BIHAR2025, parcours BIHAR-CORSE
Projet final de mise en situation professionnelle reconstituée

1 Objectifs et compétences visées

Cette mise en situation professionnelle reconstituée a pour objectif la validation des compétences pratiques des modules suivants :

- **Machine Learning II**
  - Concevoir, implémenter et documenter un pipeline de prédiction pour une série temporelle.
  - Expliquer et interpréter les résultats obtenus à l'aide d'un modèle ML.
- **Deep Learning I**
  - Concevoir, implémenter et documenter un pipeline de classification d'images à l'aide de réseaux de neurones profonds.
- **Deep Learning II**
  - Concevoir, implémenter et documenter un pipeline de classification de textes en utilisant des méthodes traditionnelles de NLP ainsi que des réseaux de neurones profonds, notamment les LSTM.
- **MLOps**
  - Développer un code Python structuré en séparant les différentes phases d'un pipeline ML, telles que le prétraitement des données, l'entraînement et la génération des prédictions à l'aide d'un modèle d'apprentissage automatique.
  - Gérer les expérimentations et le versionnement des modèles d'IA.
  - Implémenter une API exposant un modèle ML, incluant le prétraitement des données, en utilisant un framework tel que FastAPI.
- **DevOps**
  - Créer une image docker pour empaqueter le code Python de l'API incluant des fichiers nécessaires (tels que modèles ML sérialisés par exemple).
  - Ajouter un logging pertinent pour chaque handler de endpoint.
  - Ajouter un endpoint pour connaitre la version logicielle actuelle.
  - Concevoir, implémenter et exécuter des tests automatisés des endpoints d'une API.
  - Définir un pipeline basé sur GitHub Actions qui crée une image docker, l'envoie vers ghcr.io et exécute des tests de l'API.

Chaque étudiant travaillera **individuellement** sur trois sous-projets. Chacun de ces sous-projets consistera à implémenter et documenter des expérimentations visant à produire un modèle de prédiction adapté à un type de données spécifique. La mise en œuvre du pipeline MLOps et DevOps sera exigée pour un seul de ces modèles.

L'évaluation prendra la forme d'un oral durant lequel le candidat présentera et défendra les réflexions qu'il aura menées et les solutions qu'il aura apportées. Les livrables serviront de base au jury pour questionner l'apprenant sur des points particuliers.

2 Contexte, données et méthodologie

Vous travaillez en tant qu'ingénieur IA pour une entreprise spécialisée dans l'analyse et l'exploitation des données multimodales (images, textes et séries temporelles) afin d'optimiser la prise de décision et d'améliorer les performances de ses services.

Votre mission est de concevoir, implémenter et documenter des pipelines de modèles d'apprentissage automatique adaptés aux différents types de données traités par l'entreprise.

Cette mission implique la réalisation des tâches suivantes :

- Développer un pipeline de classification d'images à l'aide de réseaux de neurones profonds.
- Mettre en place un pipeline de classification de textes en combinant des méthodes traditionnelles de NLP et des modèles de réseaux de neurones, notamment LSTM.
- Développer un pipeline de prédiction basé sur des séries temporelles.
- Assurer la gestion des expérimentations et du versionnement des modèles.
- Implémenter une API pour exposer un modèle ML, incluant le prétraitement des données, en utilisant un framework tel que FastAPI.
- Déployer un pipeline MLOps avec CI/CD pour automatiser le cycle de vie des modèles développés.

Dans le cadre de cette mise en situation professionnelle reconstituée, il vous est demandé de réaliser des sous-projets dont les objectifs, les données et les méthodologies sont présentés dans les chapitres qui suivent.

2.1 Classification des images

L'objectif est de produire un modèle de classification d'images multi-classes, résultant des expérimentations basées sur différentes architectures de réseaux neuronaux convolutifs (CNN).

Modules concernés

- Deep Learning I,
- Machine Learning II (explicabilité des modèles).

Données

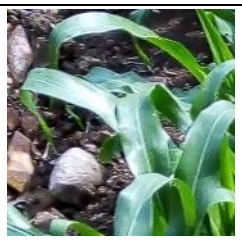
Vous aurez à votre disposition des photos prises dans un champ de maïs disponibles sur Kaggle. Elles ont été prises dans un champ de maïs à l'aide d'un smartphone positionné à une hauteur de 1 à 1,5 mètre du sol, afin de simuler l'utilisation d'une machine de pulvérisation équipée d'une caméra, en perspective. L'objectif est d'identifier les zones nécessitant un arrosage.

