

Note de Blog : Présentation projet analyse de sentiment approche Mlops et CI/CD

Dans le cadre de la mission confiée par Air Paradis visant à développer un prototype d'IA capable de prédire le sentiment associé à des tweets, la classification automatique des données est essentielle pour anticiper les bad buzz et améliorer la réactivité aux tendances sur les réseaux sociaux. Cet article détaille trois approches principales de classification basées sur différentes techniques d'embedding (TFIDF, Word2Vec, GloVe, DistilBERT) et leur intégration avec divers modèles de classification pour prédire le sentiment des tweets (Régression Logistique, Random Forest, LightGBM).

Qu'est-ce qu'un Modèle d'Embedding et Pourquoi l'Utilise-t-on ?

Un modèle d'embedding est une technique qui transforme du texte en représentations numériques (vecteurs), permettant aux modèles de machine learning de comprendre et traiter des données textuelles. Ces vecteurs capturent les relations sémantiques entre les mots, en plaçant des termes similaires dans des zones proches dans un espace multidimensionnel. Par exemple, dans Word2Vec, les mots "avion" et "aéroport" peuvent être situés à proximité car ils sont souvent utilisés ensemble.

L'embedding seul ne permet pas de prédire directement le sentiment d'un tweet. C'est pourquoi un modèle de classification est nécessaire. Une fois le texte converti en vecteurs, ces embeddings sont utilisés comme entrée pour des modèles comme la régression logistique ou les forêts aléatoires. Ces modèles apprennent à associer des vecteurs à des étiquettes de sentiment (positif, négatif, neutre) grâce à un ensemble de données annotées.

Nous explorerons également la démarche MLOps adoptée pour le déploiement et la gestion de ces modèles.

Présentation et Comparaison des Approches

1. Modèle sur Mesure Simple : TFIDF + Classifieurs Classiques

TFIDF (Term Frequency - Inverse Document Frequency) est une méthode de pondération des mots basée sur leur fréquence dans un document par rapport à l'ensemble du corpus. C'est une approche rapide et facile à mettre en place.

- **Modèles testés :**
 - Régression Logistique
 - Random Forest
 - LightGBM
- **Avantages :**
 - Rapidité d'exécution et faible coût de calcul
 - Bonne performance pour des données bien structurées

- **Inconvénients :**
- Ne capture pas les relations sémantiques entre les mots

2. Modèle sur Mesure Avancé : Word2Vec et GloVe

Word2Vec et **GloVe** sont des techniques d'embedding plus avancées qui capturent les relations sémantiques entre les mots en les projetant dans un espace vectoriel.







































- **Modèles testés :**
- Régression Logistique
- Random Forest
- LightGBM
- **Avantages :**
- Meilleure représentation du sens des mots
- **Inconvénients :**
- Temps d'entraînement plus long
- Nécessite un volume de données plus important

3. Modèle Avancé BERT : DistilBERT

DistilBERT est une version allégée de BERT (Bidirectional Encoder Representations from Transformers) qui conserve une grande partie des performances tout en réduisant les besoins en calcul.

- **Modèles testés :**
- Régression Logistique
- Random Forest
- LightGBM
- **Avantages :**
- Excellente performance sur des tâches complexes
- Capacité à capturer des relations contextuelles fines
- Possibilité d'utiliser différentes optimisations du modèle BERT via l'intégration de modèle disponible sur hugging face
- Possibilité d'utiliser le modèle avec un CPU ou GPU
- **Inconvénients :**
- Temps de calcul important
- Besoin de GPU pour accélérer l'entraînement

