l'École Nationale Supérieure d'Informatique et d'Analyse des Systèmes

LIVRABLE PFA 26 Mai 2020
Développement d'un système d'acquisition et de vérification de données dans un système de veille.

Auteur :
Mouad EL KACEM
Asmae GHALAS

Encadrante :
Bouchra El Asri

25 mai 2020

Table des matières

Chapitre 1

L'entrainement du Modele

1.1 Classifieur de base

1.1.1 Random Forest

Le classificateur Random Forest est un ensemble d'arbres de décision où les arbres simples sont construits à partir d'échantillons. À gauche et au centre, deux arbres de la forêt sont représentés en détail : À chaque nœud, la caractéristique qui permet la meilleure séparation de classe est choisie (par rapport au sous-ensemble de caractéristiques sélectionné pour ce nœud). Le partitionnement correspondant de l'espace caractéristique est illustré ci-dessous avec la frontière de décision tracée en violet. À droite, la limite de décision de la forêt aléatoire est affichée. Il est basé sur la majorité des votes des arbres individuels.

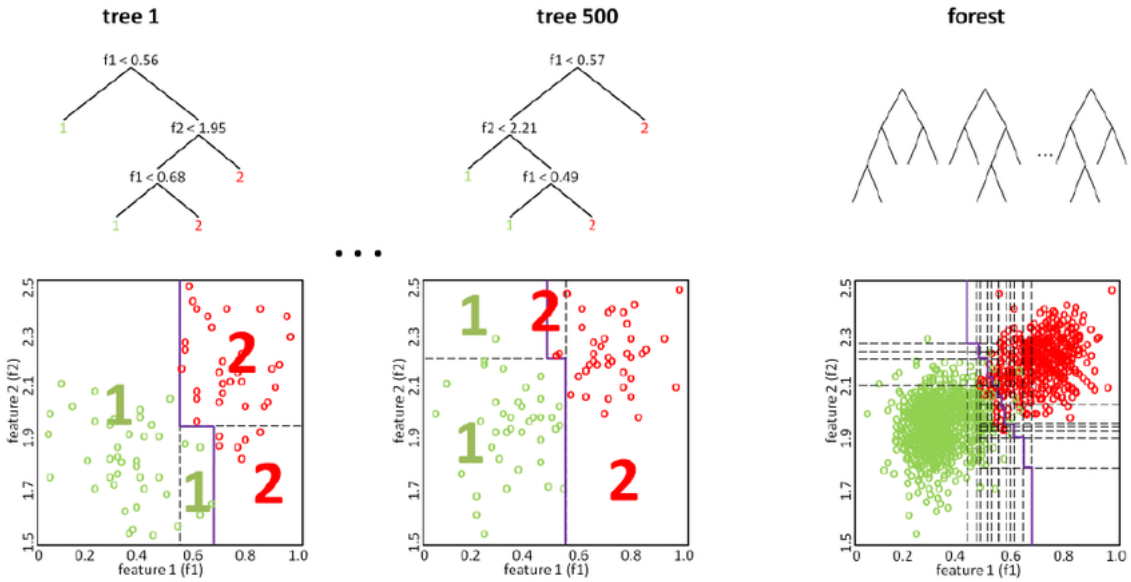


FIGURE 1.1 – Schéma descriptif du classificateur Random Forest

1.1.2 Logistic regression

Logistic regression est un algorithme d'apprentissage automatique qui est utilisé pour les problèmes de classification, c'est un algorithme d'analyse prédictive et basé sur le concept de probabilité.

Nous pouvons appeler une régression logistique un modèle de régression linéaire, mais la régression logistique utilise une fonction de coût plus complexe, cette fonction de coût peut être définie comme la «fonction sigmoïde» ou également appelée «fonction logistique» au lieu d'une fonction linéaire.

