

Projet 8

# Réaliser un dashboard et assurez une veille technique

Octobre 2024  
Lokman AALIOUI

# Contexte

- Les décisions financières reposent de plus en plus sur des algorithmes de prédiction, Prêt à dépenser, se démarque par sa volonté d'offrir de la transparence à ses clients.
- L'entreprise s'adresse à des individus ayant peu ou pas d'historique de crédit, rendant l'évaluation du risque plus complexe. Pour remédier à cela, un outil de scoring a été développé afin de prédire la probabilité de remboursement et de classer les demandes de crédit en accordées ou refusées.

# Mission



- La mission consiste à concevoir un dashboard interactif permettant aux chargés de relation client d'expliquer de manière claire et transparente les résultats de l'outil de scoring.
- Ce tableau de bord doit non seulement afficher les scores et probabilités de manière compréhensible pour les non-experts, mais aussi offrir une comparaison des données individuelles avec l'ensemble des clients.

# Feuille de route

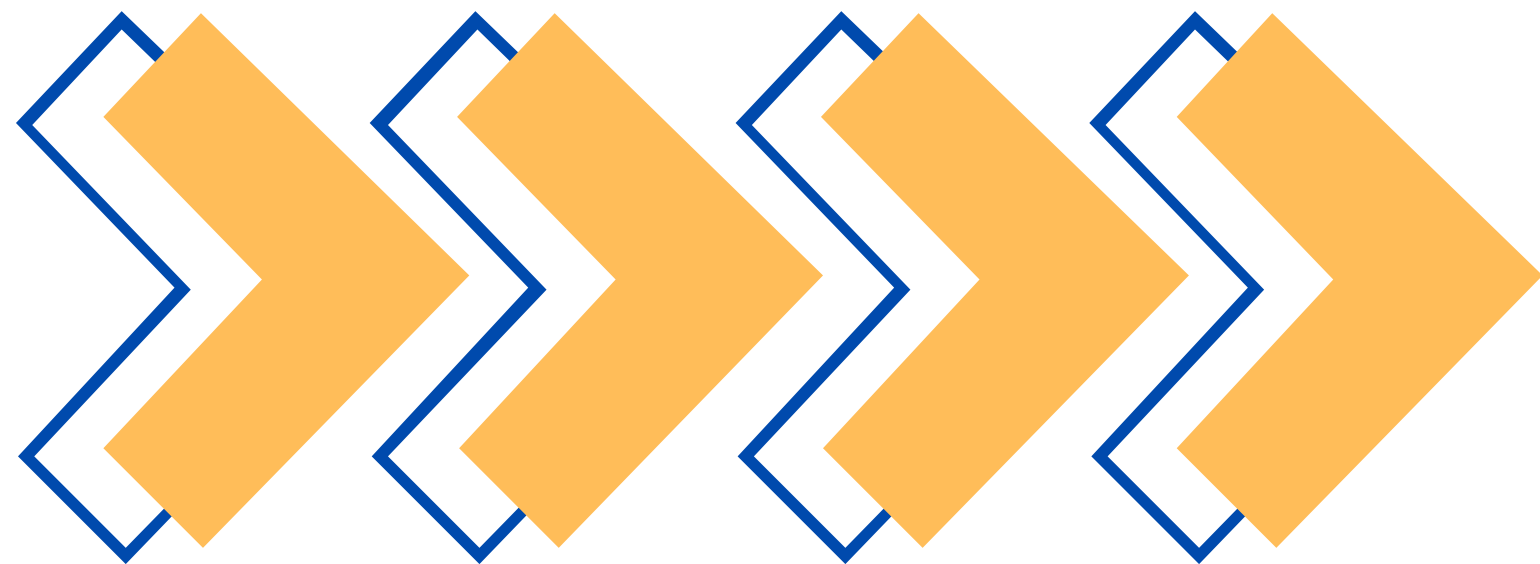


## **Partie 1 : Création d'un dashboard**

1. Mise en place de l'environnement de travail
2. Présentation des fonctionnalités

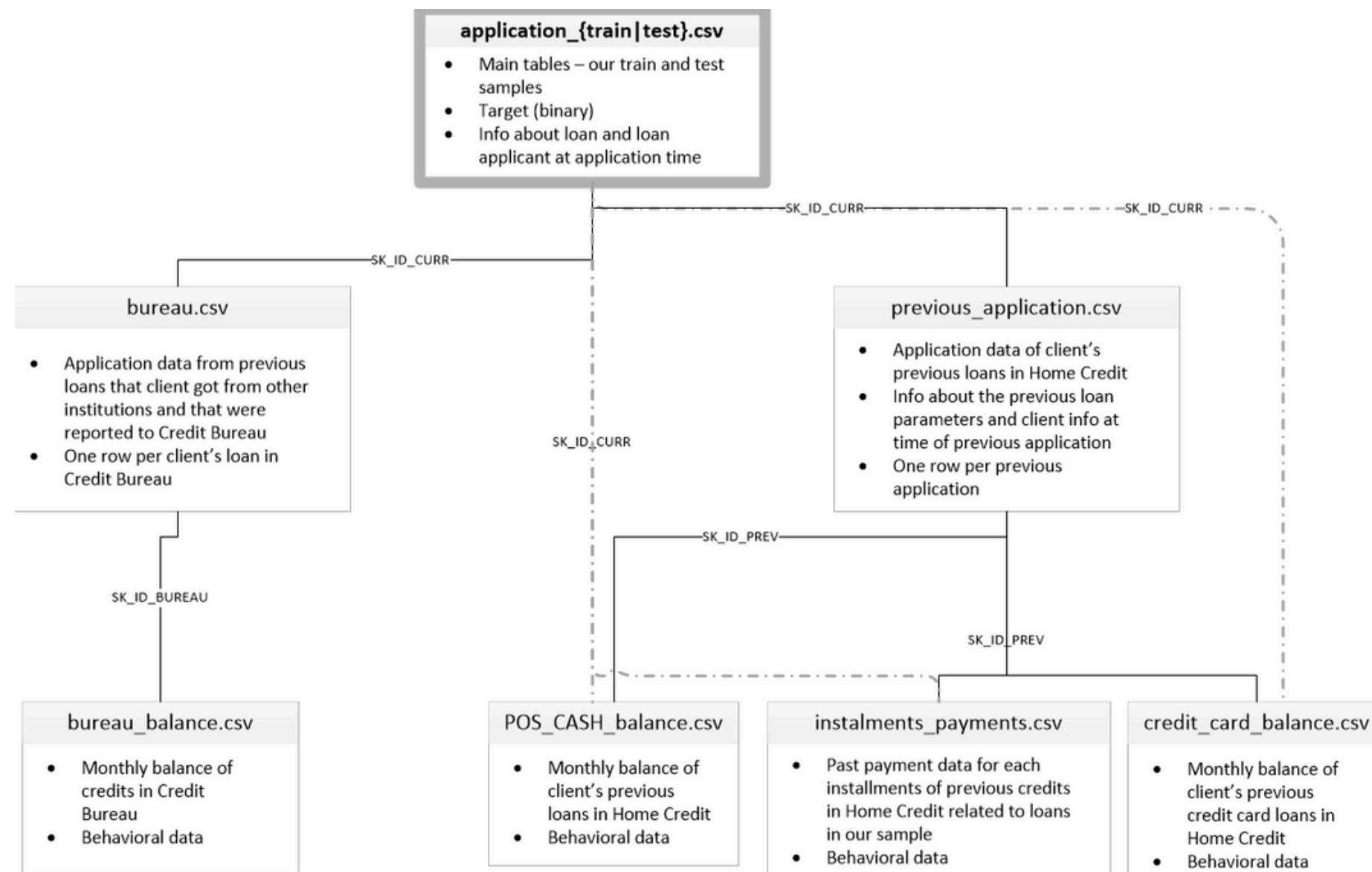
## **Partie 2 : Réalisation d'une veille technique**

1. Contexte
2. Présentation des données
3. Présentation des technologies
4. Test du modèle
5. Conclusions



## Partie 1 : Création du Dashboard

# Les données



**Les données ont déjà été traitées :**

- Nettoyage des données inutiles
- Création de nouvelles features
- Fusion des fichiers en 1

**Plusieurs modèles ont été testés via MLflow :**

- LGBM en SMOTE a été désigné comme plus performant

# Mise en place de l'environnement de travail




- Création de l'app en local
- LightGBM comme modèle
- SHAP pour expliquer le modèle
- Streamlit pour l'interface interactive


- Initialisation du dépôt Git
- Création environnement virtuel
- Test de l'api en local avec Streamlit