Variable cible

Le modèle doit prédire la classe d'une image parmi les suivantes :

- *ground* (« chao ») : sol sec sans végétation,
- *corn* (« milho ») : végétation de maïs,
- *weeds* (« hervas ») : plantes herbacées diverses,
- *corn/weeds* (« milho/ervas ») : présence mixte de maïs et de mauvaises herbes.

Exemples de données

ground	corn	weeds	corn/weeds
			
			

METHODOLOGIE

- Commencez par entraîner le modèle sur les 3 premières classes. Une fois les premiers résultats obtenus, poursuivez les expérimentations avec les 4 classes.
- Vous pouvez réduire la taille du jeu de données en ne sélectionnant qu'une partie des images, ainsi que redimensionner celles-ci pour diminuer leur taille et accélérer l'entraînement.
- Faites vos propres choix des techniques de prétraitement des données.
- Expérimentez différents optimiseurs tels que Adam, RMSprop, Adagrad, ou d'autres.
- Testez l'impact de dropout en combinaison avec la batch normalization, en faisant varier les taux de dropout.
- Utilisez des modèles préentraînés de votre choix (comme VGG16, VGG19, Xception, etc.).
- Evaluez les performances des modèles à l'aide des courbes d'accuracy et de loss, ainsi que de la matrice de confusion.
- Pour enrichir l'analyse, sélectionnez quelques images issues de différentes classes pour les tests supplémentaires :
  - Affichez chaque image avec la probabilité associée à la classe prédite par le modèle.
  - Interprétez les prédictions en visualisant les superpixels explicatifs à l'aide de la bibliothèque LIME (Local Interpretable Model-agnostic Explanations)

2.2 Classification des textes

L'objectif est de produire un modèle de classification de critiques de films selon leur tonalité, positive ou négative, résultant des expérimentations utilisant à la fois des méthodes traditionnelles de NLP et des approches basées sur l'apprentissage profond.

Modules concernés

- Deep Learning II.

Données

Vous aurez à votre disposition un ensemble des critiques de films en langue française publiées sur le site allocine.fr dans la période du 2006 à 2020. Le dataset est disponible via la bibliothèque huggingface. Il contient 100K critiques positives et 100K critiques négatives, réparties en ensembles

d'entraînement (160K), de validation (20K) et de test (20K). Le jeu de données a été produit par Théophile Blard : *(Théophile Blard, French sentiment analysis with BERT, (2020), GitHub repository)*.

Variable cible

Le modèle doit prédire la classe d'une critique parmi :

- 0: critique négative,
- 1: critique positive.

Exemples de données

review	label
Superbe! il n'y a pas d'autre mot pour qualifier "Le choix des armes" du très regrettè Alain Corneau, sorti en 1981, tirè du roman de Michel Grisolia! On regarde, admiratif, un polar français bien sombre, un film noir très noir comme on n'en fait plus aujourd'hui! C'est aussi une oeuvre majeure des 80's à la rèalisation carrèe aux couleurs automnales et aux dècors et paysages tristes de H.L.M. dèlabrès en parfaite adèquation! La distribution est tout bonnement impeccable et balaye tout sur son passage jusqu'aux seconds rôles : Yves Montand en truand repenti obligè de reprendre les armes, Gèrard Depardieu en jeune chien fou paranoïaque et incontrôlable, Catherine Deneuve en èpouse de belle classe, Gèrard Lanvin en inspecteur frais èmoulu, Michel Galabru, les jeunes Richard Anconina et Jean-Claude Dauphin, etc. [...] Bref, on retrouve ici les thèmes chers du film noir, tels les codes d'honneur du milieu ou la fureur, ainsi que l'amour pour une femme...ou pour les chevaux! il y a aussi la musique inoubliable de Philippe Sarde qui ajoute un gros plus à l'ensemble! Et puis rarement on aura vu au cinèma un mari (Montand) aimer autant sa femme (Deneuve). Admirateur de polar made in France ne pas s'abstenir : "Le choix des armes" est ni plus ni moins un chef d'oeuvre à l'atmosphère envoûtante...	1
Probablement l'un de mes films classiques préférés. L'interprétation est grandiose et les décors somptueux. Malgré les âges, ce film n'a pas pris une ride.	1
Ça dégouline de clichés et d'effets visuels foireux, ce "film" est une honte tant il est mauvais et cela malgré un casting pourtant attirant. Non, rien n'est à sauver dans cet immense néant!	0