Comparaison des différents modèles

						Metrics
<input type="checkbox"/>	Run Name	Created	Duration	Source	Models	accuracy 
<input type="checkbox"/>	 TFIDF_LogisticRegression	 5 days ago	5.0s	 mlflow_model_tfidf.py	 sklearn	0.7706061818545...
<input type="checkbox"/>	 Word2Vec_LightGBM	 5 days ago	19.7s	 mlflow_model_word2vec.py	 sklearn	0.7285998299489...
<input type="checkbox"/>	 Word2Vec_LogisticRegression	 5 days ago	8.3s	 mlflow_model_word2vec.py	 sklearn	0.7251956837051...
<input type="checkbox"/>	 TFIDF_LightGBM	 5 days ago	23.7s	 mlflow_model_tfidf.py	 sklearn	0.7211382164649...
<input type="checkbox"/>	 DistilBERT_LogisticRegression	 3 days ago	9.0s	 mlflow_distilbert.py	 sklearn	0.7166274882464...
<input type="checkbox"/>	 DistilBERT_LightGBM	 3 days ago	19.6s	 mlflow_distilbert.py	 sklearn	0.7139173001900...
<input type="checkbox"/>	 GloVe_LightGBM	 5 days ago	19.9s	 mlflow_model_glove.py	 sklearn	0.7100942782834...
<input type="checkbox"/>	 TFIDF_RandomForest	 5 days ago	38.7s	 mlflow_model_tfidf.py	 sklearn	0.7076654246273...
<input type="checkbox"/>	 Word2Vec_RandomForest	 5 days ago	20.8min	 mlflow_model_word2vec.py	 sklearn	0.70713088926678
<input type="checkbox"/>	 GloVe_LogisticRegression	 5 days ago	7.5s	 mlflow_model_glove.py	 sklearn	0.7054960238071...
<input type="checkbox"/>	 DistilBERT_RandomForest	 3 days ago	20.4min	 mlflow_distilbert.py	 sklearn	0.6915887266179...
<input type="checkbox"/>	 GloVe_RandomForest	 5 days ago	20.8min	 mlflow_model_glove.py	 sklearn	0.6887878863659...

1- Comparaison des modèles en fonction l'accuracy sur MLFLOW

Air Paradis souhaite que nous utilisions un modèle BERT pour répondre aux exigences du projet. C'est pourquoi, pour notre prototype, nous allons mettre en production le modèle DistilBERT avec régression logistique, qui offre un bon équilibre entre précision (71.66%) et temps d'entraînement (9.0s).

Démarche MLOps : Du Développement au Déploiement

Principes de MLOps

Le MLOps (Machine Learning Operations) vise à automatiser et industrialiser le cycle de vie des modèles de machine learning. Cette approche repose sur des principes d'intégration continue (CI) et de déploiement continu (CD), garantissant la reproductibilité, la traçabilité et la robustesse des modèles en production.

Étapes Clés Mise en Oeuvre

1. Suivi des Expériences

MLflow est utilisé pour suivre les expériences, enregistrer les hyperparamètres et évaluer la performance des modèles.

2. Gestion des Modèles

Les modèles sont aussi enregistrés pour pouvoir être directement intégrer à notre application Flask, et pouvoir réutiliser une version du modèle amélioré par la suite.

mlflow2.13.0

ExperimentsModels

GitHub

Docs

Treatment_analysis_model_test

TFIDF_LogisticRegression

Overview

Model metrics

System metrics

Artifacts

Description

No description

Details

Created at

2024-12-27 18:25:50

Created by

semoulolot

Experiment ID

676947414533275617

Status

Finished

Run ID

a1cc74f334be4287ad11c58f1b9d1da

Duration

5.0s

Datasets used

—

Tags

model_type: tfidf_logisticregression

Source

mlflow_model_tfidf.py

Logged models

sklearn

Registered models

—

Parameters (15)

Search parameters

Parameter

Value

C

1.0

class_weight

None

dual

False

fit_intercept

True

intercept_scaling

1

l1_ratio

None

max_iter

1000

multi_class

deprecated

n_jobs

None

Metrics (9)