- Push du dépôt sur Github
- Création des fichiers Procfile et setupsh pour liaison avec Heroku




- Connexion avec le repository Github via l'interface Heroku
- Déploiement



# Mise en place de l'environnement de travail
















 **Credit-Risk-Dashboard-4** Public


 Pin  Unwatch






 main  1 Branch  0 Tags



 Add file  Code

 <b>Lokmanads</b> Mise à jour streamlit_dashboard.py 30ec352 · 23 minutes ago 11 Commits
 venv Initial commit 4 hours ago
 .gitignore Initial commit 4 hours ago
 LightGBM_smote_tuned.pkl Ajout des fichiers de modèle et des données 4 hours ago
 Procfile Mise à jour Procfile 2 hours ago
 README.md Initial commit 4 hours ago
 df_dashboard_final.csv Ajout des fichiers de modèle et des données 4 hours ago
 df_feature_importance_25.csv Ajout des fichiers de modèle et des données 4 hours ago
 logo.jpg Ajout des fichiers de modèle et des données 4 hours ago
 requirements.txt Mise à jour requirements 2 hours ago
 runtime.txt Ajout runtime.txt 4 hours ago
 setup.sh Mise à jour setup.sh 4 hours ago
 streamlit_dashboard.py Mise à jour streamlit_dashboard.py 23 minutes ago






 Personal  dashboard4   Open app  More

 LokmanAa/Credit-Risk-Dashboard-4  main


Overview Resources **Deploy** Metrics Activity Access Settings



**Add this app to a pipeline**  
Create a new pipeline or choose an existing one and add this app to a stage in it.


**Add this app to a stage in a pipeline to enable additional features**  
Pipelines let you connect multiple apps together and **promote code** between them. [Learn more.](#)  
Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. [Learn more.](#)





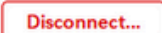
**Deployment method**


 Heroku Git  
Use Heroku CLI



 GitHub  
**Connected** 

 Container Registry  
Use Heroku CLI

**App connected to GitHub**  
Code diffs, manual and auto deploys are available for this app.

Connected to  LokmanAa/Credit-Risk-Dashboard-4 by  LokmanAa  Disconnect...