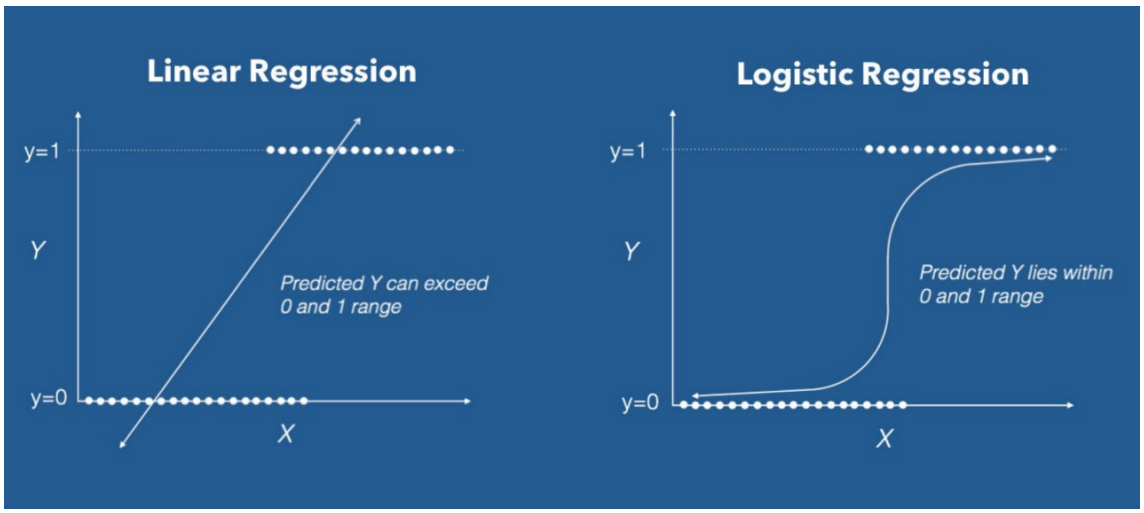


FIGURE 1.2 – Linear Regression VS Logistic Regression

1.2 Préparer les données texte

Le texte doit être analysé pour supprimer des mots, appelé tokenisation. Ensuite, les mots doivent être codés sous forme d'entiers ou de valeurs à virgule flottante pour être utilisés comme entrée d'un algorithme d'apprentissage automatique, appelé extraction de caractéristiques (ou vectorisation).

La bibliothèque scikit-learn propose des outils faciles à utiliser pour effectuer à la fois la tokenisation et l'extraction des fonctionnalités de vos données texte.

1.2.1 CountVectorizer

Le CountVectorizer fournit un moyen simple à la fois de symboliser une collection de documents texte et de créer un vocabulaire de mots connus, mais aussi d'encoder de nouveaux documents en utilisant ce vocabulaire. Un vecteur codé est renvoyé avec une longueur de tout le vocabulaire et un nombre entier pour le nombre de fois où chaque mot est apparu dans le document.

1.2.2 TfidfVectorizer

TF-IDF. Il s'agit d'un acronyme qui signifie «Fréquence des termes - Fréquence du document inverse», qui sont les composantes des scores résultants attribués à chaque mot.

— Fréquence des termes : résume la fréquence à laquelle un mot donné apparaît dans un document.

— Inverse Document Frequency : Cela réduit les mots qui apparaissent beaucoup dans les documents.

Sans entrer dans les mathématiques, TF-IDF sont des scores de fréquence de mots qui tentent de mettre en évidence des mots plus intéressants, par exemple fréquente dans un document mais pas entre les documents.

Le TfidfVectorizer tokenize les documents, apprend le vocabulaire et les pondérations de fréquence des documents inverses, et vous permet d'encoder de nouveaux documents. Alternativement, si vous avez déjà un CountVectorizer appris, vous pouvez l'utiliser avec un TfidfTransformer pour simplement calculer les fréquences inverses du document et commencer à encoder les documents.

1.3 les datasets

Le dataset que nous utilisons pour ce projet est le dataset des 'Fake news' [1]fourni par la competition kaggle Nous avons a notre disposition 44000 enregistrements ou chaque enregistrement represente un article, nous avons divise cet ensemble d'apprentissage en en deux parties : train et teste.

Aussi qu'une deuxieme Dataset pour la categorisation qui Se compose de 2225 documents du site Web de nouvelles de la BBC [2]correspondant à des histoires dans cinq domaines d'actualité

Labels de classe : 5 (affaires, divertissement, politique, sport, technologie)

1. https ://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset
2. http ://mlg.ucd.ie/datasets/bbc.html

1.4 Classification Simple

Le premier Model se base sur le CountVectorizer et Logistic Regression qu'ils sont appliquer sur le Dataset du Fake news , qui se compose de :

