SC4001/4042: Neural Networks and Deep Learning

Programming Assignment

Part A: Classification problem

- DNN to classify the **Musical Genre** dataset: **~2000** audio tracks, spanning 3 seconds each.
- The dataset has been pre-processed and **57 features** have been extracted: **audio_gtzan.csv**. Extracted using **librosa** library
- Classification task: classify whether the audio track (in the form of engineered features) belongs to the **blues** or **metal** genre.
- Start with **PartA_Template.ipynb**.

Part A

1. DNN with **Four hidden layer** (256 → 128 → 64 → 32 ReLU units), GD with ‘Adam’ optimizer. Dropout of probability 0.2 and L2 weight decay with 0.01. Divide the dataset into 75:25 train and test. Use **early-stopping**.
2. Use 5-fold CV to determine the **optimal learning rate** from {0.0001, 0.0005, 0.001, 0.005, 0.01}. Report time-taken.

Part A

3. Use 5-fold CV to determine the **optimal number of first hidden-layer** neurons from {64, 128, 256}.

4. Run model inference using the provided audio track data named ‘**audio_test.wav**’ (use the preprocessing function 'extract_features' in **common_utils.py**), find **the most important features via SHAP** and **identify the top 5 most influential features**. (import shap)

Part B: Regression problem

- The aim is to predict public housing prices in Singapore from related features (#10):
  - **Numeric features**: dist_to_nearest_stn, dist_to_dhoby, degree_centrality, eigenvector_centrality, remaining_lease_years, floor_area_sqm
  - **Categorical features**: month, town, flat_model_type, storey_range
- Data: **hdb_price_prediction.csv**.
- Start with **PartB_Template.ipynb**.

Part B1: Data Preparation

- Objective: Prepare the Housing Price Dataset for training and evaluation

- Tasks
  - Encode categorical features using OrdinalEncoder from scikit-learn
  - Scale continuous features using StandardScaler from scikit-learn
  - Implement a PyTorch Dataset and create train, validation and test dataloaders

https://scikit-learn.org/

Part B2: Model and Training

- Objective: Create and train a small MLP to predict house prices
- Tasks
  - Create a module to embed the categorical features
  - Create a model that concatenates the continuous features with embeddings for categorical features and predicts the price by using **nn.Module**
  - Use PyTorch **lightning.LightningModule** to train the model
    - Implement checkpointing, early stopping, learning rate tracking and Tensorboard logging
  - Report test RMSE and R^2 values, and display plot for validation loss

https://lightning.ai/docs/pytorch/stable/

Part B3: Integrated Gradients

- Objective: Use **Integrated Gradients** to interpret the model's predictions
- Tasks
  - Use Integrated Gradients implementation from **captum** library to interpret the model's predictions
  - Local Explanation: For one sample from the dataset, plot the signed contribution for continuous features
  - Global Explanation: Run IG on all samples in the validation set and plot the attributions for each continuous feature

https://captum.ai/

https://captum.ai/docs/extension/integrated_gradients

Notes

- Marking based on your codes in Jupyter notebooks.
- Marks: **45** for Part A + **45** for Part B + **10** for presentation.
- Late submissions: penalized for 5 marks for each day up to 3 days.
- This assignment is to be done **individually**. Absolutely NO copying, duplicating, or plagiarism. You can discuss it with your classmates, but your submission must be your own unique work.
- **Follow the format** in the 2 notebooks provided.
- Post your queries on the **Discussion Board** in NTULearn (TAs will update a list of FAQ in there).
- Approach TAs Deepank Girish, Yan Yige, Tiara Natasha, and Ajith Senthisenan for help via cx4042@e.ntu.edu.sg