ECE
ÉCOLE D'INGÉNIEURS
ENGINEERING SCHOOL

Cahier des Charges Fonctionnelles et Techniques

Projet d'intégration des modèles IA

Détection des Fake News avec l'IA

1. Contexte et Objectifs

Contexte : Dans un contexte de multiplication des contenus médiatiques en ligne, la désinformation se répand rapidement, particulièrement via les réseaux sociaux. Notre projet, réalisé par **Gémima ONDELE**, **Hubert CHAVASSE**, **Mohamed GHARMAOUI**, **Niangoran BOKA** et **Hadrien LENNON**, vise à créer un système automatisé de détection de fake news basé sur l'IA.

Objectif principal : Développer une application complète capable :

- d'identifier et de signaler les articles potentiellement faux,
- de fournir un score de confiance pour chaque prédiction.
- D'améliorer les modèles par les nouvelles données scrappées

Sous-objectifs :

1. Définir les critères de classification des fake news (phase Prompter).
2. Entraîner un modèle NLP performant (phase Modéliser).
3. Nettoyer et vectoriser les données (phase Processer).
4. Déployer le modèle via une API et une interface utilisateur (phase Robotiser).
5. Tester la robustesse sur diverses sources (phase Généraliser).
6. Intégrer un système de feedback pour amélioration continue (phase Data-driven).

2. Portée et Périmètre

- **Fonctionnel** :
    - Traitement de texte libre ou d'URLs.
    - Choix entre plusieurs modèles (TF-IDF+RF, XGBoost, Gradient Boosting, BERT).
    - Retour d'une étiquette (Fake/Real) et score de confiance.

- Collecte de feedback utilisateur.

- **Non-fonctionnel** :
  - Performance : réponse API < 200 ms pour 90 % des requêtes.
  - Sécurité : protection des endpoints, validation des entrées.
  - Scalabilité : montée en charge sur plusieurs milliers de requêtes/jour.

3. Exigences Fonctionnelles

ID	Exigence
F1	L'utilisateur saisit un texte ou une URL d'article.
F2	Le système nettoie et prétraite automatiquement le texte soumis.
F3	Le système propose le choix du modèle de classification.
F4	L'API renvoie une prédiction binaire (Fake/Real) et un score de confiance.
F5	L'utilisateur peut indiquer via l'interface si la prédiction était correcte.
F6	Le feedback utilisateur est enregistré pour réentraînement futur.

4. Exigences Techniques

1. **Langages & Frameworks** : Python, FastAPI, Streamlit.
2. **Modèles ML** : scikit-learn, XGBoost, Hugging Face Transformers(bert-tiny-finetuned-fake-news), random forest, gradient boosting.
3. **Modèles Deep learning : Réseau de neurones** (BERT_Classifier)
4. **Stockage** : fichiers Joblib pour modèles/vectorizer, CSV pour feedback, dataset et données scrappées.
5. **Infrastructure** : déploiement local sur le web.

5. Livrables

- Dataset nettoyé (all_news_cleaned.csv).
- Modèles entraînés : fake_news_classifier, xgboost_fake_news, gradient_boosting_fake_news, bert.
- API FastAPI avec endpoints / et /predict.

- Interface utilisateur Streamlit : détection et dashboard feedback.
- Rapport technique détaillé.

6. Planning et Jalons

Jalons	**Date cible**
Cadrage & CDC	Semaine 1
Prétraitement des données	Semaine 2
Entraînement modèles	Semaine 3 à 4
Développement API	Semaine 5
Interface Streamlit	Semaine 6
Tests & Généralisation	Semaine 7
Feedback & Itérations	Semaine 8
Présentation finale	Semaine 9

7. Rôles et Responsabilités

Rôle	**Étudiant(s)**
Chef de projet	Hubert CHAVASSE
Data Engineer (Processer)	Esther ONDELE
Data Scientist	Gémima ONDELE
Développeur API	Mohamed GHARMAOUI
Développeur UI/Tests	Niangoran BOKA, Hadrien LENNON