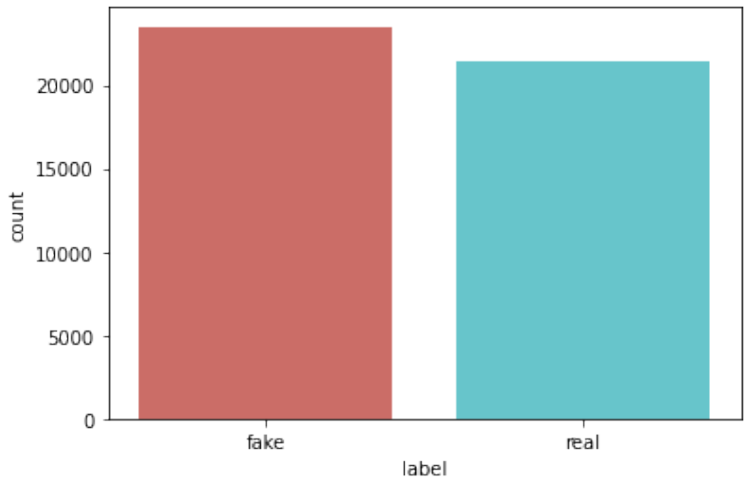


FIGURE 1.3 – Proportion du fake/real articles sur le dataset

Le resultat de l'entrainement etait comme de suit :
accuracy = 0.89
La matrice de confusion :

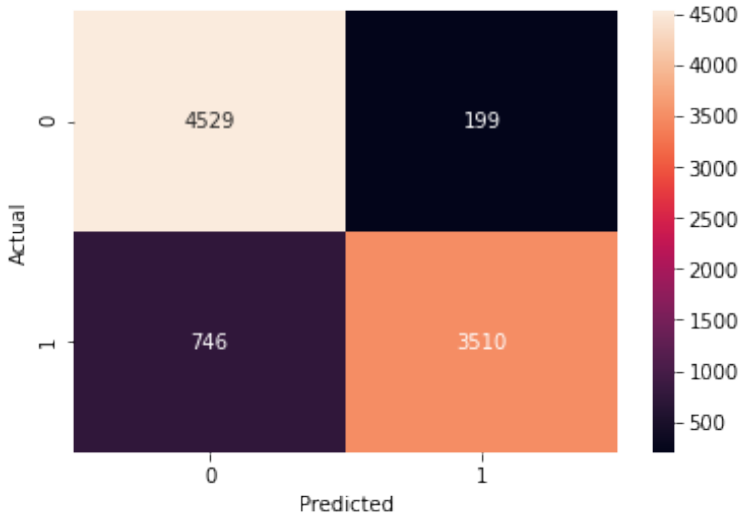


FIGURE 1.4 – Matrice de confusion du Classification simple

1.5 Classification optimisée

Notre classification optimisé consiste a divisé les articles en catégories (affaires, divertissement, politique, sport, technologie) et puis l'entraînement des modèles selon ces catégories

1.5.1 Modèle de catégorisation

Ce modèle se base sur le TfidfVectorizer et Random Forest qu'ils sont appliquer sur le Dataset du BBC , qui se compose de :

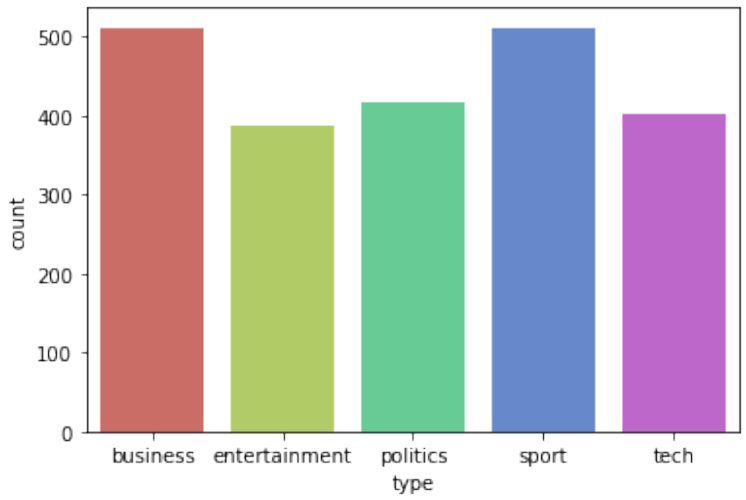


FIGURE 1.5 – Distrubution des categories sur le Dataset

Le resultat de l'entrainement etait comme de suit :
accuracy = 0.96
La matrice de confusion :

