

Programming Assignment

SC4001/4042: Neural Networks and Deep Learning

Deadline: October 10, 2025

- This assignment is to be done **individually**. You can discuss the questions with others but **your submission must be your own unique work**.
- Data files and other supporting codes for both parts are found in the folder ‘Programming Assignment’ under ‘Assignments’ on NTULearn. You must use the templates to begin and provide answers to the assignment. Please follow the formats given in the template to fill in your solutions. You may add cells to the template as needed to make your solutions clearer.
- The assessment will be based on the correctness of the codes and solutions. **Each part carries 45 marks, and 10 marks are assigned to the presentation and clarity of the solutions. The total score of marks is 100.**
- You do not need GPUs for this assignment. Local PCs will be sufficient. Attempt your assignments using Jupyter Lab / Notebook with version 3 and greater.
- TAs Ms Yan Yige, Ms Tiara Natasha, Mr Girish Kumar Deepank and Mr Ajith Santhisenan are in charge of this assignment. Please email cx4042@e.ntu.edu.sg for any queries regarding the assignment. Note that you are requested to post your queries on the **Discussion Board**.

Submission procedure

- **Complete both parts A and B** of the assignment and submit your solutions online via NTULearn before the deadline.
- All submissions should be within the notebooks provided. **Do not** include data nor model checkpoints in your submission. Only submit **2 notebooks** and **common_utils.py** (for part A) in this format:
 - Two files for part A
 - └ └ └ **<lastname>_<firstname>_PartA.ipynb**
 - └ └ └ **common_utils.py**
 - One file for part B
 - └ └ └ **<lastname>_<firstname>_PartB.ipynb**
- Late submissions will be penalized: **5% for each day up to three days**.

Part A: Classification Problem

Part A of this assignment aims at building neural networks to perform music genre classification from music recordings, based on data in the GTZAN dataset, which is obtained from [here](<https://www.kaggle.com/datasets/carlthome/gtzan-genre-collection>).

The original dataset consists of 1000 audio tracks each 30 seconds long. It contains 10 genres, each represented by 100 tracks. The tracks are all 22050 Hz monophonic 16-bit audio files. In this assignment, we split each audio into 3 seconds long audio files, and only pick two of the music genres, namely blues and metal.

Using the librosa library, the recordings have been preprocessed to extract features such as chromagrams, Mel spectrograms, MFCCs and various other features. The preprocessed csv file is provided in this assignment. We will be using the CSV file named audio_gtzan.csv, which is provided to you. The features from the dataset are engineered. The aim is to determine the music genre from the engineered feature dataset. The csv file is called audio_gtzan.csv with a row of 57 features that you can use, together with the filename. The “filename” column has the labels associated to them.

Type of features	Explanation
Chroma (e.g. chroma_stft_mean)	Describes the <u>tonal content</u> of a musical audio signal in a condensed form (Stein et al, 2009) [2]
Rms (e.g. rms_mean)	Square root of <u>average of a squared signal</u> (Andersson) [3]
Spectral (e.g. spectral_centroid_mean)	Spectral Centroid is a metric of the <u>centre of gravity</u> of the <u>frequency power spectrum</u> (Andersson) [3]
Rolloff (e.g. rolloff_mean)	Spectral rolloff is a metric of <u>how high in the frequency spectrum</u> a certain part of energy lies (Andersson) [3]
Zero crossing (e.g. zero_crossing_mean)	Zero-crossing rate is the number of time domain zero-crossings <u>within a processing window</u> (Andersson) [3]
Harmonics (e.g. harmony_mean)	Sound wave that has a <u>frequency</u> that is a <u>n integer multiple</u> of a <u>fundamental tone</u> Refer to link: https://professionalcomposers.com/what-are-harmonics-in-music/
Tempo	Periodicity of note onset pulses (Alonso et al, 2004)
MFCC (Mel Frequency Cepstral Coefficient)	Small set of features (usually about 10-20) which concisely describe the <u>overall shape of a spectral envelope</u> Refer to link: https://musicinformationretrieval.com/mfcc.html

Question A1 (15 marks)

Design a feedforward deep neural network (DNN) which consists of four hidden layers with tapering width: $256 \rightarrow 128 \rightarrow 64 \rightarrow 32$ neurons with ReLU activation function, and an output layer with sigmoid activation function. Apply dropout of probability 0.2 to each of the hidden layers and implement L2 weight decay with weight_decay = 0.01.