Search metrics

Metric

Value

accuracy

0.77060611818545563

precision_negative

0.781676798034898

recall_negative

0.747550298238226

f1_negative

0.7642326573624284

support_negative

159101

precision_positive

0.7605648915939887

recall_positive

0.7934180332456484

f1_positive

0.7766441841778471

support_positive

160803

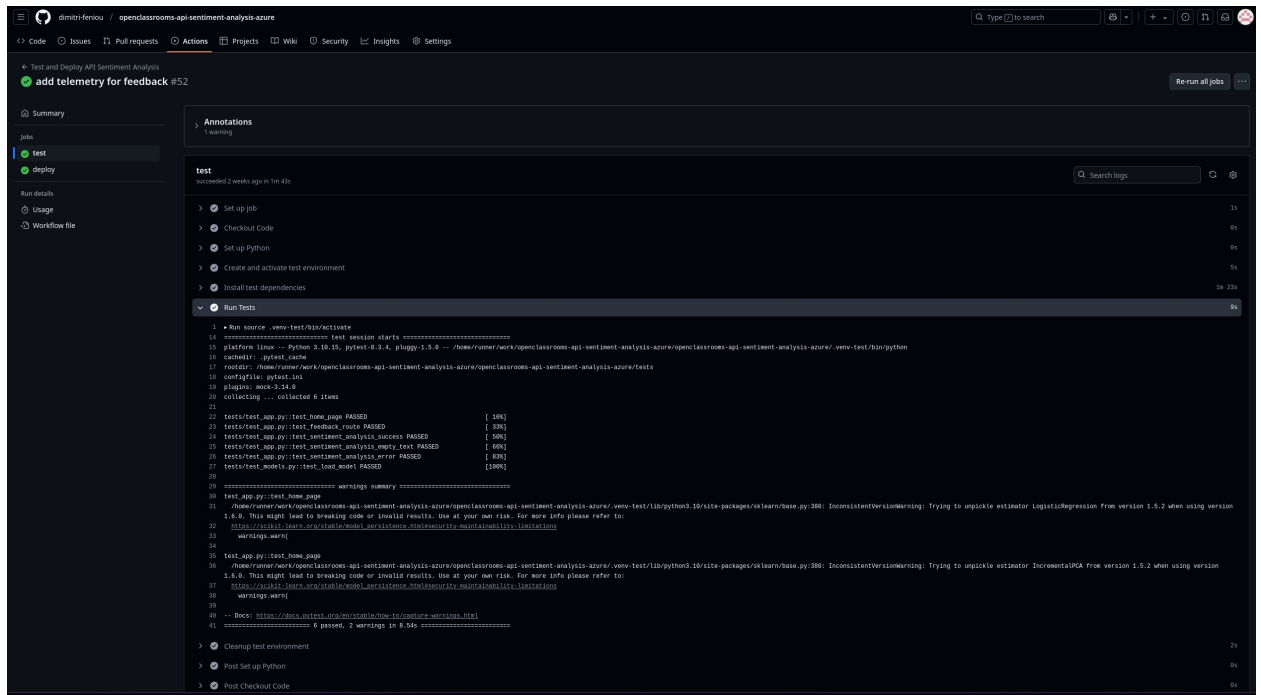
2 Exemple enregistrement des différentes versions des modèles sur mlflow

3. Tests Unitaires

Les tests unitaires sont effectués sur l'API Flask développée pour ce projet. Ces tests couvrent plusieurs aspects critiques du fonctionnement de l'application afin de garantir sa stabilité et sa robustesse. Voici les principaux tests réalisés :

- Test des Routes Flask : Vérification que les routes principales de l'application, comme la route d'accueil, répondent correctement aux requêtes GET et POST.
- Test de la Prédiction du Sentiment : Simulation d'une prédiction de sentiment avec des modèles mockés pour s'assurer que l'API renvoie des résultats cohérents.
- Test de Gestion des Erreurs : Envoi de textes trop longs pour vérifier que l'application renvoie des erreurs appropriées.
- Test de Chargement des Modèles : Vérification que les modèles sont correctement chargés, avec ou sans CUDA disponible.

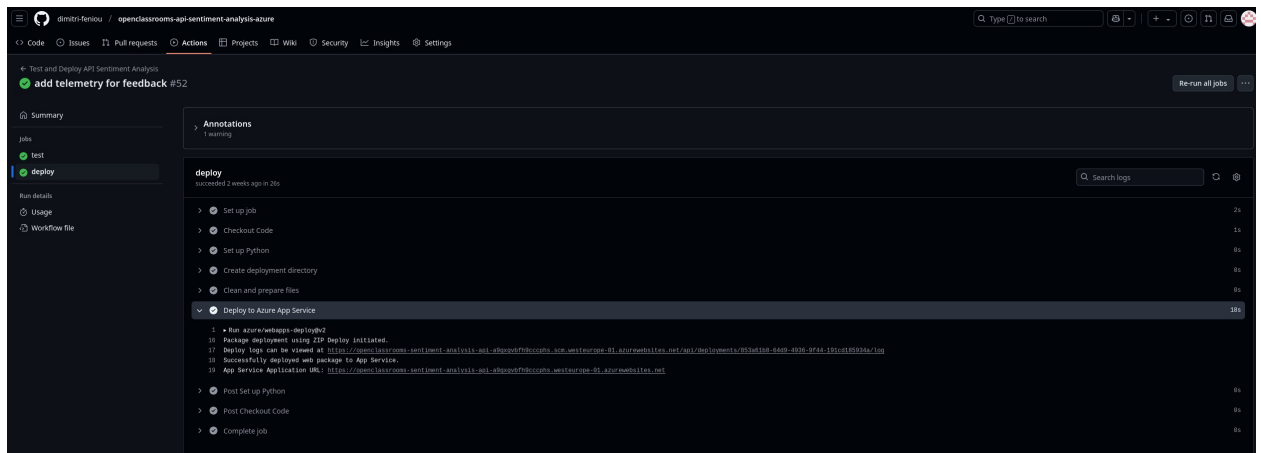
L'ensemble de ces tests est automatisé grâce à GitHub Actions. À chaque commit sur la branche principale, les tests sont déclenchés automatiquement pour garantir que l'application est stable avant tout déploiement