 Releases in the [activity feed](#) link to GitHub to view commit diffs

 Automatically deploys from  main



# Le tableau de bord

Prêt à dépenser

Client

Identifiant :

100013

Calculer risque

Métriques

Variable univariée :

Score client

Variable 1 (bivariée) :

Score client

Variable 2 (bivariée) :

Score client

## Tableau de bord : risque client

Informations sur le client :

ID client	Prédiction crédit	Score client (sur 100)	Type contrat	Genre	Âge
100,013	Non défaillant	10	Cash loan	H	55

Niveau de risque :



Ecran principal :

- Informations sur le client
- Jauge risque
- Graphiques supplémentaires

Bandeau lateral :

- Sélection du client et des différentes métriques

<https://dashboard4-6d4fea82be3e.herokuapp.com/>

Prêt à dépenser

Client

Identifiant :

100001

Calculer risque

Métriques

Variable univariée :

Score client

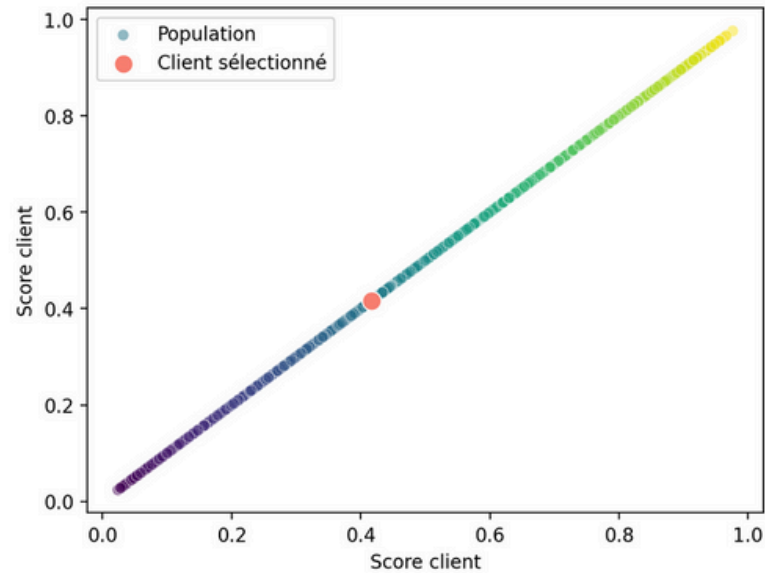
Variable 1 (bivariée) :

Score client

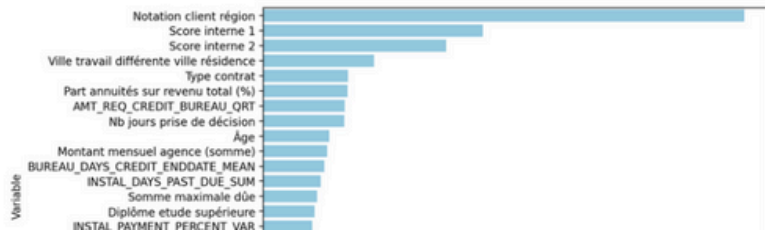
Variable 2 (bivariée) :

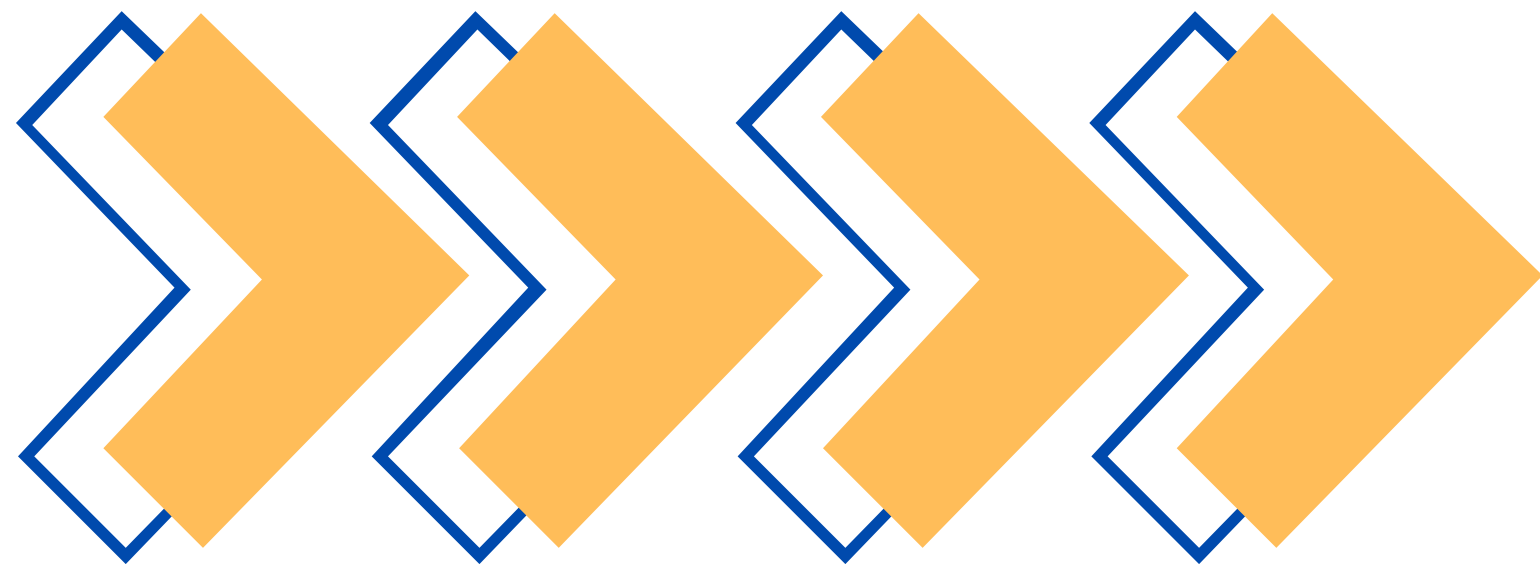
Score client

Analyse bivariee (population complete) :



