Medicine Recommendation System

**Step 1: Import Libraries and Load Data**

* **Purpose**: Set up the environment for data processing and machine learning.
* **Actions**:
  + Import libraries such as pandas for data manipulation, numpy for numerical operations, and scikit-learn and TensorFlow for machine learning and deep learning tasks.
  + Load the dataset containing medical symptoms and their corresponding diagnoses from a CSV file.

**Step 2: Preprocess the Data**

* **Purpose**: Prepare the data for modeling by cleaning and formatting it.
* **Actions**:
  + Separate the dataset into features (symptoms) and labels (diagnoses).
  + Use a LabelEncoder to convert categorical labels into numerical format, which is essential for machine learning algorithms.
  + Split the dataset into training and testing sets (e.g., 70% training, 30% testing) to ensure that the model can be evaluated on unseen data.

**Step 3: Define Classification Models**

* **Purpose**: Establish a variety of models to find the best one for the task.
* **Actions**:
  + Define multiple classification algorithms, including:
    - **Support Vector Classifier (SVC)**: Effective for high-dimensional spaces.
    - **Random Forest**: An ensemble method that improves accuracy by combining multiple decision trees.
    - **Gradient Boosting**: Another ensemble technique that builds models sequentially to minimize errors.
    - **K-Neighbors**: A simple algorithm that classifies based on the closest training samples.
    - **Naive Bayes**: A probabilistic classifier based on Bayes' theorem.

**Step 4: Build the GAN Model**

* **Purpose**: Create a Generative Adversarial Network to generate synthetic data.
* **Actions**:
  + **Generator**: A neural network that takes random noise as input and produces synthetic data resembling the training data.
  + **Discriminator**: Another neural network that evaluates whether the input data is real (from the dataset) or fake (generated by the generator).
  + The generator and discriminator are trained in opposition to each other, improving their performance over time.

**Step 5: Train the GAN**

* **Purpose**: Improve the GAN's ability to generate realistic data.
* **Actions**:
  + For each training epoch:
    - Generate random noise and use the generator to create synthetic samples.
    - Randomly select real samples from the training data for comparison.
    - Train the discriminator on both real and synthetic data, adjusting its weights based on how well it distinguishes between the two.
    - Train the generator by trying to fool the discriminator into thinking the synthetic data is real.

**Step 6: Generate Synthetic Data**

* **Purpose**: Augment the training dataset to improve model robustness.
* **Actions**:
  + After training the GAN, use it to generate a specified number of synthetic samples (e.g., 5000).
  + Combine the synthetic data with the original training data, creating an augmented dataset that provides more examples for the models to learn from.

**Step 7: Train Classifiers on Augmented Data**

* **Purpose**: Evaluate the performance of various models using the enhanced training set.
* **Actions**:
  + For each defined model, fit it to the augmented training data.
  + Use the test set to assess how well each model performs, calculating accuracy and generating confusion matrices to visualize performance across different classes.
  + Print results for comparison, helping to identify the best model for the task.

**Step 8: Make Predictions**

* **Purpose**: Prepare the best-performing model for real-world use.
* **Actions**:
  + Select the model with the highest accuracy (e.g., SVC) for making predictions.
  + Save the trained model using pickle, allowing it to be reused without retraining.

**Step 9: Integrate with MLflow**

* **Purpose**: Enhance experiment tracking and model management.
* **Actions**:
  + Install and set up MLflow, a platform for managing the machine learning lifecycle.
  + Log model parameters, metrics (like accuracy), and the trained model itself during an MLflow run.
  + This allows tracking of different model versions, hyperparameters, and performance metrics over time, facilitating reproducibility and experimentation.

**Step 10: Implement Recommendation System**

* **Purpose**: Provide users with actionable health recommendations based on their symptoms.
* **Actions**:
  + Load additional datasets that contain information about diseases, precautions, medications, and diets.
  + Create a helper function that retrieves relevant information based on the predicted disease.
  + Develop a user interface where users can input their symptoms, and the system predicts the likely disease and provides recommendations for treatment, precautions, and lifestyle changes.

This detailed breakdown provides a comprehensive understanding of each step in the medicine recommendation system, highlighting the purpose and actions involved. This format is suitable for a presentation, as it clearly outlines the workflow and rationale behind each component.

**Team Members**

* **Abdelrahman Ahmed Eldaba** – Data Scientist - [LinkedIn](https://www.linkedin.com/in/abdelrahmaneldaba)
* **Mohamed Yasser Esmaeil** – ML Engineer - [LinkedIn](http://www.linkedin.com/in/mohamed-dawaba-865074263)
* **Mohamed Rabiee Abdallah** - ML Engineer - [LinkedIn](https://www.linkedin.com/in/mohamed-rabie-6247862b6/)
* **Khaled Emad Eddin Salem** - ML Engineer - [LinkedIn](http://www.linkedin.com/in/khaled-emad-397751261)
* **Refaat-Allah Tarek Elgohary** - ML Engineer - [LinkedIn](https://www.linkedin.com/in/refaat-elgohary-07b94b2a3/)

**GitHub Repository**

[Link to Project GitHub](https://github.com/Abdelrahman47-code/DEPI-Graduation-Project-2024)

**Presentation Link**

[Link to Project Presentation](https://gamma.app/docs/Medicine-Recommendation-System-q3ck5j1vija2l9f)