Additional information

Task	Prompt	Research Plan and Status(deepseek-V3)	Research Plan and Status(deepseek-V3)
parkinsons-disease	Go through the data_description.txt file to understand the data and the machine learning task. You can use train_clinical_data.csv, train_peptides.csv, train_proteins.csv, supplemental_clinical_data.csv as training set data, example_test/** contain test set files similar to training sets.If empty data is encountered, it is supplemented to 0. Script train.py already contains custom SMAPE metrics, select the appropriate algorithm or model to complete the task, train epochs within 20 to save time, and use SMAPE metric grouping cross-validation to evaluate the model. The code is saved as a noScript.py, and you need to predict the updrs_[1-4] score of the patients in the test set and modify the updrs_test column in the example_test/test.csv.The prediction results need to be submitted in the form of submission.csv, such as example_test/test.csv. Please make sure that noScript.py is up and running and able to submit submission.csv before you complete the task.Please save the final code.	1. High-level plan: - Understand the data and machine learning task from 'data_description.txt'. - Identify the appropriate algorithm or model for the task. - Modify 'train.py' to select the appropriate model, set the training epoch within 20, and save the training process as 'noScript.py'. - Execute 'noScript.py' and submit the results in 'submission.csv'. 2. Current status: - Step 1: Understand the data and machine learning task from 'data_description.txt'. In progress. - Steps 2-4: Not started.	Research Plan and Status: 1. High Level Plan 1.1 Inspect data_description.txt to understand data schema and task requirements 1.2 Review existing train.py's SMAPE implementation and model architecture 1.3 Verify data loading code handles all required CSV files and null values 1.4 Implement/test model training with cross-validation using SMAPE metric 1.5 Create prediction pipeline for test data 1.6 Validate submission.csv generation 2. Current Status - No steps completed yet

EEG Brainwave Dataset	The task is to predict emotional states based on EEG EEG datasets, exploring data_description.txt files to understand the data. Given a dataset data/emotions.csv, 80% of it is used as the training set and 20% as the test set. Select the appropriate algorithm or model to complete the task, the training epoch should be within 20 to save time, the code should be saved as noScript.py, and the training process and results need to be submitted in the form of submission.csv.	1. High Level Plan: - Understand the data and machine learning task by reading the 'data_description.txt' file. - Summarize the data and task in the research logs. - Fill in the 'train.py' script to train a language model. - Evaluate the model's performance and make necessary adjustments. - Generate predictions for the test set and save them as 'submission.csv' as described in 'evaluation_details.txt'. 2. Current Status: - No steps have been completed yet.	1. Initial Exploration - 1.1 List directory contents to find data_description.txt and emotions.csv [IN PROGRESS] - 1.2 Review data_description.txt to understand features and labels - 1.3 Inspect emotions.csv structure (columns, data types) 2. Data Preprocessing Plan - 2.1 Handle missing values - 2.2 Split into 80% train / 20% test - 2.3 Normalize/scale features if needed 3. Model Development - 3.1 Select baseline model (e.g., Random Forest or simple neural network) - 3.2 Implement training loop with ¡=20 epochs - 3.3 Evaluate accuracy on test set 4. Output Generation - 4.1 Save trained model - 4.2 Generate submission.csv with predictions