METHODOLOGIE

- Définissez un pipeline de prétraitement en vous basant sur les techniques vues dans les TP du module (nettoyage, tokenisation, suppression des stopwords, etc.) et adaptez-le si nécessaire.
- Implémentez un modèle baseline à l'aide des méthodes simples comme bag-of-words et TF-IDF, en testant au moins deux algorithmes de classification de votre choix.
- Encodez les textes en utilisant un modèle word2vec préentraîné pour la langue française. Vous pouvez utiliser un des modèles développés par Jean-Philippe Fauconnier.
- Utilisez LSTM pour la classification. Comme pour la classification des images, expérimentez les différentes architectures et optimiseurs et documentez les résultats obtenus.
- Evaluez la performance des différents modèles en se basant sur les matrices de confusion et les métriques adaptées.
- Étudiez quelques erreurs de classification produites par vos modèles. Tentez d'identifier des motifs ou hypothèses explicatives.

2.3 Prévision des séries temporelles

L'objectif est de produire un modèle de prévision des séries temporelles, en expérimentant à la fois des méthodes statistiques et des modèles de régression basés sur l'apprentissage automatique. Un pipeline MLOps intégrant une approche CI/CD sera mis en place pour le déploiement et le suivi du modèle.

Modules concernés

- Machine Learning II
- MLOps
- DevOps

Vous utiliserez des données météo historiques fournies par Historical Weather API (Open-Meteo). Cette API repose sur des ensembles de données de réanalyse des observations provenant de différentes sources (stations météorologiques, radars, satellites, etc.) et, grâce à des modèles mathématiques, fournit des informations météorologiques historiques détaillées, y compris pour des zones sans station météorologique à proximité.

Vous devez récupérer une **série temporelle des températures à 2m du sol** (« Temperature (2 m) ») pour une ville de votre choix, en utilisant ses coordonnées géographiques (latitude, longitude).

Objectif

L'objectif est de développer un modèle de prédiction capable d'estimer la température de la ville sélectionnée pour une journée donnée, avec un pas de temps de 3h (00h, 03h, 06h, ..., 21h).

Exemple de données

time	temperature_2m (°C)
2022-01-01T00:00	27.2
2022-01-01T01:00	26.8
2022-01-01T02:00	27.0
2022-01-01T03:00	27.0
2022-01-01T04:00	27.1
2022-01-01T05:00	27.2
2022-01-01T06:00	27.2
2022-01-01T07:00	27.2
2022-01-01T08:00	27.2
2022-01-01T09:00	27.1
2022-01-01T10:00	27.1
2022-01-01T11:00	27.2
2022-01-01T12:00	27.1

METHODOLOGIE

- **Acquisition des données**
  - Définir la période d'étude en fonction des données disponibles via l'API et les résultats de l'analyse exploratoire (ex. : dernière année, période plus longue).
  - Vérifier s'il y a des valeurs manquantes. En cas de données manquantes, appliquer une interpolation linéaire.
- **Transformation de la série temporelle**
  - Agréger les données pour obtenir une nouvelle série avec un pas de temps de 3 heures : la valeur à 00h sera la moyenne des valeurs mesurées à 00h, 01h et 02h ; la valeur à 03h sera la moyenne des valeurs mesurées à 03h, 04h et 05h ; et ainsi de suite.
  - Utiliser cette série transformée dans les expérimentations.
- **Analyse exploratoire**
  - Étudier les tendances, saisonnalités et résidus de la série temporelle.
  - Visualiser la série et identifier d'éventuelles anomalies ou patterns récurrents.
- **Expérimentation avec des méthodes statistiques**
  - Implémentez et documentez les expérimentations avec ARIMA, SARIMA et SARIMAX.
  - Pour SARIMAX, explorer des variables exogènes issues de la même API.
  - Effectuer un tuning des hyperparamètres.
- **Expérimentation avec des méthodes de régression basées sur le Machine Learning**
  - Tester différentes configurations de variables explicatives : variables retardées (lag), variables construites à partir de valeurs agrégées, variables exogènes (possibilité d'utiliser la même que pour SARIMAX).