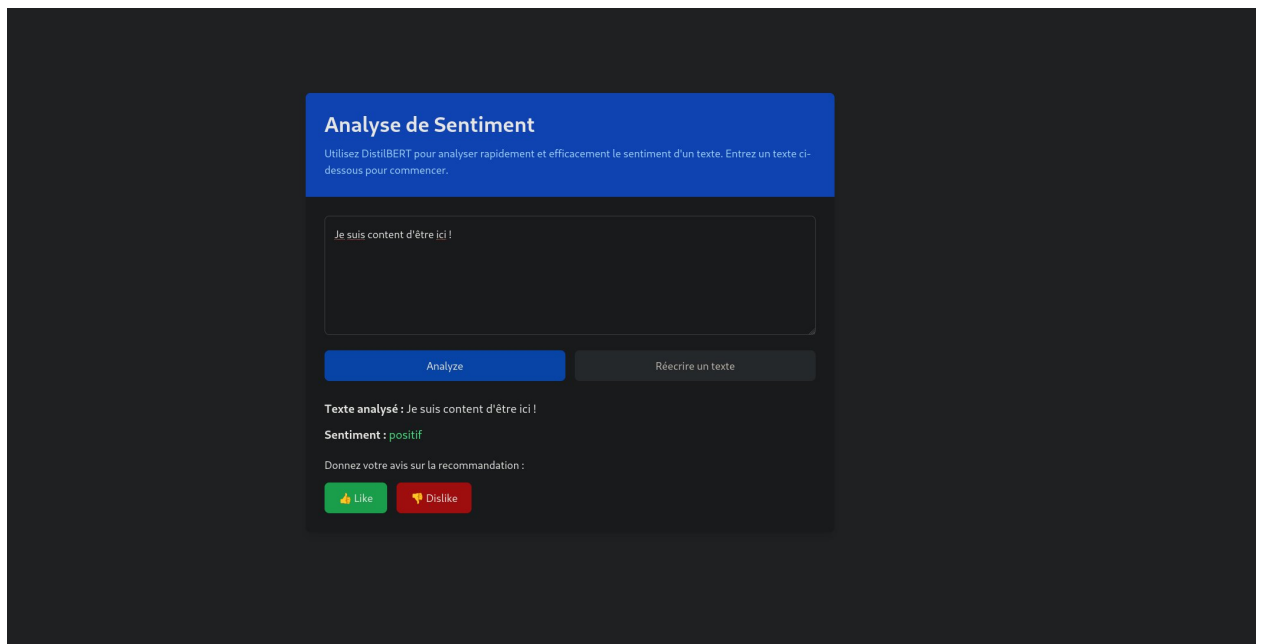
3 Exemple de test unitaire avec GITHUB Actions avant déploiement de l'api

4. Déploiement en Production

Le déploiement en production est automatisé grâce à GitHub Actions. À chaque nouveau commit sur la branche principale, un pipeline CI/CD est déclenché pour tester, construire et déployer automatiquement l'application sur Azure App Service. Cette automatisation garantit des mises à jour rapides et fiables, réduisant ainsi les risques d'erreurs humaines lors du processus de déploiement. L'hébergement de l'application est ensuite géré par Azure App Service.



4 Déploiement de l'api sur azure App service via github actions



5 Application déployer via Azure app service

5. Suivi de la Performance en Production

Pour assurer un suivi efficace de la performance du modèle en production, nous avons mis en place Azure Application Insights, un service qui permet de collecter des métriques et des traces sur les prédictions du modèle. Ce suivi nous permet de détecter rapidement les éventuelles erreurs ou dérives dans les résultats de l'IA.

Voici les principales actions mises en œuvre :

- **Traces des tweets mal prédits** : Lorsque l'utilisateur signale un tweet comme mal prédit, le texte du tweet ainsi que la prédiction faite par le modèle sont remontés dans Application Insights.
- **Déclenchement d'alertes** : Si trois tweets sont mal prédits en l'espace de cinq minutes, une alerte est automatiquement envoyée par SMS ou email aux équipes techniques afin de réagir rapidement.
- **Analyse des statistiques** : Les traces collectées sont analysées afin d'identifier les modèles de tweets qui posent problème. Cela permet d'adapter et d'améliorer continuellement le modèle au fil du temps.

