Project 3: Federated Document Classification for Customer Support

Project Description

The goal of this project is to build a federated learning (FL) model that classifies customer support documents while preserving data privacy. In traditional document classification, customer support centers must aggregate sensitive customer data into a central location for model training, which poses privacy risks. Federated Learning addresses this by training the model directly on decentralized data at different customer support branches or locations without sharing raw data, thus maintaining customer confidentiality.

Project Goals:

1. Develop an NLP-based document classifier using embeddings and classification algorithms.
2. Implement a federated learning setup where model updates are shared instead of raw data.
3. Simulate multiple clients to represent decentralized customer support centers.
4. Evaluate model performance across different clients and assess the effects of federated learning on data privacy and model accuracy.

Key Steps:

1. **Data Collection & Preprocessing**: Use customer support datasets, such as the Customer Support on Twitter dataset, with tokenization and embeddings (e.g. BERT, Word2vecm, TF-IDF).
   Dataset Link : https://www.kaggle.com/datasets/thoughtvector/customer-support-on-twitter

2. **Local Model Development**: Start with simpler models (e.g., logistic regression with TF-IDF) and advance to transformer-based models (e.g., BERT). Train and evaluate a document classifier locally before federating.
   Important: You are free to choose the models you prefer.

3. **Federated Learning Setup**: Use libraries like PySyft, Flower, or TensorFlow Federated to set up decentralized training. Simulate multiple clients with local data, aggregate their model updates using federated averaging.
   I recommend Flower Library, an inetersting link for flower tutorial
   https://www.deeplearning.ai/short-courses/intro-to-federated-learning/

4. **Model Aggregation and Evaluation**: Deploy a server to collect and average client updates. Assess the model’s performance (accuracy, precision, recall, F1 score) and analyze privacy benefits.
5. **Optimization and Tuning**: Tune hyperparameters (learning rates, batch sizes) and benchmark federated versus centralized training to optimize accuracy and communication efficiency while preserving privacy.

6. **Evaluation**: Test the model on holdout data, assess accuracy, and analyze privacy benefits. Compare federated learning results with centralized training and analyze trade-offs in accuracy, communication costs, and privacy.