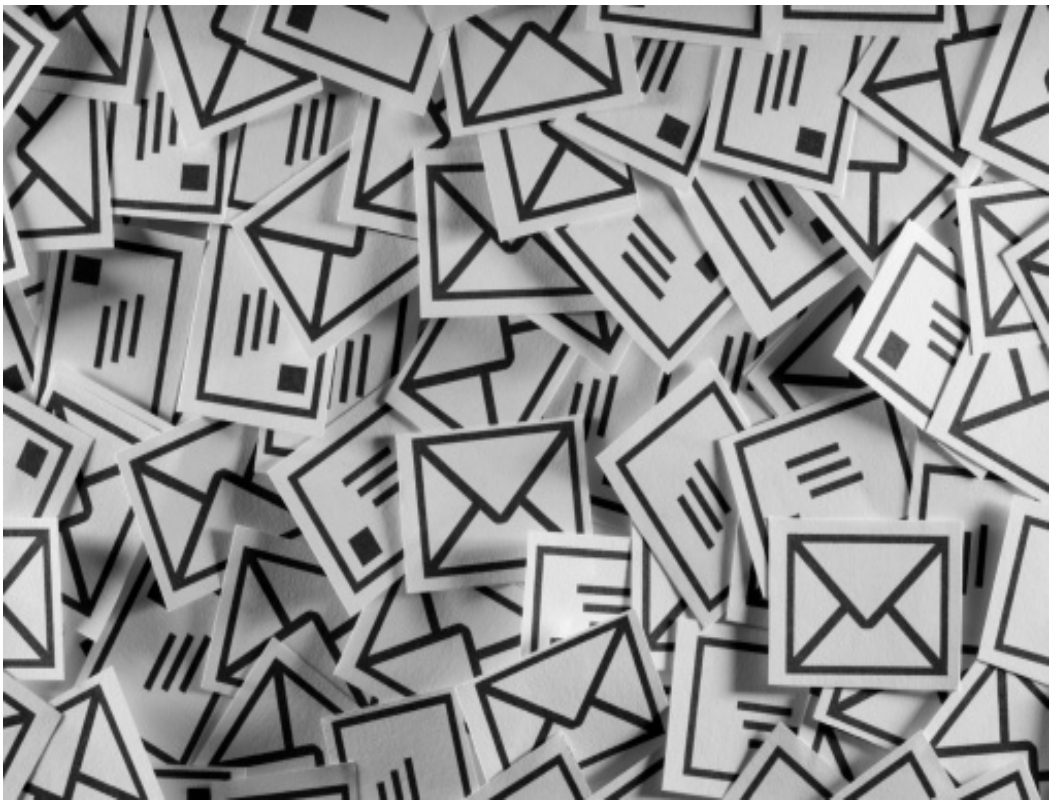
RAPPORT TECHNIQUE DÉTECTIONS DE SPAM

Natural Language Processing



Zohra - Mélody - Imen

21/01/2022

Data Analyst, Data Scientist & Data Engineer

TABLE DES MATIÈRES

INTRODUCTION

Contexte

Nous travaillons dans l'équipe data d'une agence data spécialisée en natural language processing (NLP). Notre agence vient de gagner un appel d'offres avec 6 projets différents.

Pour répondre à cet appel d'offre, le chief data officer a créé 6 équipes avec un chef de projet. Notre équipe est composée d'un data analyst, d'un data scientist et d'un data engineer.

Dans un premier temps, nous vous proposons un modèle de NLP qui répond aux besoins du projet client. Pour cela, nous disposons de différentes sources de données.

Dans un deuxième temps, nous avons réalisé une interface utilisateur qui puisse montrer comment le modèle fonctionne.