Importance des variables :





Partie 2 :

## Réalisation d'une veille technologique

# La veille technologique

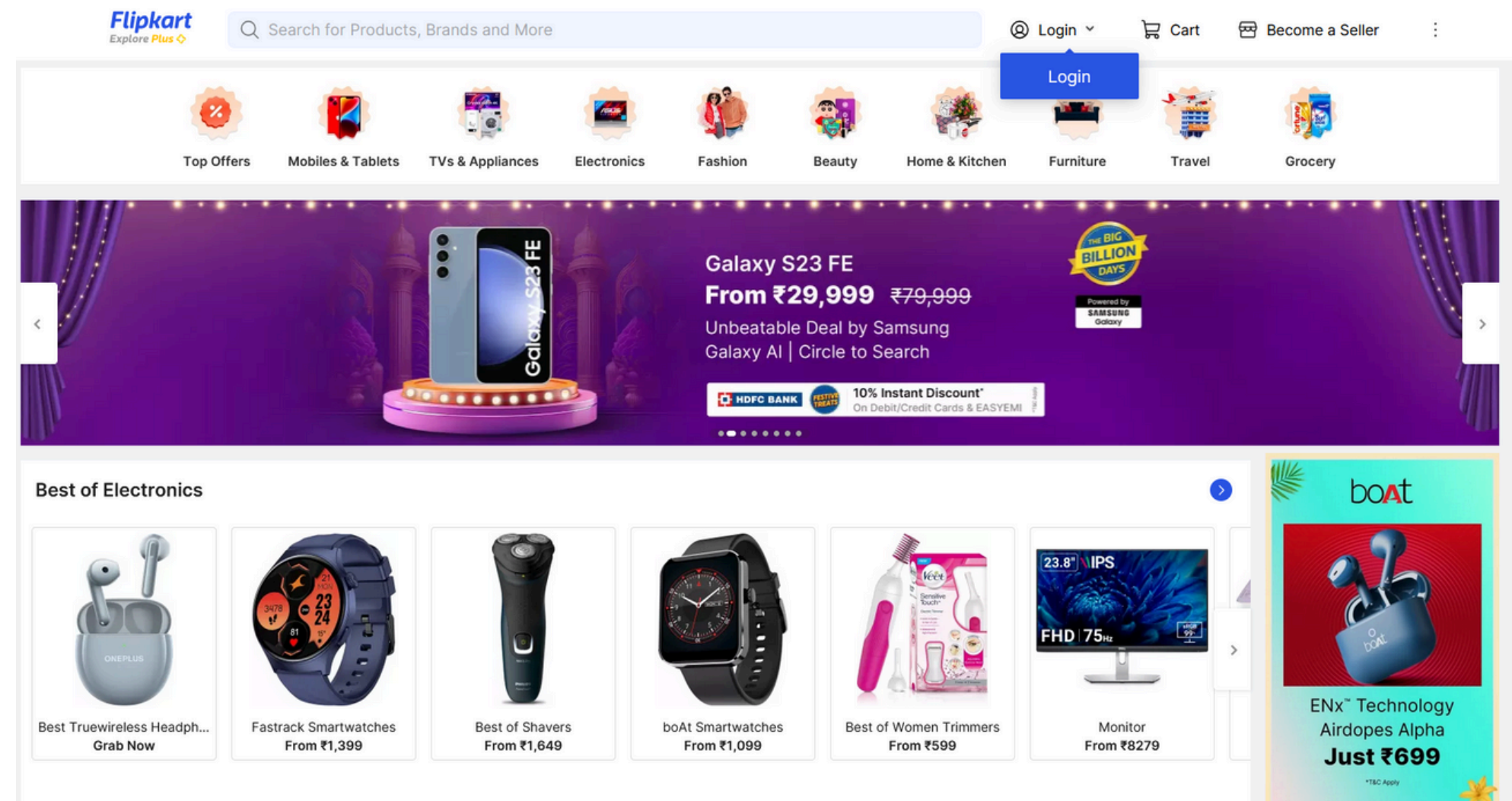


Processus continu de surveillance des innovations, tendances, et évolutions technologiques. La data science est un domaine en constante évolution. La veille permet de :

- Optimiser ses processus
- Anticiper les tendances
- Acquérir des compétences professionnelles
- S'adapter aux besoins du marché

La veille technologique est essentielle pour évoluer dans un domaine aussi dynamique que la data science.

# Les données



Données issues du site Flipkart.com disponibles sur Kaggle.

Fichier CSV :

- ID
- Nom d'produit
- Description
- Etc.