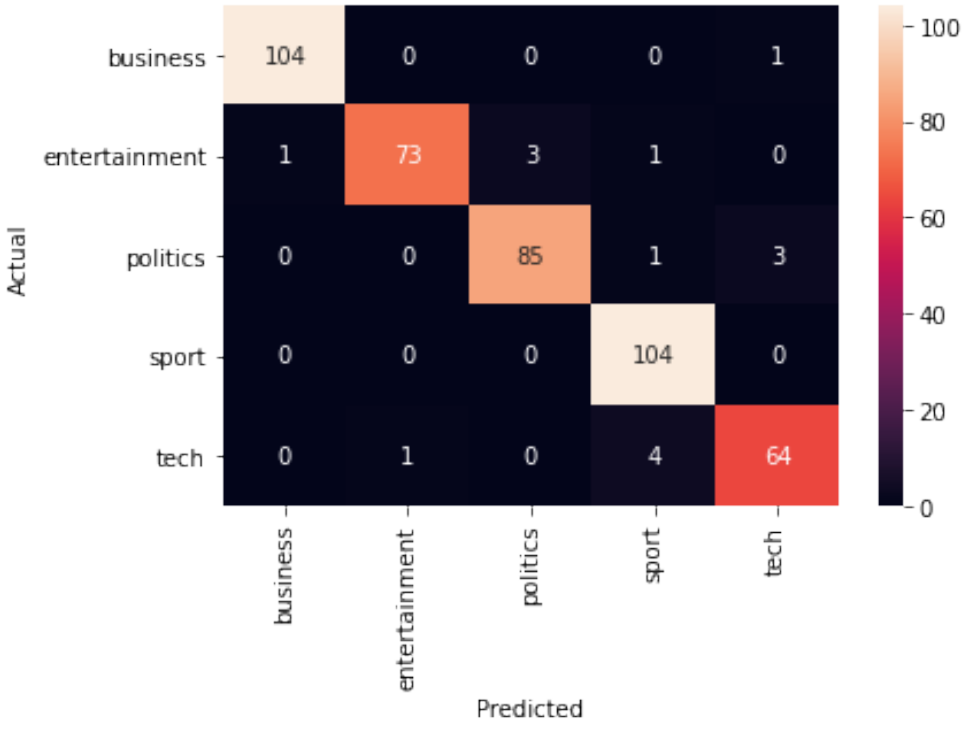


FIGURE 1.6 – Matrice de confusion du Modèle de categorisation

Le résultat et satisfaisant , nous pouvons prendre des décisions à partir de ce modèle, ce modèle sera utiliser par suit sur la division des données sous des catégories

1.5.2 Les sous-modèle

Comme premier pas, nous allons appliquer le modèle précèdent sur le dataset du 'Fake News', la distribution du dataset alors devenu comme de suit :