Cette approche garantit une amélioration continue du modèle grâce à un suivi rigoureux des prédictions et à des alertes en temps réel.

Azure Application Insights est utilisé pour surveiller les performances du modèle en production. Des alertes sont configurées pour détecter les dérives.

Amélioration Continue

Pour garantir la performance et la précision du modèle sur le long terme, une démarche d'amélioration continue est mise en place, s'appuyant sur les données collectées en production via Azure Application Insights.

Analyse des Statistiques et Boucle d'Amélioration :

- Collecte des Traces :** Chaque prédiction signalée comme incorrecte par l'utilisateur est enregistrée dans Azure Application Insights, avec les détails du texte du tweet et de la prédiction.
- Analyse des Modèles de Données :** Des tableaux de bord sont créés pour visualiser les tendances des erreurs et identifier les types de tweets qui posent fréquemment problème.
- Identification des Faiblesses du Modèle :** L'analyse des erreurs permet de repérer les motifs récurrents qui révèlent les limites du modèle. Par exemple, des tweets contenant des sarcasmes ou des formulations ambiguës peuvent être mal classifiés.
- Ré-entraînement et Mises à Jour :** Les tweets mal prédits sont ajoutés à l'ensemble d'entraînement pour ré entraîner régulièrement le modèle et corriger ses biais.
- Tests et Validation :** Avant chaque mise à jour du modèle, des tests sont effectués pour s'assurer que les nouvelles versions surpassent les anciennes en termes de précision et de stabilité grâce a mlflow.

Accueil

Journaux

newlines.com,signisight

Nouvelle requête...

sent...

Selectionner une etendue

Interval de temps : Personnalisee

Enregistrer

Partager

Nouvelle regle d'alerte

Exporter

Epingler

Mettre en forme la requete

Tables

Requetes

Fonctions

Rechercher

Rechercher par : Solution

Réduire tout

Favorites

Vous pouvez ajouter des favoris en cliquant sur l'icône

Application Insights

availabilityResults

browserTimings

customEvents

customMetrics

dependencies

exceptions

pageViews

performanceCounters

requests

traces

1

customEvents

2

where name == "FeedbackReceived"

3

where customDimensions.feedback == "dislike"

Résultats

Graphique

timestamp [UTC]	name	itemType	customDimensions	operation_SyntheticSource	client_Type	client_OS	client_IP	client_City	client_StateOrProvince	client_CountryOrRegion	client_browser	appId
10/12/2024 11:14:01.061	FeedbackReceived	customEvent	["feedback":"dislike","predicted_sentiment":"positif","t..."]	Python-utils	Bot	813MP Sun Aug 18 19:16:21 UTC 20...	0.0.0.0	Amsterdam	Noord-Holland	Netherlands	Python-utils 3.10	09350293-3a76-4860-9a64-f...
timestamp [UTC]												
name												
itemType												
customDimensions												
operation_SyntheticSource												
client_Type												
client_OS												
client_IP												
client_City												
client_StateOrProvince												
client_CountryOrRegion												
client_browser												
appId												
appName												
key												
sdkVersion												
itemId												
itemCount												
_ResourceId												

6 Exemple de récupération de feedback via telemetry Azure

Fired:Sev2 Azure Monitor Alert feedback dislike alert on sentiment_analysis_appinsight (microsoft.insights/components) at 12/19/2024 11:15:41 AM

[View the alert in Azure Monitor >](#)

[Investigate >](#)

Summary

Alert name	feedback dislike alert
Severity	Sev2
Monitor condition	Fired
Affected resource	sentiment_analysis_appinsight
Resource type	microsoft.insights/components
Resource group	sentiment-analysis-resource-group
Description	Alerte par mail lorsque les feedback d'analyse de sentiment sont négatif après 3 feedback négatif en 5 minutes
Monitoring service	Log Alerts V2
Signal type	Log
Fired time	December 19, 2024 11:15 UTC
Alert ID	e3a4a902-9fab-da78-3e73-7f55ec0e000e
Alert rule ID	https://portal.azure.com/#resource/subscriptions/f75428b6-6827-4132-8a64-e4a16c46722f/resourceGroups/sentiment-analysis-resource-group/providers/microsoft.insights/scheduledqueryrules/feedback dislike alert

Filtered search results	View query results
Search results	View query results
Search query	customEvents where name == "FeedbackReceived" where customDimensions.feedback == "dislike"
Target resource types	['microsoft.insights/components']