Dossier supplémentaire :

- Images des produits (titre = ID)

# Choix de la technologie



- CTrouver et tester une nouvelle solutions sur le dataset Flipkart vu précédemment Choix du sujet du NLP Traitement similaire du dataset :  
Passage en minuscules  
Tokenisation Retrait des stop words Lemmatisation  
Recomposition des phrases

- Initialisation du dépôt Git
- Création environnement virtuel
- Test de l'api en local avec Streamlit

- Push du dépôt sur Github
- Création des fichiers Procfile et setupsh pour liaison avec Heroku

- Connexion avec le repository Github via l'interface Heroku
- Déploiement

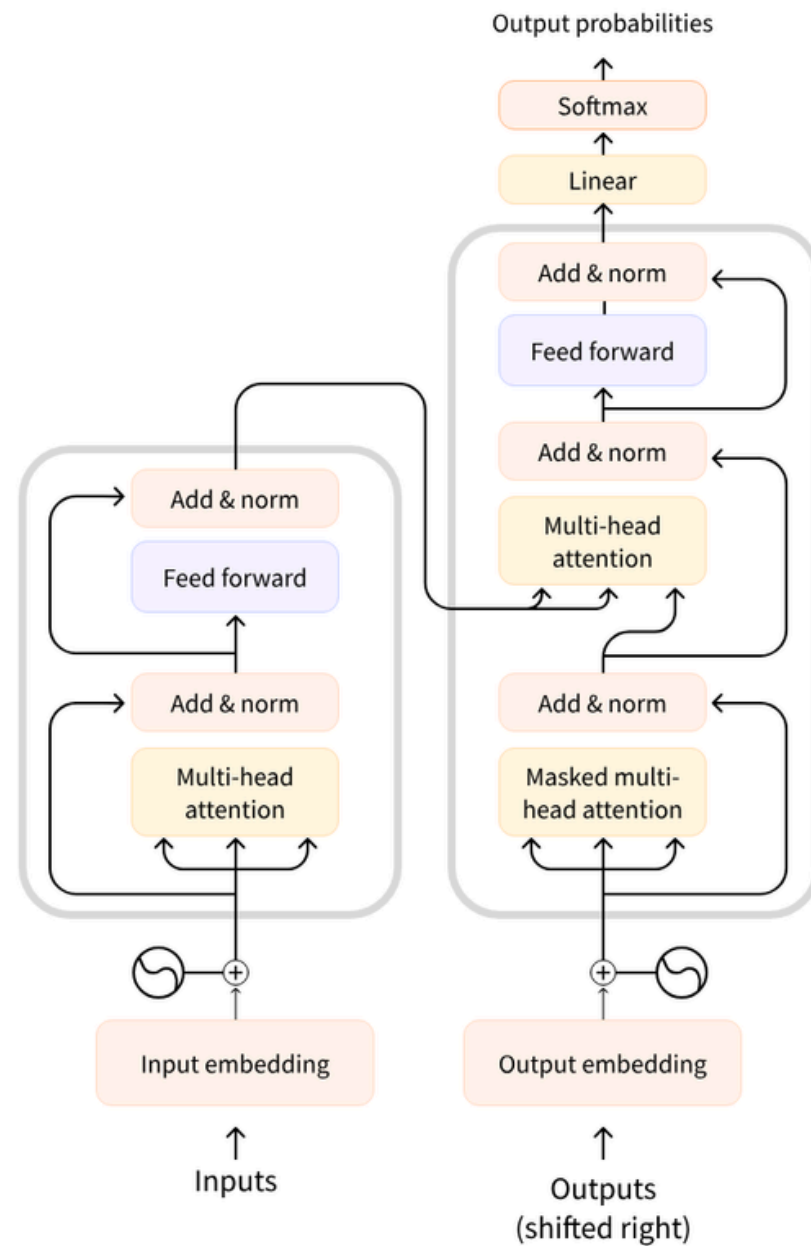
# Choix de la technologie



## Modèle mxbai-embed-large-v1 :

- Publié en mars 2024 par la société MixedBread AI
- Modèle conçu pour capturer des représentations contextuelles riches et profondes dans les phrases
- Supporte une grande variété d'applications NLP (analyse de sentiments, chatbots, etc.)
- Fonctionne en plusieurs langues
- Optimisé pour la rapidité et peu gourmand en ressource
- Populaire dans la communauté open-source

