School of Computer Science Engineering and Technology

Course- B. Tech
Course Code- CSET228
Year- 2024-25
Date: 26-03-2025

Type- Data Science Specialization
Course Name: DMPM
Semester- 4th Sem
Batch- ALL

Lab Assignment 10.1

CO-Mapping

Exp. No.	Name	CO1	CO2	CO3
1.	SVM		✓	
2.	ANN, MLP			✓

Objective:

- Students will understand the implementation of **Support Vector Machines (SVM)**, **Artificial Neural Networks (ANN)**, and **Multilayer Perceptron (MLP)**.
- Students will explore different hyperparameters, activation functions, and optimizers for SVM and ANN.
- Students will evaluate model performance using accuracy, precision, recall, and F1-score.

Q1. Implementing Support Vector Machine (SVM) for Classification. You will train an SVM model on the **Wine Quality dataset**, which is available on the **UCI Machine Learning Repository**.

Dataset Link: https://archive.ics.uci.edu/ml/datasets/wine+quality

Follow the steps below:

- Download and Load the Dataset
- Perform Data Pre-processing to check missing values and normalize the datset
- Split the dataset in 80:20 ratio
- Train SVM model
- Evaluate the performance of the model
- Compute accuracy, precision, recall, and F1-score.
- Experiment with different kernels (**RBF, polynomial**) and compare results.

Q2. Implementing ANN & MLP for breast cancer classification. The dataset contains features extracted from **digitized images of breast cancer cell nuclei**. The goal is to classify tumors as **malignant (0) or benign (1)**.

Dataset link:

https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic

School of Computer Science Engineering and Technology

Follow the steps below to implement **both ANN (using TensorFlow/Keras) and MLP (using Scikit-learn)** to classify breast cancer tumors.

Step 1: Load and Explore the Dataset

i. Import the dataset using load_breast_cancer from sklearn.datasets.
ii. Convert the dataset into a Pandas **DataFrame**.
iii. Display the first five rows using df.head().
iv. Print dataset statistics using df.describe().

Step 2: Data Pre-processing

i. Check for missing values using df.isnull().sum().
ii. Standardize features using StandardScaler from sklearn.preprocessing.
iii. Convert target labels (Malignant = 0, Benign = 1) to categorical format (only for ANN).

Step 3: Splitting the Data

Use train_test_split: 80% training data, 20% testing data

Step 4: Build the ANN Model

i. Import Sequential from tensorflow.keras.models.
ii. Add **two hidden layers** using Dense with **ReLU activation**.
iii. Use **sigmoid activation** for the output layer (since it's a binary classification problem).

Step 5: Compile and Train the ANN Model

i. Use **binary_crossentropy** as the loss function
ii. Use **adam** as the optimizer
iii. Train the model using model.fit()
Epochs: 50
Batch size: 32

Step 6: Build the MLP Model

i. Import MLPClassifier from sklearn.neural_network.
ii. Set the following parameters:
Activation Function: ReLU
Optimizer: Adam

Step 7: Train the MLP Model: Train the model using fit().

Step 8: Performance Evaluation

For ANN (Keras Model):

Evaluate the model on test data using model.evaluate().
Compute precision, recall, and F1-score using classification_report()
Plot loss and accuracy curves using matplotlib.

For MLP (Scikit-learn Model):

Predict labels for the test data.
Compute accuracy, precision, recall, and F1-score using classification_report.
Compare the ANN and MLP results.

Additional Fun (Not Evaluated)

- Try different optimizers (SGD, RMSProp) for ANN.
- Modify MLP architecture (increase layers/neurons) and observe performance.
- Add **Dropout** and **Batch Normalization** layers in ANN.