Divide the dataset into a 75:25 ratio for training and testing. Use appropriate scaling of input features. We solely assume that there are only two datasets here: training and test.

Use the training dataset to train the model for 100 epochs. Use a mini-batch gradient descent with 'Adam' optimizer with learning rate of 0.001, and batch size = 128. Implement early stopping with patience of 5.

Plot train and test accuracies and losses on training and test data against training epochs and comment on the line plots.

Question A2 (10 marks)

In this question, we will determine the optimal learning rate for mini-batch gradient descent by training the neural network and evaluating the performances for different learning rates. Note: Keep the batch size fixed at 128. Use 5-fold cross-validation on the training partition to perform hyperparameter selection. You will have to reconsider the scaling of the dataset during the 5-fold cross validation.

Plot mean cross-validation accuracies and mean cross-validation AUC scores on the final epoch for different learning rates as a scatter plot. Limit search space to learning rates $\{0.0001, 0.0005, 0.001, 0.005, 0.01\}$.

Next, create a table of time taken to train the network on the last epoch against different learning rates.

Additionally, generate and submit a confusion matrix for the selected optimal configuration.

Finally, select the optimal learning rate and state a reason for your selection.

Question A3 (10 marks)

In this question, we will find the optimal number of hidden neurons for the first **hidden layer** of the **5-layer network (4 hidden layers, output layer)** designed in **Question 1 and 2**.

Plot the mean cross-validation accuracies on the **final epoch** for different numbers of hidden-layer neurons using a scatter plot. Limit the search space of the number of neurons to {64, 128, 256}. Continue using 5-fold cross validation on the training dataset.

Additionally, plot the ROC curve on the **validation set** for the final selected architecture.

Select the optimal number of neurons for the hidden layer. State the rationale for your selection.

Plot the **train and test accuracies against training epochs** with the optimal number of neurons using a line plot.

Note: **use this optimal number of neurons for the rest of the experiments.**

Question A4 (10 marks)

In this section, we will explore the utility of such a neural network for test audio samples.

Please use the 'audio_test.wav' as a **test sample**. Preprocess the data using the function 'extract_features' in common_utils.py.

Do a model prediction on the sample test audio and obtain the predicted label using a threshold of 0.5. The model used is the **optimized pretrained model** using the selected optimal learning rate and optimal number of neurons.

Find the **most important features** on the model prediction for the test sample using SHAP. Plot the local feature importance with a force plot and explain your observations.

Identify the top 5 most influential features (both positive and negative) and explain why these features make sense for music genre classification

(Refer to the documentation and these three useful references:

<https://christophm.github.io/interpretable-ml-book/shap.html#examples-5>,

<https://towardsdatascience.com/deep-learning-model-interpretation-using-shap-a21786e91d16>,

<https://medium.com/mlearning-ai/shap-force-plots-for-classification-d30be430e195>)

Part B: Regression Problem

This assignment uses publicly available data on HDB flat prices in Singapore, obtained from data.gov.sg on 17th August 2023. The [original dataset](#) is combined with other datasets to include more informative features and they are given in the '[hdb_price_prediction.csv](#)' file.

