Regression Models for Cancer Mortality Prediction

Overview

This project implements regression models to predict TARGET_deathRate using the dataset cancer_reg-1.csv. Models included:

- Linear Regression
- Deep Neural Networks with 1, 2, 3, and 4 hidden layers

Dependencies

Install the required packages:

Go to the project folder and enter this to download all packages

- `pip install -r requirements.txt`

File Directory

- main.py – Contains the function code of all the models (Linear Regression and DNNs)
- test.py – Contains function to test different models (Created for easier grading)
- image/ - folder containing images of all plots and model training iteration. I have learning rate to classify the model images
- DeepNeuralNetwork_30_16_8.pt – Is the model state saved of the best model found

How to Run

1. Run the Best Model

   - The test_model() method in test.py executes the best model found by comparing the R² score

     ```
     from main import DeepLearning

     DeepLearning().test_model()
     ```

2. Run a specific model:

   - Choose one model and specify hyperparameters (learning rate, epochs, nonlinear function)

```
DeepLearning.run_per_model(
   learning_rate=0.01,
   model_name="DeepNeuralNetwork_30_16_8",  # choose from list below
   epochs=1000, # choose epoch
   nonlinear="Tanh"  # choose activation
)
```

- Available Models:
  - LinearRegression
  - DeepNeuralNetwork_16 — One hidden layer of 16 neurons
  - DeepNeuralNetwork_30_8 — Two hidden layers with 30 and 8 neurons
  - DeepNeuralNetwork_30_16_8 — Three hidden layers with 30, 16, and 8 neurons
  - DeepNeuralNetwork_30_16_8_4 — Four hidden layers with 30, 16, 8, and 4 neurons

3. Run All Models for a Specific Learning Rate

- The test_per_learning_rate() function executes all models (Linear + DNNs) with predefined epochs and activation functions tuned for each learning rate:

```
DeepLearning().test_per_learning_rate(lr=0.01)
```

- Available learning rates:
  - 0.1
  - 0.01
  - 0.001
  - 0.0001

Output

- Training and validation losses are printed on the console.
- Loss plots and training execution images are saved in the images/ folder. It is categorized using learning rate values.
- Final R^2 score is displayed for the test set and listed in the report document.

Notes

- Activation functions available: "Sigmoid", "Tanh", "ReLU", "LeakyReLU"