une

semaine à l'autre, et les responsables régionaux ne vérifieront pas le SQL : un suivi récurrent se fige dans une requête validée, il ne se repose pas chaque lundi à un assistant.

Deux réponses restent acceptables si elles sont argumentées : un dashboard enrichi d'un agent Genie à côté, pour les questions ponctuelles que les responsables poseraient en plus ; ou l'application, si l'énoncé fourni contient une action que vous avez repérée et nommée. Le piège à repérer dans votre propre rendu : une recommandation qui décrit la brique choisie sans dire pourquoi les deux autres conviennent moins. L'arbitrage, c'est l'élimination argumentée, pas le choix seul.