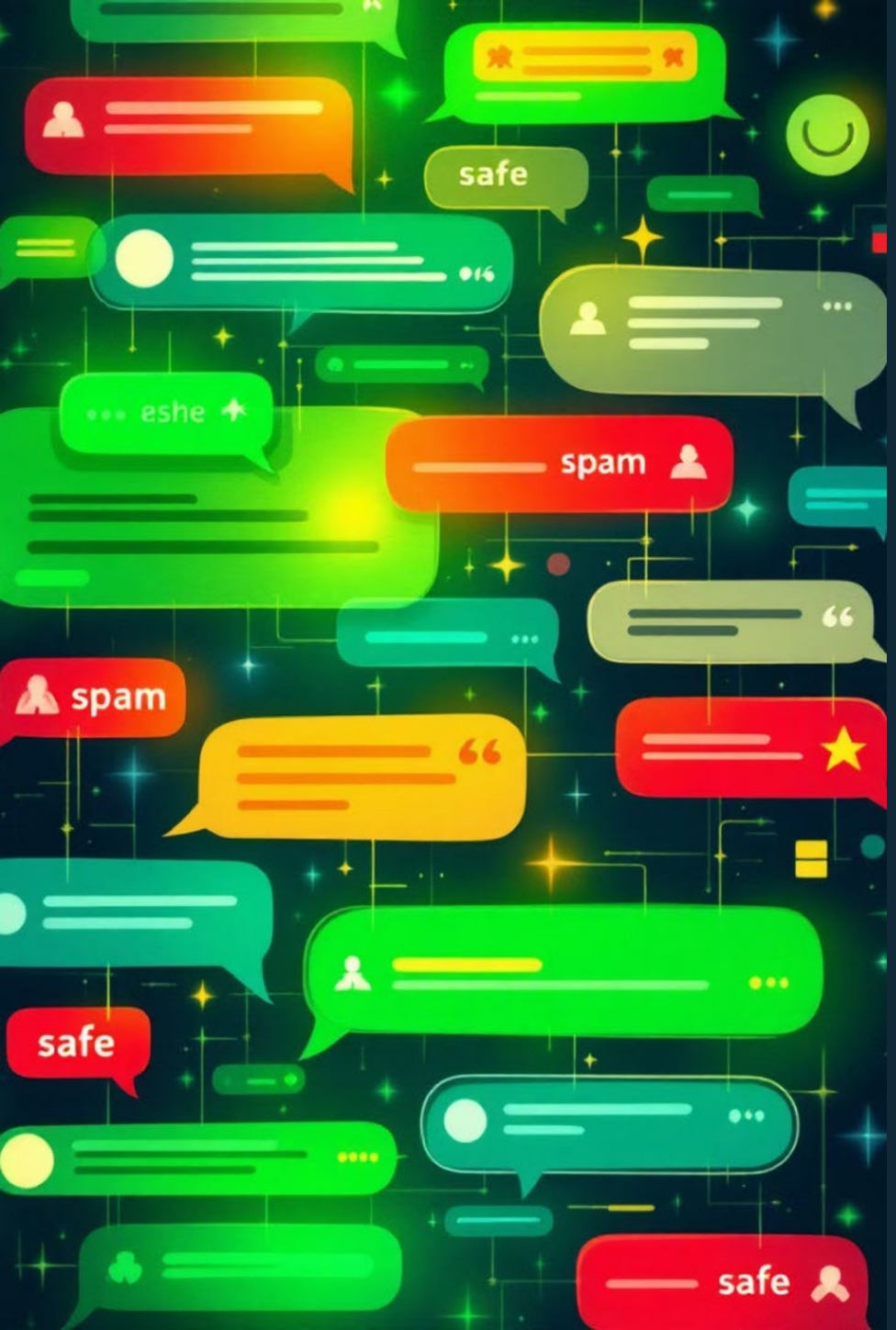

Présentation Technique du Projet SMS Spam Classifier

Classification et Analyse des SMS pour détecter le spam

by Sanollea & Sebastien

Introduction et Objectifs

Introduction au Problème

Importance de la détection du spam dans les SMS.

Conséquences du spam sur la vie quotidienne des utilisateurs.

Objectifs du Projet

Classifier les SMS en "Spam" ou "Non-Spam".