Définition du NLP

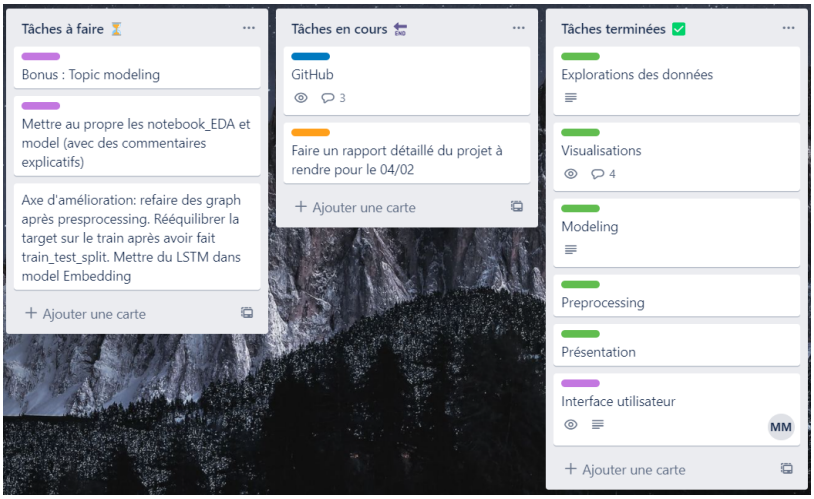
Le **NLP** est une technologie d'intelligence artificielle visant à permettre aux ordinateurs de comprendre le langage humain. L'objectif de cette technologie est de permettre aux machines de lire, de déchiffrer, de comprendre et de donner sens au langage humain.

Il existe une version de notre projet où notre modèle utilise une méthode en Deep Learning, notamment une couche **d'Embedding**. Il s'agit d'un espace dit "peu dimensionnel" dans lequel nous pouvons traduire des vecteurs à haute dimension. Cela facilite l'apprentissage automatique de notre IA sur des entrées de grande taille.

L'avantage de ce modèle est qu'il pourra apprendre et être réutilisé dans plusieurs modèles (donc sur d'autres jeux de données).

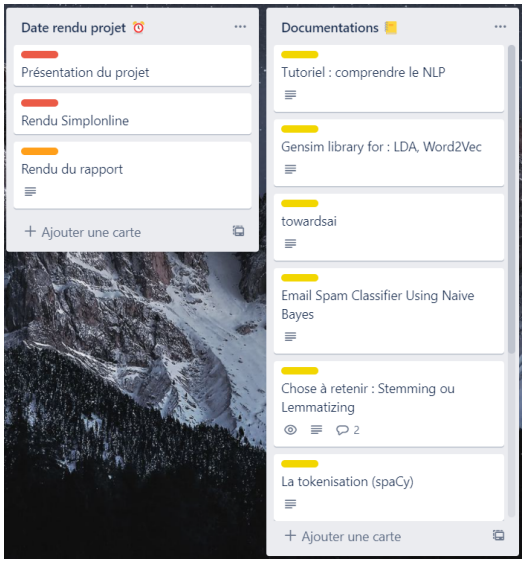
PLANNING

En tant que professionnelles de la data, nous nous sommes réparties les tâches de façon que le projet prenne forme rapidement.



Nous avons défini et planifié les tâches importantes à effectuer, dans les temps impartis (indiqués par des étiquettes).

Nous avons eu à cœur de partager nos découvertes et informations essentielles pour avancer ce projet dans les meilleures conditions en attachant les articles trouvés.



CAS D'USAGE

Explication du problème

Aujourd'hui, les boîtes mail sont envahies de courriers publicitaires ou non-importants qui peuvent faire perdre un temps précieux aux personnes dépendantes d'un bon fonctionnement de leur outil Email.

Dans le jargon courant, les mails inutiles sont nommés "SPAM" qui vient du mot anglais spamming (en français, SPAM veut littéralement dire "pourriel", un courrier bon à être jeté à la poubelle).

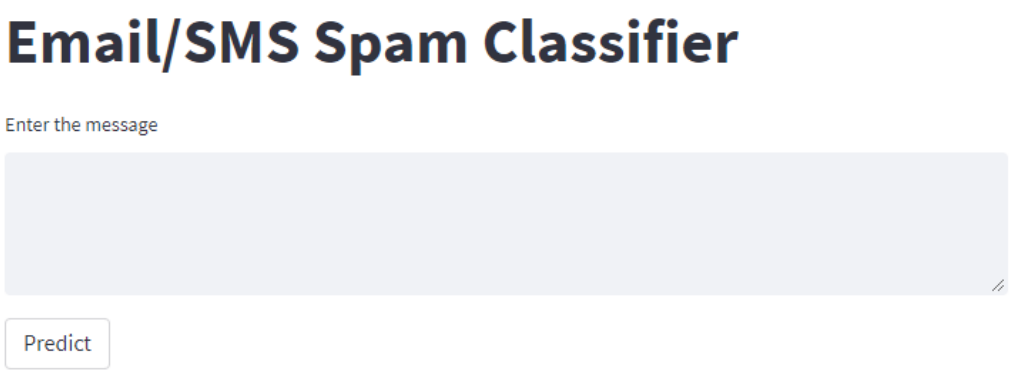
Il est possible de nos jours d'entraîner des modèles informatiques avec une base d'IA pour automatiser le tri des courriers Gmail.