  - Comparer plusieurs modèles de régression.
- **Analyse des résidus et évaluation des performances**
  - Étudier la distribution des erreurs résiduelles.
  - Comparer les performances des modèles sur des métriques adaptées.
  - Documenter et interpréter les résultats obtenus.
- **Déploiement du modèle**
  - Implémenter le pipeline d'entraînement, résultant de la phase d'expérimentation, dans un script Python.
  - Créer une base de données SQLite (prévoir la possibilité d'évoluer vers PostgreSQL ou autre) :
    - Les données météo historiques,
    - Les métadonnées des modèles : période utilisée pour l'entraînement, version, paramètres d'accès,
    - Les prédictions générées : identifiant du modèle utilisé, valeurs des variables explicatives, prédictions.
  - Mettre en place une approche batch pour la génération des prédictions. Les prédictions sont précalculées périodiquement, stockées dans une base de données et accessibles via une API. **Dans le cadre de ce projet, il n'est pas nécessaire d'automatiser l'exécution de ce workflow de réentraînement de modèle.**
    - Télécharger les données récentes, par rapport à une date donnée, via l'API et sauvegarder les dans la base de données,
    - Entraîner un nouveau modèle en utilisant votre pipeline d'entraînement,
    - Générer des prédictions pour une période prédéfinie,
    - Stocker les prédictions dans la base de données.
  - Réaliser une API REST avec FastAPI pour exposer les prédictions stockées.
  - Mettre en place des tests unitaires et d'intégration) et un logging pertinent
  - Réaliser un pipeline CI/CD basé sur GitHub Actions

3 Livrables

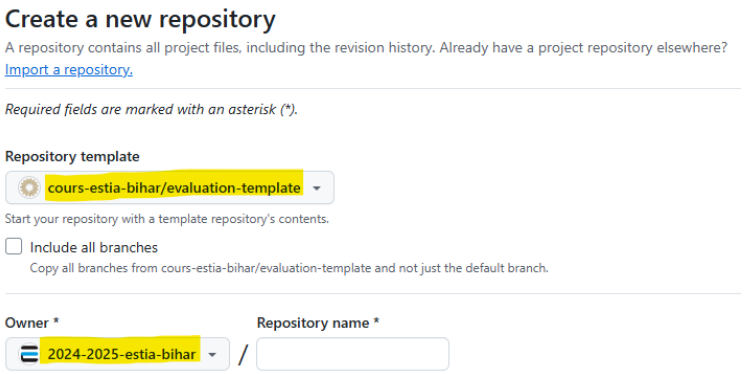
Vous devez créer un dépôt github unique pour la totalité des livrables du TPT.

La documentation doit être faite en anglais ou en français.

Bonnes pratiques :
- Créer un repository github dans l'organisation **2024-2025-estia-bihar** dès le début du projet et poussez le code graduellement,
- Utiliser le template du projet (cf. chapitre 3.1),
- Respecter des conventions de nommage des dossiers et des fichiers.

3.1 Structure du dépôt github

Créer un repository github dans l'organisation **2024-2025-estia-bihar** en utilisant le template suivant : https://github.com/cours-estia-bihar/evaluation-template.



Voici les détails de la structure :

- **notebooks/**
  - Un Jupyter notebook pour la classification d'images
  - Un Jupyter notebook pour la classification de textes
  - Un Jupyter notebook pour la prédiction de série temporelle
- **data/**
  - Base de données des données d'entraînement des modèles, des métadonnées des modèles entraînés et des prédictions générées.
  - Script Python pour l'acquisition des données récentes, en fonction d'une date donnée
- **model/**
  - Script Python pour la génération des prédictions à une date donnée.
  - **registry/** : registre des modèles entraînés.
- **monitoring/**
  - Script Python pour générer un graphique comparant les prédictions aux données réelles observées
  - output/ (dossier contenant les images générées)
- **api/**
  - main.py : ce script doit contenir l'implémentation des endpoints pour les fonctionnalités suivantes :
    - Récupération des prédictions pour une date donnée
    - Récupération des prédictions combinées avec des données réelles observées pour une période donnée
    - Récupération de la version logicielle actuelle (type string)
      - Retourne 0.0.0 en local
      - Retourne le commit ID quand compilé depuis le CI/CD
- **README.md**
  - Description du projet, de l'architecture et des flux de données
  - Instructions pour l'installation des dépendances, run, build et test
  - Description du pipeline CI/CD

3.2 Structure des notebooks Jupyter

Utilisez les cellules markdown pour structurer vos notebooks de façon suivante :

- Description synthétique du projet et de ses objectifs
- Chargement des données

- Analyse exploratoire des données
- Séparation des données en ensembles d'entraînement, de validation et de test
- Prétraitement des données
- Modélisation et évaluation des performances des modèles
- Analyse et interprétation des résultats obtenus

Les notebooks doivent inclure les résultats des cellules exécutées. Aucun notebook ne sera réexécuté par les examinateurs.
Cependant, **veillez à ne pas inclure les sorties trop volumineuses** (images trop lourdes, tensors, logs longs…), car cela alourdit inutilement le repository. Le notebook doit uniquement contenir les informations essentielles. Les impressions (prints) que vous utilisez pour votre propre compréhension ne doivent pas être incluses dans le fichier final.

