Déployement d’un modèle de classification des tweets en respectant une démarche MLOps

Rappel de la problématique

La société fictive Air Paradis souhaite créer un modèle de classification des tweets pour connaître l’opinion de ses clients.

Toutefois la direction à décidé de se servir de données publiques pour le développement et l’entrainement du prototype.

Afin de présenter le prototype à l’équipe marketing, le prototype devra inclure le modèle en lui-même, une interface utilisateur qui sera utilisée en local et qui permet à l’utilisateur de donner un retour sur la prédiction du modèle, un système de monitoring du modèle pour s’assurer que celui-ci ne commet pas trop d’erreur et envoyer automatiquement une alerte dans le cas contraire

1. Exploration et préparation des données
2. Comparaison des modèles avec MLFlow
3. Déploiement du meilleur modèle sur une VM Azure
4. Création d’un UI
5. Monitoring du Modèle
6. Exploration et préparation des données

Avant de commencer la modélisation, nous devons nous familiariser avec les données. Ici, nous avons 80 000 tweets avec un sentiment positif et 80 000 avec un sentiment négatif.

En regardant les données de plus près, nous pouvons noter quelques doublons. Une fois supprimés, nos données sont très légèrement déséquilibrées 79 000 pour 80 000. Ce faible déséquilibre ne devrait pas causer de biais dans notre modèle.

(image distribution)

Il est préférable de ne garder que le corps du tweet et la classification pour l’entrainement. Le reste des données comme le nom d’utilisateur risquent d’induire le modèle en erreur.

La préparation de notre texte pour les trois modèles à tester nécessite plusieurs étapes. Dans un premier temps, nous allons retirer les tags utilisateurs et les urls.

1. Comparaisons des modèles avec MLFlow

Les trois modèles que nous allons comparer dans ce projet sont : une régression logistique, un modèle de réseau de neurones, et un modèle BERT

Les performances de ces modèles seront enregistrées avec MLFlow. Cet outil permet enregistre les données des modèles et offre un dashboard afin de faciliter la comparaison des modèles testées.

Les données que nous allons fournir pour nos trois types de modèles ne sont pas les mêmes :

Pour la régression logistique, la préparation des données va inclure un bag of words (CountVectorizer ou Tf-IDF)

Pour les deux modèles de deep learning, les données vont subir un embedding.

1. Déploiement du modèle sur une VM Azure

Le modèle va être déployé sur une API stocké dans une VM Azure. L’API consiste en deux Endpoints.

* Le premier endpoint est celui qui fait tourner le modèle. Lorsque la requête POST est reçue, l’API envoit le sentiment associé au tweet
* Le second Endpoint permet de recevoir le feedback de l’utilisateur et de stocker ce dernier.

Plutôt que de simplement stocker la valeur du feedback et de compter le nombre de feedback similaire. Il a été choisi de stocker plus d’information sur l’usage de l’API. En plus du feeback, la date, le tweet, et le sentiment prédit sont stockées afin de pouvoir assurer un meilleur suivie du modèle par les équipes techniques.

1. Création d’une UI

Afin d’intéragir avec le modèle, nous devons créer une interface utilisateur. Le choix se porte sur Streamlit car cette bibliothèque permet de créer l’interface en quelques lignes de code Python.

Notre interface nécessite plusieurs éléments. Une zone de saisie du texte pour taper ou coller le tweet à envoyer au modèle, un message indiquant que le tweet est en cours d’analyse, les résultats, ainsi qu’un bouton permettant à l’utilisateur de donner un retour au modèle pour que ce dernier puisse connaître les prédictions inexactes.

Notre interface va communiquer deux fois avec l’API pour le besoin du projet :

* Une première requête post sera envoyé après la saisie du tweet lorsque l’on clique sur Prediction
* Une seconde requête post une fois que l’utilisateur a donné un feedback sur la prédiction

Pour les besoins de ce projet, l’ui ne sera pas déployée sur le cloud mais uniquement en local.

Pour démarrer l’interface streamlit nous devons saisir la ligne de commande suivante :

Streamlit run user\_interface.py

Attention : Le terminal utilisé pour démarrer l’application Streamlit doit rester ouvert pendant l’usage de celle-ci.

1. Suivi du modèle

Lorsque l’utilisateur fait un retour au modèle, celui-ci compte le nombre de prédiction incorrectes.

Si le modèle effectue trois erreurs à la suite, une alerte est envoyée.