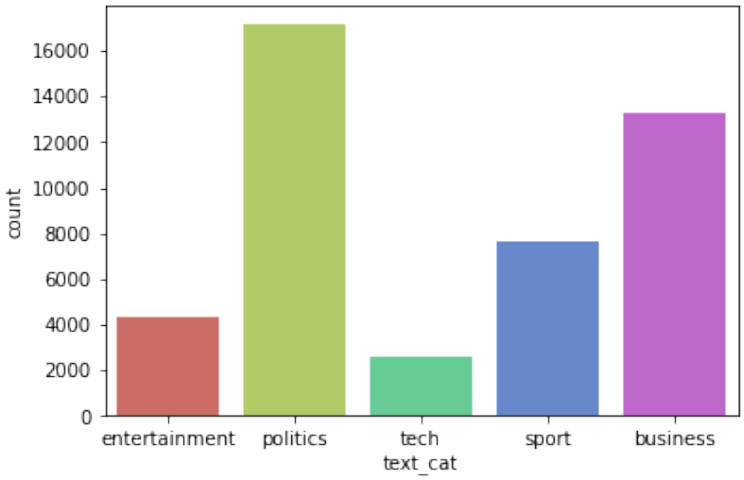
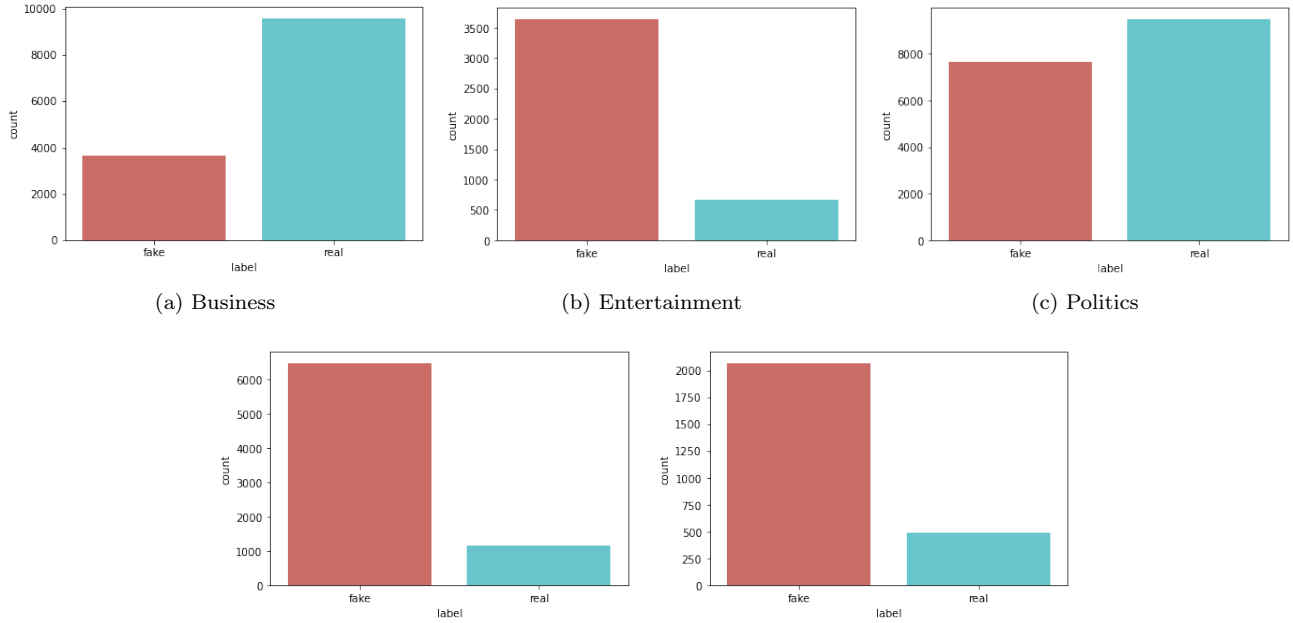


FIGURE 1.7 – Distrubution des categories sur le Dataset du 'fake news'

l'entraînement des Modèles sera basée sur le CountVectorizer et Logistic Regression qu'ils sont appliquer sur chaque catégorie

