Projet complet Data Science & Machine Learning - ToDo détaillé pour débutant

Ce document est un guide étape par étape sur 8 semaines (2 mois), destiné à un débutant qui souhaite réaliser un projet complet de Data Science et Machine Learning avec des outils modernes. Chaque jour a une tâche claire, réalisable, qui mène au résultat final : une application prédictive complète, conteneurisée avec Docker.

Semaine 1 – Mise en place et compréhension du projet

• Jour 1 : Installer Git, VS Code, Python (Anaconda recommandé)
  - Tutoriel Git : https://git-scm.com/book/en/v2
  - Installer VS Code : https://code.visualstudio.com/
  - Installer Anaconda : https://www.anaconda.com/

• Jour 2 : Créer un repo GitHub, cloner localement, organiser le projet
  - Fichiers : /data, /notebooks, /scripts, /models, /app, README.md
  - Commencer un README décrivant le projet

• Jour 3 : Rechercher un jeu de données sur Kaggle ou HuggingFace Datasets
  - Exemple : Customer Support on Twitter (https://www.kaggle.com/datasets/thoughtvector/customer-support-on-twitter)
  - Télécharger et organiser les données dans le dossier `/data`

• Jour 4 : Créer un environnement virtuel (conda ou venv), installer les libs : pandas, numpy, matplotlib, seaborn, jupyter
  - `conda create -n ml_project python=3.10`
  - `conda activate ml_project`
  - `pip install pandas numpy matplotlib seaborn jupyter`

• Jour 5 : Charger et afficher les données dans un notebook Jupyter
  - Identifier les colonnes utiles, types, données manquantes

• Jour 6 : Lire un tutoriel sur le nettoyage des données textuelles et numériques
  - Exercice : nettoyer les noms de colonnes, supprimer les doublons

• Jour 7 : Backup GitHub + résumé de la semaine dans le README

Semaine 2 – Prétraitement des données

• Jour 1 : Nettoyer les textes (lowercase, remove punctuation, stopwords)
• Jour 2 : Encoder les catégories (LabelEncoder, OneHot)

• Jour 3 : Gérer les valeurs manquantes (imputation, suppression)
• Jour 4 : Normaliser/standardiser les données numériques (StandardScaler)
• Jour 5 : Split en X (features) / y (target)
• Jour 6 : Diviser jeu de données : train / validation / test
• Jour 7 : Commit GitHub + résumé dans README

Semaine 3 – Analyse exploratoire (EDA)

• Jour 1 : Visualiser les distributions (histogrammes, boxplots)
• Jour 2 : Corrélation (heatmap), pairplot, analyse multivariée
• Jour 3 : WordCloud, TF-IDF, analyse de texte de base
• Jour 4 : Générer des insights (ex. satisfaction par canal)
• Jour 5 : Préparer une synthèse visuelle (matplotlib/seaborn)
• Jour 6 : Préparer les features finales pour le modèle
• Jour 7 : Backup GitHub + résumé dans README

Semaine 4 – Modélisation ML

• Jour 1 : Implémenter un modèle de base (LogisticRegression, DecisionTree)
• Jour 2 : Entraîner et évaluer (accuracy, F1, confusion matrix)
• Jour 3 : Implémenter XGBoost et Random Forest
• Jour 4 : GridSearch + validation croisée
• Jour 5 : Sauvegarder le modèle (`joblib`, `pickle`)
• Jour 6 : Créer un notebook de comparaison de modèles
• Jour 7 : GitHub commit + update README

Semaine 5 – Modèle avancé NLP (Textes)

• Jour 1 : Installer `transformers`, `datasets`, `torch`
• Jour 2 : Tokenisation avec BertTokenizer ou DistilBERT
• Jour 3 : Fine-tuning sur données textes (5k lignes max)
• Jour 4 : Évaluer le modèle NLP (F1, recall)
• Jour 5 : Fusionner le modèle texte + numérique (pipeline)
• Jour 6 : Créer une fonction `predict()` unifiée
• Jour 7 : Backup + GitHub + documentation

Semaine 6 – Création d'une API REST avec FastAPI

• Jour 1 : Installer FastAPI, Uvicorn
• Jour 2 : Créer une API de prédiction simple (GET + POST)

• Jour 3 : Intégrer le modèle sauvegardé
• Jour 4 : Tester avec curl, Postman
• Jour 5 : Ajouter gestion des erreurs
• Jour 6 : Créer un fichier requirements.txt
• Jour 7 : Commit + GitHub + test complet

Semaine 7 – Interface utilisateur avec Streamlit

• Jour 1 : Installer Streamlit
• Jour 2 : Créer interface simple avec formulaire utilisateur
• Jour 3 : Appeler l'API et afficher la prédiction
• Jour 4 : Ajouter graphiques (ex: score de confiance)
• Jour 5 : Styliser l'interface (layout, CSS de base)
• Jour 6 : Test complet
• Jour 7 : Backup + vidéo démo (facultatif)

Semaine 8 – Dockerisation et finalisation

• Jour 1 : Installer Docker Desktop
• Jour 2 : Créer Dockerfile pour API + modèle
• Jour 3 : Créer docker-compose.yml (API + UI)
• Jour 4 : Build + test local de l'image
• Jour 5 : Push sur Docker Hub (optionnel)
• Jour 6 : Nettoyer code, notebooks, commentaires
• Jour 7 : Faire un README final + rapport PDF