3.3 Support de présentation

Pour chaque sous-projet ML/DL, l'apprenant devra présenter :

- Les spécifications, les objectifs et les métriques d'évaluation de performance choisies,
- Les résultats synthétiques de l'analyse exploratoire des données,
- La description synthétique du prétraitement réalisé et la justification des choix faits,
- La description synthétique des expérimentations menées et leur justification : algorithmes et leurs paramètres, architectures,
- Les résultats obtenus, leur comparaison et interprétation.

Pour la partie MLOps/DevOps, l'apprenant devra présenter :

- L'architecture de l'application et le flux des données
- Le pipeline de CI/CD avec la description des outputs de ses différentes étapes

Le travail fourni doit être soigneusement relu et maitrisé. L'accent sera donné à la capacité à expliquer les différents aspects techniques et points durs rencontrés.

La **démonstration** concerne uniquement l'API exposant le modèle ML requis. L'apprenant devra préparer un **scénario** qui permet de montrer le fonctionnement des différents endpoints de l'API dans le créneau imparti.

4 Modalités d'évaluation

Pour l'ensemble des modèles produits, la méthodologie suivie et la justification des différents choix effectués seront prioritaires dans l'évaluation, par rapport à la qualité des prédictions obtenues.

MACHINE LEARNING II, **DEEP LEARNING I, DEEP LEARNING II** :

- Les notebooks Jupyter respectent la structure demandée (voir le chapitre correspondant).
- La méthodologie est respectée (voir le chapitre correspondant à chaque sous-projet).
- Les blocs de codes sont commentés. Les commentaires peuvent être succincts mais ils doivent permettre de comprendre la logique et l'organisation de votre code.
- Les graphiques doivent avoir des titres, des noms corrects des axes, des légendes et des commentaires textuels.

- Les choix doivent être justifiés.
- Transformations des données requises par les algorithmes utilisés, sont faites.
- Il ne doit pas y avoir de fuite des données.
- Les principaux résultats doivent être résumés et expliqués.
- Les résultats doivent être reproductibles.

DEVOPS

- *Compétence : Créer une image docker pour empaqueter le code Python de l'API incluant des fichiers nécessaires (tels que modèles ML sérialisés par exemple). Références des séquences de cours : devops_3_docker, devops_5_deploy.*

  Critères d'évaluation :
  - Le conteneur démarre et sert les endpoints.
  - Les couches sont optimisées pour ne pas réinstaller les dépendances lorsque le code change.
  - La documentation sur la façon de générer l'image et l'architecture est fournie.

- *Compétence : Ajouter un logging pertinent pour chaque handler de endpoint. Ajouter un endpoint pour connaitre la version logicielle actuelle. Références de la séquence de cours : devops_6_monitor_operate.*

  Critères d'évaluation :
  - Une ligne de log dans les handlers de endpoint.
  - Le logging systématique des erreurs.
  - Le niveau de log approprié
  - Un endpoint GET /version qui fonctionne et retourne la version du logiciel (string).
  - La documentation sur le endpoint GET /version.

- *Compétence : Concevoir, implémenter et exécuter des tests automatisés des endpoints d'une API. Références de la séquence de cours : devops_4_testing_strategies.*

  Critères d'évaluation :
  - Les tests passent et permettent d'effectuer des contrôles pertinents.
  - La documentation sur la façon d'exécuter les tests localement.

- *Compétence : Définir un pipeline basé sur GitHub Actions qui crée une image docker, l'envoie vers ghcr.io et exécute des tests de l'API. Références de la séquence de cours : devops_7_automate.*

  Critères d'évaluation :
  - Les dépendances des jobs sont correctement définies.
  - La création et l'envoi des images docker vers GHCR.
  - L'exécution des tests de l'API pour la dernière image docker créée.
  - Le pipeline est opérationnel et ne contient pas de secrets en texte brut dans le code.