# Modèle mxbai-embed-large-v1

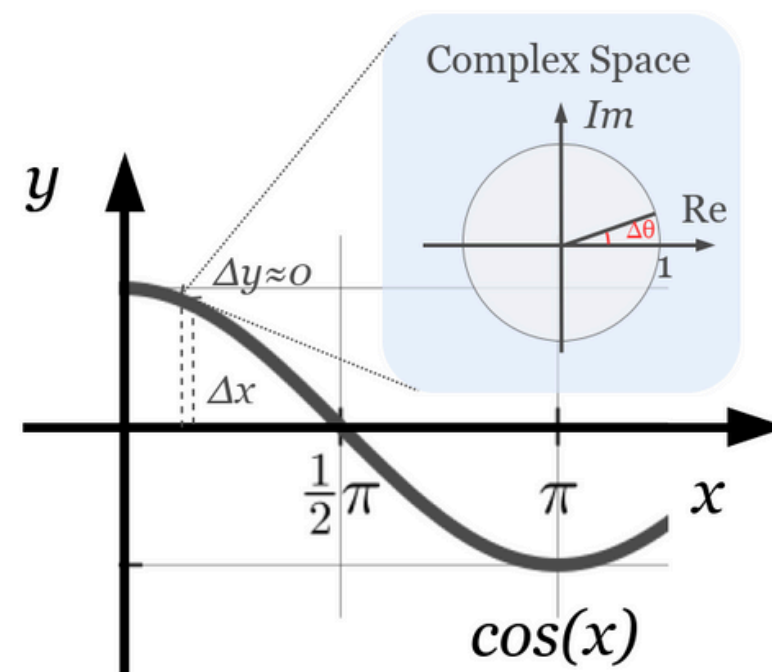
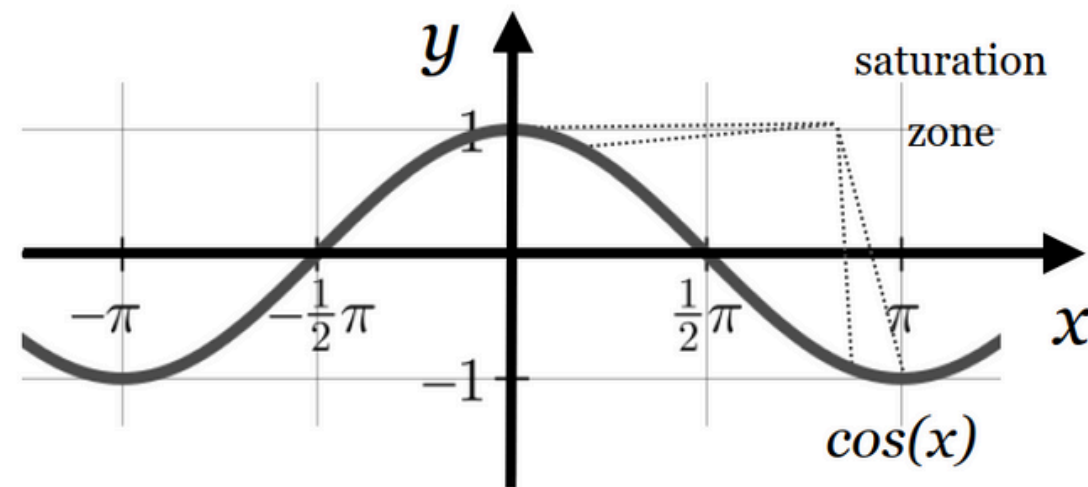


## Fonctionnement :

- Modèle de type Transformers : transforme un texte en vecteur
- Pas de détail officiel sur la structure, mais potentiellement inspiré de E5 ou Matryoshka
- Entraîné sur un vaste jeu de données par paires (similaires ou non)
- Fine-tuning à l'aide de triplet les plus pertinents possible



# Modèle mxbai-embed-large-v1



## Optimisation :

- Optimisation de l'entraînement et du fine tuning grâce à une nouvelle métrique : Angle Loss function
- Fonction cosinus habituellement utilisée présente une forme de saturation
- Correction de cette saturation du gradient
- Mesure d'un angle entre deux vecteurs projetés dans un espace complexe