Pour cela, nous pouvons utiliser des modèles en Machine Learning qui vont se baser sur le Natural Language Processing (NLP).

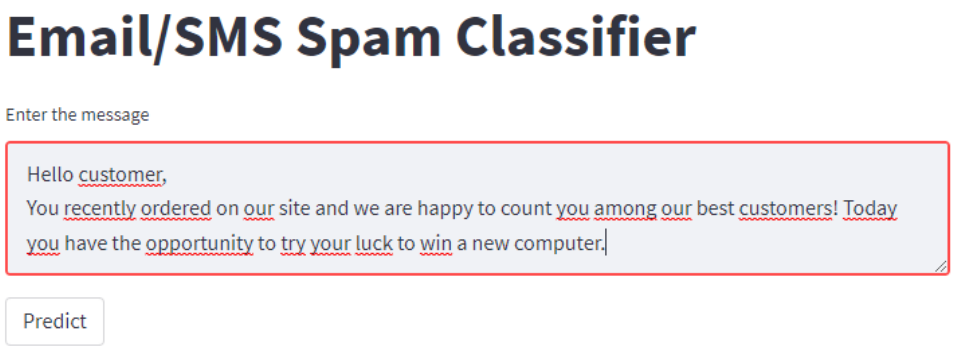
Exemple d'un premier cas d'usage

Voici les premiers résultats de notre modèle entraîné pour vérifier si les mails présentés sont des SPAM ou des No-SPAM.

L'interface utilisateur se présente avec un titre et un espace où l'on peut entrer un texte type d'Email.

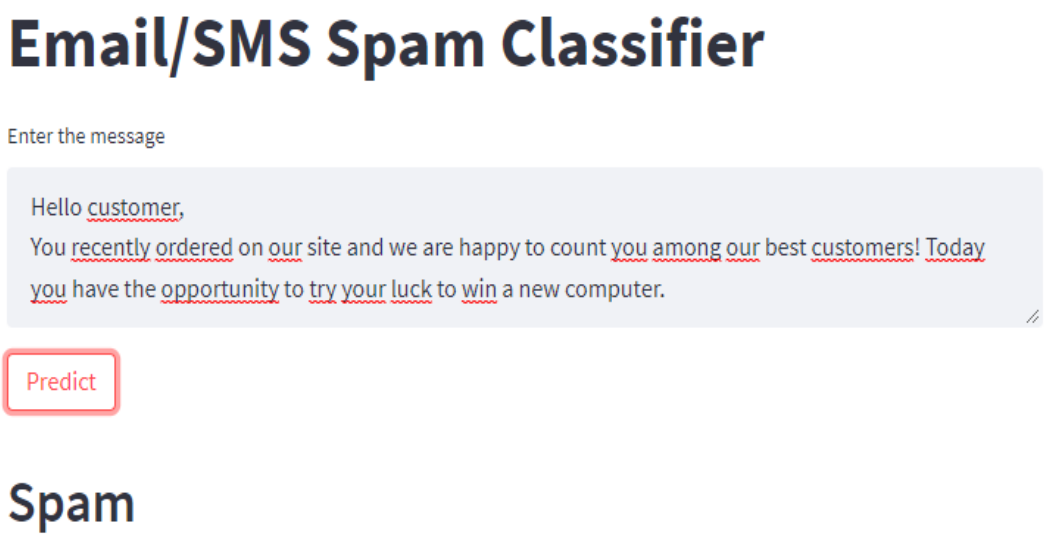


Une fois le texte inséré, l'application va le prendre en compte. **ATTENTION** : Pour le moment, notre modèle est entraîné exclusivement sur l'anglais. Un email dans une autre langue ne pourra pas être lu par notre IA.



En cliquant sur le bouton "Predict", le modèle va prendre en compte l'email et le traiter dans un délai court (possiblement dans un délai plus long selon la taille du texte).

L'application va ensuite indiquer si le mail est un SPAM ou un No-SPAM, comme ceci :



L'email exemple fait penser à ces Email type de publicité. Cependant, en prenant en considération la taille totale du texte (son nombre de mots) et sa rédaction, il est possible que notre modèle mette en SPAM des textes (email) en exemple qui pourraient être catégorisés dans les No-SPAM.