Fournir des métriques d'évaluation pour évaluer la performance du modèle.

Développer une interface utilisateur intuitive.

Contexte du Projet

Besoin d'une solution efficace pour classifier les SMS.

Utilisation de techniques de machine learning pour améliorer la précision.

Bibliothèques Utilisées

os

Gestion des fichiers et des chemins.

pandas

Manipulation de données sous forme de tableaux.

joblib

Sauvegarde et chargement de modèles ou d'objets Python.

sklearn

Contient des outils pour l'apprentissage automatique (modèles, métriques, etc.).

matplotlib et seaborn

Visualisation des données (graphiques et matrice de confusion).

nltk

Traitement du langage naturel (stopwords, lemmatisation).

streamlit

Interface utilisateur pour les applications web interactives.

Structure Principale et Préparation des Ressources

Structure Principale

Constantes et Configuration:

- MODEL_PATH: Chemin pour sauvegarder le modèle formé.

- VECTORIZER_PATH: Chemin pour sauvegarder le vectoriseur TF-IDF.
- OUTPUT_DIR: Dossier pour sauvegarder les résultats (ex. matrice de confusion).

Création du dossier de sortie: os.makedirs(OUTPUT_DIR, exist_ok=True).

Préparation des Ressources NLTK

Téléchargement des stopwords, de la base de lemmatisation et des ressources additionnelles via nltk.download().

Fonctions Principales (Partie 1)

load_model_and_vectorizer()

Description: Charge le modèle et le vectoriseur précédemment sauvegardés.

Fonctionnement:

- Essaie de charger le modèle et le vectoriseur depuis MODEL_PATH et VECTORIZER_PATH.
- Si les fichiers sont introuvables, affiche un message d'erreur et arrête l'exécution.

save_model_and_vectorizer()

Description: Sauvegarde le modèle et le vectoriseur dans des fichiers pour une utilisation ultérieure.

Fonctionnement:

- Utilise joblib.dump() pour sauvegarder le modèle et le vectoriseur.

Fonctions Principales (Partie 2)

1

clean_text(text)

Description: Nettoie et prétraite un texte.

Étapes:

- Convertit en minuscules.
- Supprime les caractères non alphabétiques.
- Retire les mots vides (« stopwords »).
- Applique une lemmatisation sur chaque mot.

2

train_model(data)

Description: Entraîne un modèle de classification Naïve Bayes multinomial.

Étapes:

1. Nettoie les messages avec clean_text().
2. Vectorise les messages avec TF-IDF.
3. Entraîne le modèle avec les labels donnés.
4. Sauvegarde le modèle et le vectoriseur.

Génération de Métriques et Visualisations

generate_metrics(data, model, vectorizer)

Description: Génère des métriques d'évaluation et des visualisations.

Étapes:

- **Rapport de classification:** Précision, rappel, F1-score, etc.
- **Matrice de confusion:** Visualisation des prédictions correctes et erronées.
- Sauvegarde la matrice de confusion sous forme d'image.

Exemples de Graphiques

Exemple de rapport de classification.

Exemple de matrice de confusion.

Interface Utilisateur avec Streamlit

1 Menu Principal

« Prédire un SMS »: Permet à l'utilisateur d'entrer un texte et d'obtenir une classification (Spam ou Ham).

« Réentraîner le Modèle »: Charge un fichier CSV contenant les données (étiquettes « spam » et « ham »). Réentraîne le modèle et affiche les métriques d'évaluation.

2 Exemples d'Interface

Capture d'écran de l'interface pour la prédiction.

Capture d'écran de l'interface pour le réentraînement.

3 Instructions pour Utilisation

Installer les dépendances: pip install -r requirements.txt

Lancer l'application Streamlit: streamlit run SMS_Spam_Classifier.py

Interagir via l'interface utilisateur.

4 Améliorations Possibles

Ajouter d'autres modèles de classification (SVM, Random Forest).

Permettre le téléchargement direct des résultats.

Améliorer le nettoyage du texte pour inclure des expressions régulières avancées.

Ajouter une option pour évaluer le modèle sur un ensemble de test différent.

5 Conclusion

Récapitulatif des points clés.

Prochaines étapes.

Remerciements.

6 Questions/Réponses

Merci pour votre attention.

Questions ?