# Test du modèle



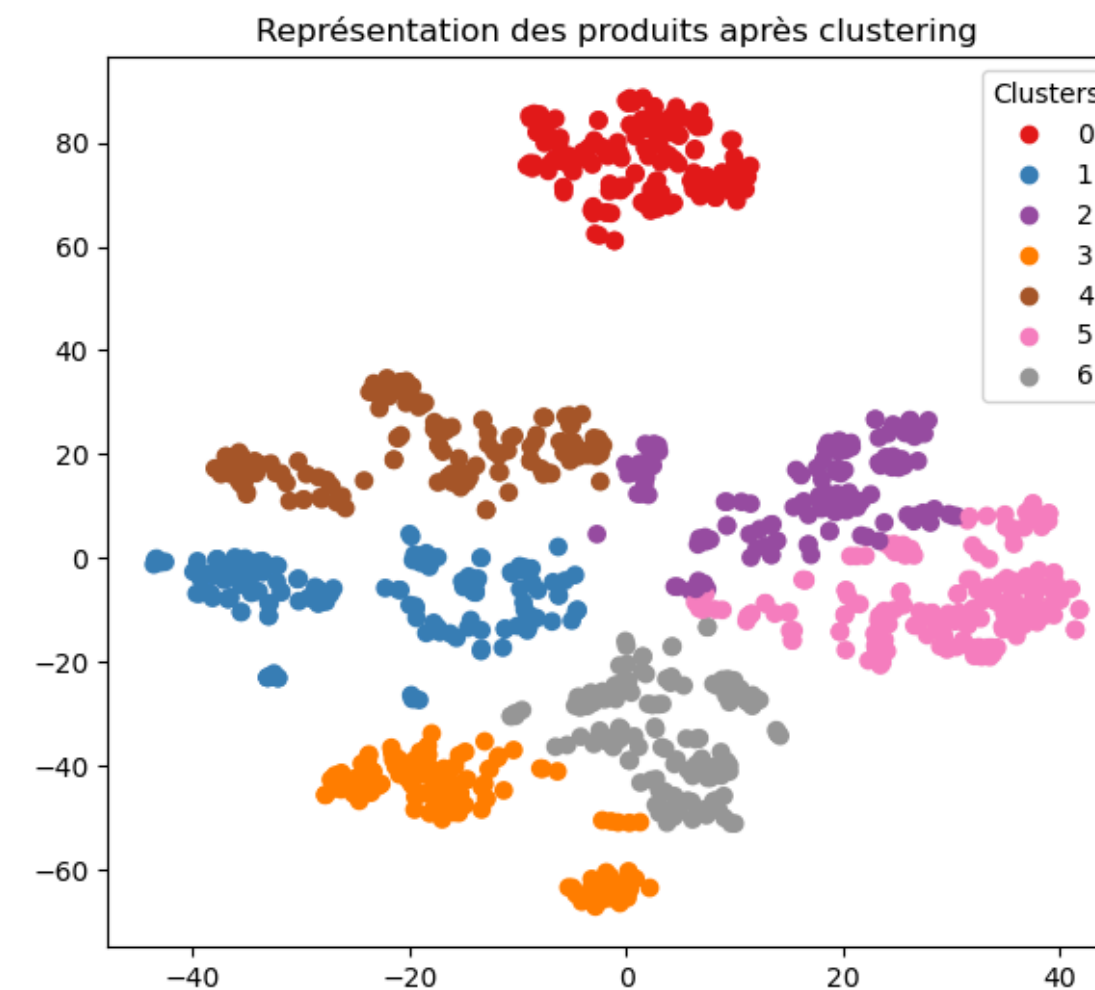
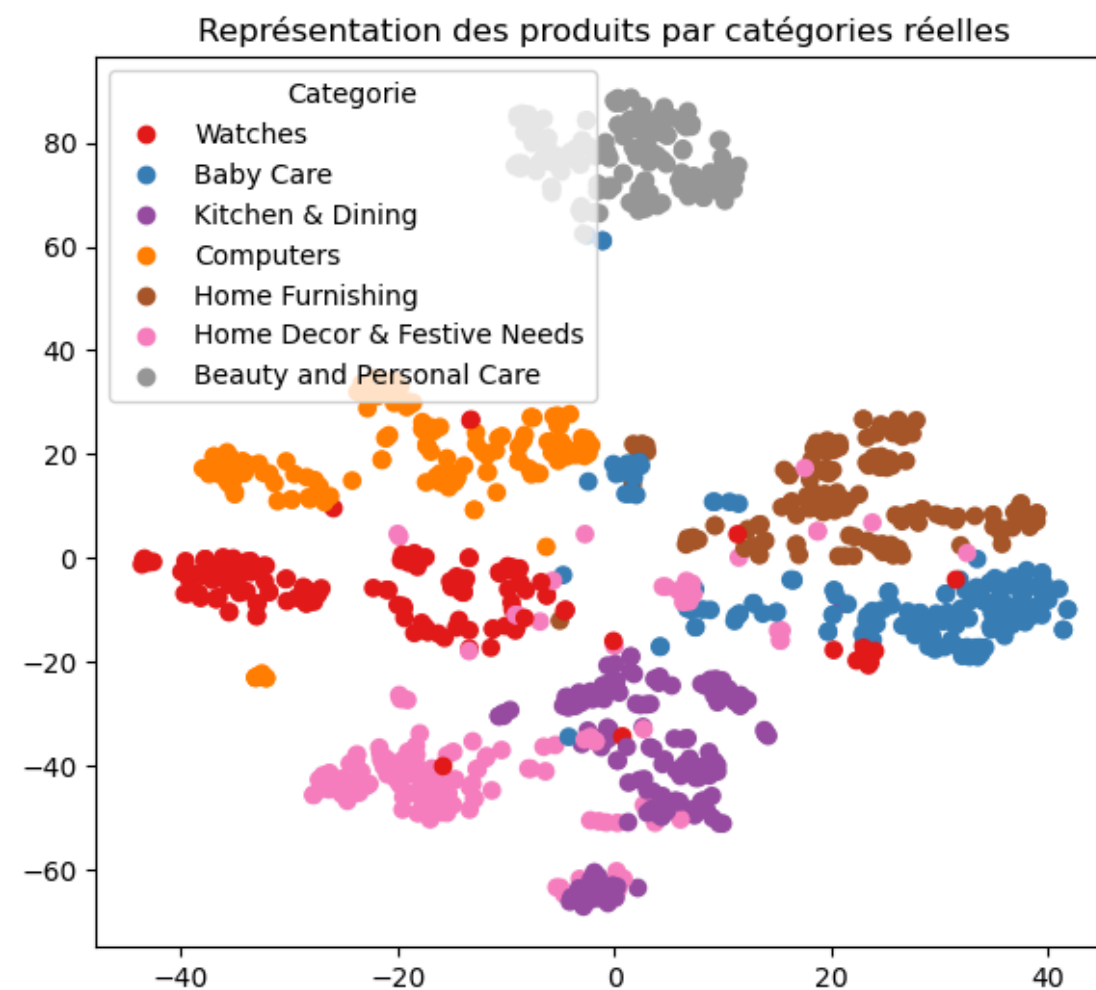
## Préparation :

- Essais sur Ollama et Sentence Transformers
- Différence d'utilisation (OOTB vs objet)
- Génération des embeddings en adaptant le code selon la librairie
- Récupération des vecteurs (1024) dans des arrays NumPy

# Classification

## Scoring :

- Utilisation de la même métrique : score ARI sur un T-SNE
- Pertinence des vecteurs selon les classes
- Essais de classification avec mesure de l'accuracy
- Utilisation d'un classifieur simple d'utilisation (Random Forest)