Nous en apprendrons plus sur le fonctionnement de l'IA derrière dans " **SOLUTION** ".

Explications des modèles : Architectures plus en détail

Modèle 1

Model Deep Learning : Embedding

- Etapes :
  - **Séquencer la donnée**, en réduisant la taille max des séquences (nombre de mots maximum par ligne du dataset)
  - **Tokenizer** pour attribuer des nombres à chaque mot.
  - En même temps que la tokenisation, on **normalise la donnée**.
  - Division du dataset en utilisant **train_test_split**. L'entraînement compte 70% de la données pour 30% de test.
  - **Construction du modèle** :
    - Une couche d'**Embedding**, avec 64 dimensions.
    - Une couche de **Flatten**, servant à aplatir nos données
    - Une couche en sortie avec "**sigmoid**".
    - Le modèle est compilé avec les hyperparamètres **Adam, binary_crossentropy et l'accuracy en metrics**.

- Résultats : **Accuracy 99.1**

```
Test Loss: 0.0229
Test Accuracy: 99.19%
Test AUC: 0.9989
```

- Explications : Choix du modèle

Embedding est un espace dit "peu dimensionnel" dans lequel nous pouvons traduire des vecteurs à haute dimension. Cela facilite l'apprentissage automatique de notre IA sur des entrées de grande taille. L'avantage de ce modèle est qu'il pourra apprendre et être réutilisé dans plusieurs modèles (donc sur d'autres jeux de données).

Modèle 2

Model Machine Learning : LightGBM Classifier

- Etapes :
  - Standardisation avec **Standardize_text.**
  - Application du stopwords avec **NTLK.**
  - Lemmatisation du texte avec **WordNetLemmatizer().**
  - Création du corpus de mots.
  - Application de **TfidfVectorizer()** pour revectoriser les mots et pouvoir être lu par le futur modèle.
  - Utilisation de train_test_split pour séparer nos données : une partie pour l'entraînement (80% de la données) et pour le test (20% de la données)
  - Modélisation de notre modèle avec **LightGBM Classifier**

- Résultats : **Accuracy à 99**

	precision	recall	f1-score	support
0	0.99	0.99	0.99	798
1	0.99	0.97	0.98	362
accuracy			0.99	1160
macro avg	0.99	0.98	0.99	1160
weighted avg	0.99	0.99	0.99	1160

- Explications : Choix du modèle

LightGBM, abréviation de Light Gradient Boosting Machine, est un framework d'amplification de gradient distribué gratuit et open source pour l'apprentissage automatique développé à l'origine par Microsoft. Il fonctionne comme un modèle de classification.

Comme nous devions régler un problème de classification, ce modèle nous a paru être un bon choix. De plus, il possède pas mal de fonctions réglant des problèmes, notamment celui du gradient.

CAHIER DES CHARGES

Acteurs

Nom	Qualité / Rôle
Zohra	Projet / Chef de projet, Data analyst
Mélody	Projet / Data scientist, engineer
Imen	Projet / Data engineer
Arturo	Client / Chef rendu projet

Validations

Nom	Date	Validation O/N	Commentaires
Equipe Data	21/01/2022	Oui	Présentation premier jet projet

Historique des modifications

N° de version	Date	Etat	Description de la modification
version 1	14/01/2022	**Modèle 1** : en deep learning (accuracy =0.99)	Validé
version 2	19/01/2022	Essai déploiement	Echec
version 3	20/01/2022	**Modèle 2** : en machine (LightGBMClassifier) avec un accuracy (0.99)	Validé
version 4	21/01/2022	Déploiement avec streamlit (en local)	Validé
version final	22/01/2022	Présentation du rendu final	Validé