Feature	Type	Explanation
month	Categorical (Int)	Which month the resale transaction was performed.
year	Categorical (Int)	Which year the resale transaction was performed. Used for train/test split. NOT used to train the model.
town	Categorical (Str)	An area zoned for public housing. Generally, a town consists of a self-sufficient group of HDB flats with amenities and a town centre for that town. Each town has a population of around 50,000 - 250,000.
full_address	Categorical (Str)	Address of the flat. Not used for modelling as other metrics derived from it are used instead (dist_to_nearest_stn, dist_to_dhoby).
nearest_stn	Categorical (Str)	Closest MRT station to the flat. Not used for modelling as other metrics derived from it are used instead (degree_centrality, eigenvector_centrality).
dist_to_nearest_stn	Numeric	Distance from the flat to the nearest MRT station, in kilometres. Flats near stations tend to fetch higher prices.
dist_to_dhoby	Numeric	Distance from the flat to Dhoby Ghaut MRT station, in kilometres. Dhoby Ghaut is chosen as it is centrally located. Flats in the Central region are typically more costly.
degree_centrality	Numeric	A metric (computed for the MRT station closest to the flat) that represents the degree of the node, i.e., how many edges are connected to the node. (Rationale: flats near 'interchange' stations - stations with more than 1 MRT line - offer more transport options and thus have higher value. Stations in the central areas tend to have > 1 MRT line too).
eigenvector_centrality	Numeric	A more global metric than degree_centrality as it captures neighbourhood information. When the eigenvector centrality of a node is high, the nodes adjacent to it are likely to have high values too.
flat_model_type	Categorical (Str)	Type of flat. See this reference for more details. You're not expected to understand all flat types.
remaining_lease_years	Numeric	HDB flats are originally sold by HDB with a 99-year lease. Generally, with other variables held constant, flats with higher remaining lease will fetch a higher value. The original data was stored in years and months – this was turned into a scalar by converting it into months and dividing by 12.
floor_area_sqm	Numeric	Size of the flat in square meters. Bigger flats generally fetch higher prices.
storey_range	Categorical (Str)	Which floor the flat is at. For the same HDB block, flats on higher floors typically fetch higher prices..
resale_price	Numeric	Flat prices in Singapore Dollars. Target to predict.

14
cols



Question B1 (10 marks)

In this part, you will prepare the housing price dataset for training and evaluation. You will clean, encode, and split real-world tabular data to make it suitable for deep learning. Detailed instructions for each step are provided in the Jupyter notebook for Part B.

Perform the following steps:

- Load the dataset (`hdb_price_prediction.csv`).
- Split into training, validation, and test sets.
- Apply preprocessing:
 - Encode categorical features (`month`, `town`, `flat_model_type`, `storey_range`).
 - Scale continuous features (`dist_to_nearest_stn`, `dist_to_dhobby`, `degree_centrality`, `eigenvector_centrality`, `remaining_lease_years`, `floor_area_sqm`).
- Create a custom PyTorch Dataset and DataLoader for batching.

Report the number of samples in each split, and show a small batch (5 rows) from your DataLoader to verify correctness.

Question B2 (25 marks)

a. Create a PyTorch Model

- Build embedding layers for categorical features and concatenate with continuous features.
- Implement an MLP with ReLU activations and dropout; use MSE as the loss.

b. Write a LightningModule for fitting the model

- Define the model, loss function (MSE), optimizer (Adam), and metrics (MAE, RMSE).
- Add callbacks: `ModelCheckpoint` (save best model) and `EarlyStopping` (monitor validation loss).
- Use a logger (e.g., TensorBoard) for monitoring.

c. Train the model

- Train with a Lightning Trainer (up to 50 epochs).
- Report training and validation RMSE and plot training/validation loss curves.

d. Predict using the trained model on the test set

- Report test RMSE and test R2.

Detailed instructions for each step are provided in the Jupyter notebook for Part B.

Question B3 (10 marks)

Building accurate models is important, but interpreting them is equally crucial. Explainability techniques will help you connect model predictions to domain intuition. In this part, you will use Captum's Integrated Gradients (IG) method to analyze the impact of continuous features.

Perform the following steps:

- Apply IG to 1 example from the test set and show the top 3 most influential features.
- Apply IG across the whole validation set to compute global average feature importance.
- Plot a bar chart of feature importances.

- Discuss briefly: do the top features (e.g., floor area, distance to nearest station) make intuitive sense?

Detailed instructions for each step are provided in the Jupyter notebook for Part B.

References

- [1] Koh JX, Mislan A, Khoo K, Ang B, Ang W, Ng C, Tan YY. Building the Singapore English national speech corpus. Malay. 2019;20(25.0):19-3
- [2] Stein M, Schubert BM, Gruhne M, Gatzsche G, Mehnert M. Evaluation and comparison of audio chroma feature extraction methods. InAudio Engineering Society Convention 126 2009 May 1. Audio Engineering Society.
- [3] Andersson T. Audio classification and content description. 2004.
- [4] Miguel Alonso BD, Richard G. Tempo and beat estimation of musical signals. In Proceedings of the International Conference on Music Information Retrieval (ISMIR), Barcelona, Spain 2004.