PhysioNet CinC	Classification of AF from short single-lead ECG recordings improves the accuracy of baseline model predictions. You can browse data_description.txt files to learn about the data and machine learning tasks. Given the dataset training2017/*, 90% is used as the training set and 10% is used as the test set. You can look at the top ten data to see what's going on. Choose the appropriate algorithm or model to complete the classification of AF, I hope you choose the pytorch framework for training, and the training epoch should be within 20 to save time. The code is saved as a noScript.py, and the loss and accuracy of the training process need to be submitted in the form of submission.csv.	1. High Level Plan: - Understand the dataset and machine learning task by exploring 'data_description.txt'. - Split the dataset into training and test sets (90% training,10% test). - Select an appropriate algorithm or model for the task. - Implement the model in a Python script ('noScript.py'). - Train the model with a maximum of 20 epochs. - Save the training process and results in 'submission.csv'. 2. Current Status: - Step 1: Understand the dataset and machine learning task. In Progress - Step 2: Split the dataset. Not Started - Step 3: Select an appropriate algorithm or model. Not Started - Step 4: Implement the model. Not Started - Step 5: Train the model. Not Started - Step 6: Save results. Not Started	Research Plan and Status: 1. Understand Data (In Progress) - 1.1 Locate data_description.txt (In Progress) - 1.2 Analyze dataset structure (labels, features, file formats) - 1.3 Inspect sample ECG files (first 10 files in training2017) 2. Implement PyTorch Model - 2.1 Define data loader - 2.2 Design CNN architecture - 2.3 Implement training loop 3. Train Evaluate - 3.1 Split data into 90% train / 10% test - 3.2 Train for 20 epochs - 3.3 Save loss/accuracy to submission.csv

Diabetes	Go through the data_description.txt file to understand the data and the machine learning task. Given a dataset data/diabetes.csv, 70% is used as the training set, 15% as the test set, and 15% as the validation set. Select the appropriate algorithm or model to complete the task, the training epoch should be within 20 to save time, the code is saved as noScript.py, and the training process needs to be submitted as a submission.csv	1. High Level Plan: - Understand the data and machine learning task by reading the 'data_description.txt' file. - Split the dataset 'data/diabetes.csv' into training, test, and validation sets as specified. - Select an appropriate machine learning algorithm or model for the task. - Implement the model in 'noScript.py' with a training epoch within 20. - Train the model and save the training process as 'submission.csv'. - Confirm the performance of the model and submit the final answer. 2. Current Status: - Step 1: Understand the data and machine learning task. Not started. - Step 2: Split the dataset into training, test, and validation sets. Not started. - Step 3: Select an appropriate algorithm or model. Not started. - Step 4: Implement the model in 'noScript.py'. Not started. - Step 5: Train the model and save the training process as 'submission.csv'. Not started. - Step 6: Confirm the performance of the model and submit the final answer. Not started.	1. Check if data_description.txt exists in the current directory. - Status: Pending 2. Read data_description.txt to understand the dataset and task. - Status: Pending 3. Inspect the first 10 rows of diabetes.csv to verify data structure. - Status: Pending 4. Determine the machine learning task type (e.g., classification/regression) based on data_description.txt and data inspection. - Status: Pending 5. Select a model suitable for the task with training 20 epochs . - Status: Pending 6. Implement the model training, validation, and testing pipeline in noScript.py. - Status: Pending 7. Execute noScript.py to generate submission.csv and verify correctness. - Status: Pending

Respiratory Sound Database	The task is to classify respiratory diseases such as asthma, pneumonia, bronchiolitis, etc. 1. Browse the data_description.txt file to understand the data. 2. Extract data features from WAV files, and preprocess and enhance the data.You can look at the top 10 data to see what it looks like. 3. 80% of the enhanced data is used as the training set and 20% is used as the test set. 4. Please use deep learning or machine learning based on the PyTorch framework to classify respiratory diseases, and the training epoch should be less than 100 to save time. 5. The final code should be saved as a noScript.py, and the training process and results need to be submitted in the form of submission.csv.	The high level research plan is as follows: 1. Understand the data by browsing the data_description.txt file. 2. Extract data features from WAV files and preprocess the data. 3. Split the data into training and test sets. 4. Use PyTorch to create a deep learning or machine learning model to classify respiratory diseases. 5. Train the model with less than 100 epochs and save the final code as noScript.py. 6. Submit the training process and results in submission.csv. Current status: We have not started any steps yet.	1. Read data_description.txt to understand dataset structure, features, and labels. 2. Inspect sample WAV files to determine preprocessing needs. 3. Develop feature extraction and preprocessing code. 4. Split data into train/test sets. 5. Design and train PyTorch model. 6. Save code as noScript.py and generate submission.csv.