Liste de diffusion / Partage de documents

Nom	Partage (Lecture seule / Commentaire / Modification / Propriétaire
Imen	https://github.com/campusx-official/sms-spam-classifier/blob/main/sms-spam-detection.ipynb https://www.kaggle.com/guttiparameswararao/spam-email-classification-using-rnn
Mélody	https://docs.google.com/document/d/1TyZEwzpamprVpWQR1_qnHyl2Zh64MKqlqaCW75nXy-A/edit#heading=h.sl1fen9qa154 https://trello.com/invite/b/OEmgdTfH/52d65e348764c955a093450c3799543f/planning-nlp https://www.kaggle.com/purvitsharma/spam-sentiment-classification-98-accuracy
Zohra	https://www.kaggle.com/gcdatkin/email-spam-classification https://www.canva.com/design/DAE1VByO4a4/share/preview?token=Oe4cUZ2pglJwp7R_RLXOkQ&role=EDITOR&utm_content=DAE1VByO4a4&utm_campaign=designshare&utm_medium=link&utm_source=sharebutton

SOLUTION

Comme expliqué dans l'introduction à ce rapport, l'un de nos modèles possède une base de NLP, notamment une couche d'Embedding.

Le modèle que nous avons présenté avec l'interface utilisateur a été plus simple pour le développement de la futur application de détection de SPAM Email.

Il possède des méthodes de Machine Learning, avec le **TfidfVectorizer** de **Scikit-learn**, qui permet de re-vectoriser nos données. C'est une technique qui consiste à remettre les bonnes dimensions en place pour que les données soient lues et comprises par l'Intelligence Artificielle.

Au préalable, les données ont été nettoyées et passées sous **NLTK** pour qu'il identifie les mots anglais. Nous avons ensuite utilisé la technique **Lemmatizer** avec **WordNetLemmatizer()**.

La lemmatisation est le processus de regroupement des différentes formes fléchies d'un mot afin qu'elles puissent être analysées comme un seul élément. La lemmatisation est similaire à la radicalisation mais elle apporte un contexte aux mots.

C'est ainsi que nous avons pu obtenir notre **corpus de mots** que nous donnons à notre modèle pour qu'il puisse s'entraîner.

Pour notre projet nous avons à notre disposition une base de données comportant des mails, classés par 1 ou 0.

- 0 sont les Emails No-SPAM.
- 1 sont les Emails considérés comme des SPAM.

LIVRABLES

Le projet de groupe s'est fait via GitHub :
https://github.com/Melodyellinn/Projet_NLP/tree/master

Plusieurs branches pour chaque collaborateur peuvent être visitées.

Rendez-vous dans la branche de Mélody pour voir le processus du modèle et dans la branche de Zohra pour retrouver des visualisations plus approfondis sur la base de données.

Les livrables sont :

- Notebook_EDA : L'étude de la forme de la base de donnée, avec visualisation.
- Notebook_model : Preprocessing (cleaning text) et construction du modèle.
- Notebook_predict : Un Step by Step guidé pour utiliser notre modèle.
- app.py : Fichier de l'application (fonctionne en local)
- model.pkl, corpus.pkl, vectorizer.pkl : Le modèle et le nécessaire pour le faire fonctionner (corpus et vectorizer) sous format Pickle.
- Procfile, requirements.txt, setup.sh : Fichier nécessaire pour faire fonctionner l'application avec Streamlit.
- planning.png : Planning de l'équipe
- Projet détection spam.pdf : Présentation du premier jet de l'application.

Le dossier "first_app_dl" contient notre premier modèle, avec la méthode Embedding.

AXES D'AMÉLIORATION

1/ Améliorer le modèle

- Nous pouvons essayer un meilleur traitement des données et permettre à notre modèle d'avoir une compréhension améliorée :
  - Prendre en considération la taille des textes, pour qu'il soit capable de lire n'importe quel Email, qu'il s'agisse des Email officiel ou de la publicité avec peu de mots ou inversement avec énormément de mots.
  - Effectuer un meilleur nettoyage des mots pour améliorer le "corpus de mots", par exemple :
    - Améliorer le stopword (fonction permettant de retirer des mots inutiles)
    - Permettre à notre modèle de lire dans plusieurs langues (Il ne lit actuellement qu'en anglais)
- Rééquilibrer la target pour l'entraînement du modèle.
- Mettre du LSTM (une couche supplémentaire) dans le premier modèle de Deep Learning, qui contient déjà une couche d'Embedding.

2/ Déployer le second modèle (LightGBM Classifier) sur une plateforme en ligne

- Le modèle fonctionne en local avec Streamlit. Il est possible de le déployer en ligne grâce à la plateforme en ligne. Régler le problème des versions du modèle nous permettra de le faire.
- Autre piste : Déployer sur une autre plateforme (exemple : Heroku)

3/ Essayer un déploiement en local et en ligne le premier modèle (Embedding)