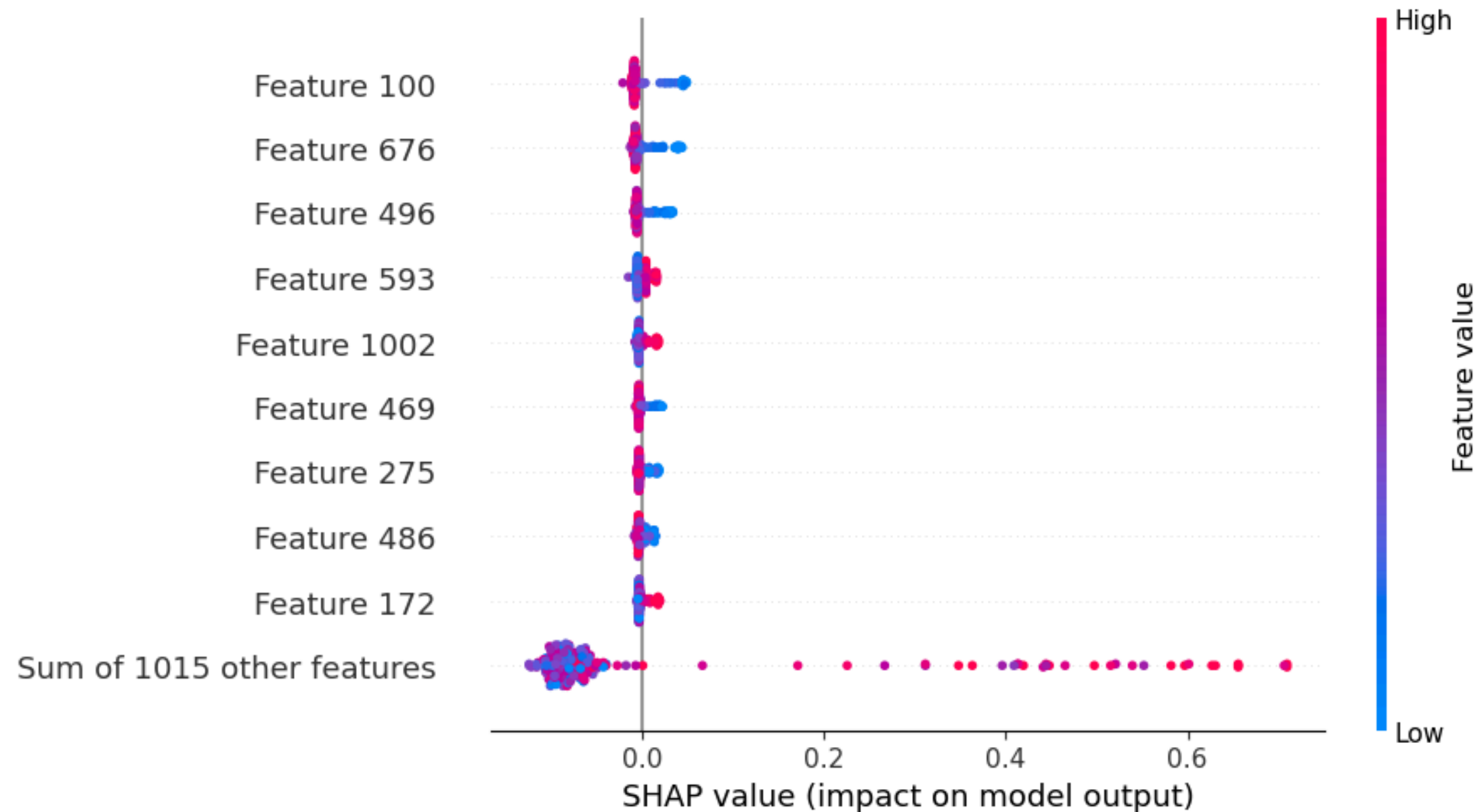
# Résultats

Scoring :

- Ollama est plus performant
- Modèle possiblement mis à jour
- Résultats effectivement bien supérieurs aux modèles testés précédemment

	Ollama	ST (512)	ST (1024)	BERT	USE
Score ARI	<b>0,709</b>	0,686	0,582	0,322	0,443
Accuracy	0,952	0,943	<b>0,962</b>	<b>X</b>	<b>X</b>

# Feature Importance



## SHAP :

- Modèle dit « boîte noire »
- Difficile d'interpréter l'influence des mots sur la génération des vecteurs
- Analyse avec SHAP du classifieur peu pertinente
- Une seule feature (le vecteur)
- Difficile d'établir un lien entre des mots et « l'importance » des dimensions du vecteurs

# Conclusions



## **Tableau de bord :**

- Le dashboard est ergonomique pour les conseillers et simplement interprétable par les clients

## **Veille technique :**

- Comparaison d'un modèle de NLP récent de MixedBread AI avec des méthodes plus classique résultats nettement meilleurs
- Modèle limité à des documents courts (512 tokens maximum)
- Faible interprétabilité du modèle et assez peu de ressources disponibles pour le moment
- Amélioration possible : Utilisation du RAG pour améliorer les performances, ajout de couche de réductions de dimensions / classification pour utiliser des outils d'interprétation pensés pour BERT



Projet 8

Merci pour votre attention

Octobre 2024  
Lokman AALIOUI