1.5.2.1 Les distribustions



(a) Business (b) Entertainment (c) Politics

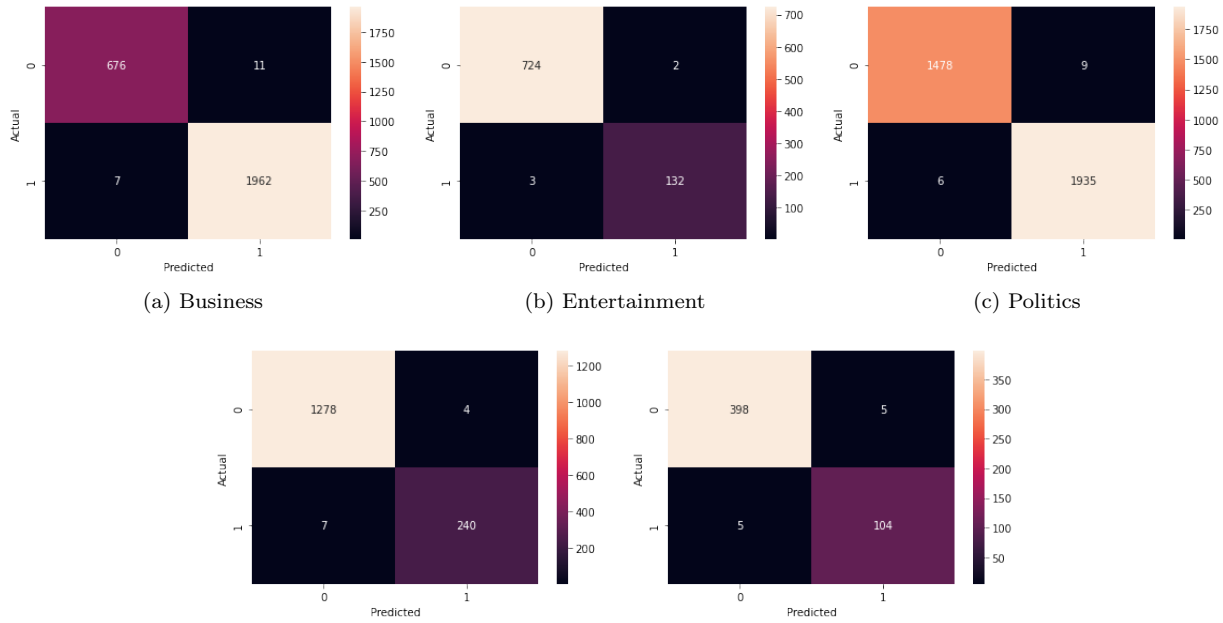
(d) Sport (e) technique

FIGURE 1.8 – Différents diffusions selon les catégories

1.5.2.2 Résultats d'entraînements

L'accuracy de chaque categorie etait comme de suit :
Entertainment : 0.98
technique : 0.97
Business : 0.99

Politics : 0.99
Sport : 0.98



(a) Business (b) Entertainment (c) Politics

(d) Sport (e) technique

FIGURE 1.9 – Matrices de confusion selon les catégories

1.5.2.3 Résultat d'assemblage

Apres l'assemblage du modéle de categorisation avec les sous modéle, l'accuracy finale est devenu : 0.99

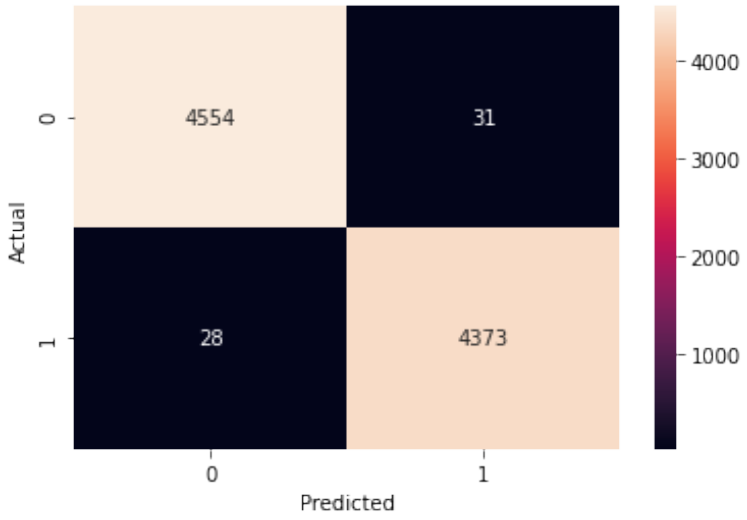


FIGURE 1.10 – Distrubution des categories sur le Dataset

Chapitre 2

Interface utilisateur graphique

2.1 Outil

Flask : c'est un micro-framework Open source, il se différencie par sa légèreté permettant de disposer d'une solide base de développement tout en conservant la flexibilité et la lisibilité du langage de programmation Python. Facile à prendre en main, il optimise le processus de développement. Il accompagne ainsi les entreprises dans leurs besoins en applications web de petite et moyenne envergure.

FIGURE 2.1 – Logo du Flask

Flask a été conçu pour être un micro-framework simple et léger. Son but premier est de permettre de démarrer le développement d'une application web sur une base solide sur laquelle appuyer le reste des phases de développement. À partir de cette base, il est possible de construire son projet bloc à bloc, selon les besoins.

2.2 Résultat actuel

le Framework python léger Flask utilisé pour faire tourner le serveur web de notre application.
L'image ci-dessous représente la page d'accueil (route '/') de l'application Flask elle permet d'ajouter un texte qui l'utilisateur souhaite vérifier. Une fois toutes les informations saisies, l'utilisateur appui sur la touche « predict » .

FIGURE 2.2 – la page d'accueil

L'image ci-après représente la page (route '/predict') permettant à l'utilisateur de savoir le type(business, sport, tech, entertainment, politics) et la véracité(fake/Real) de son texte :

FIGURE 2.3 – la page des prédictions