Lesson Plan:

Applications Development Laboratory (**Machine Learning Applications**)

Course Code CS33002 (L-T-P-Cr: 0-0-4-2)
Credits: 2
Session: December-2024 to May-2025
Course faculty: Dr Partha Pratim Sarangi
Email: pp.sarangifcs@kiit.ac.in

Week 1: Introduction to Machine Learning

Objective: Understand the basics of machine learning and Python / R,etc

Experiment:

- Write a Python program to demonstrate data preprocessing steps: handling missing values, encoding categorical data, and feature scaling.
- Load a sample dataset using pandas (e.g., Iris or a custom dataset).
- Plot the distribution of a feature using matplotlib.pyplot.hist().
- Create scatter plots to understand relationships between features using seaborn.scatterplot().
- Use a correlation heatmap to find the relationship between multiple features with seaborn.heatmap().

Week 2: Regression Analysis

Objective: Learn simple linear regression for predictive analysis.

Experiment:

Implement simple linear regression to predict house prices based on features such as area and number of rooms.

- Evaluate the model using metrics such as mean_squared_error and r2_score.

- Visualize the regression line on a scatter plot of the data

Note: *A common dataset we can explore is the "House Prices - Advanced Regression Techniques" dataset from kaggle. The UCI Machine Learning Repository has many datasets for classification, regression, and clustering tasks. A popular dataset related to housing is the "California Housing Prices" dataset.*

Week 3: Multiple Linear Regression

Objective: Extend regression analysis to handle multiple predictors.

Experiment:

Implement multiple linear regression to predict student performance based on study hours, class attendance, and assignment scores.

- Calculate metrics like Mean Squared Error (MSE) and R^2 Score to assess performance.

- Create a scatter plot comparing the actual vs. predicted values

Note: We can get Student Performance Dataset from kaggle

Week 4: Classification with Logistic Regression

Objective:Understand binary classification using logistic regression.

Experiment:

Build a logistic regression model to classify emails as spam or not spam.

- Accuracy: A percentage indicating how many emails were correctly classified.

- Assess model performance using accuracy, confusion matrix, and classification report (Classification Report: Precision, Recall, and F1-score for both spam and non-spam classes)

Note: We will use a publicly available dataset such as the SpamBase dataset from the UCI Machine Learning Repository or create a simple synthetic dataset for demonstration purposes

Week 5: Decision Trees

Objective: Learn decision tree classifiers for non-linear decision boundaries.

Experiment:

Develop a decision tree model to classify species in the ***Iris dataset.***

- Evaluate the model using accuracy and a detailed classification report with precision, recall, and F1-score for each species.

- Use plot_tree() to visualize the decision tree. This helps understand how the model splits the data at different nodes.

Week 6: Random Forests

Objective: Understand ensemble methods using random forests.

Experiment:

Implement a random forest classifier to predict customer churn in a telecom dataset.

- Accuracy: Find overall prediction accuracy.

- Confusion Matrix: Breakdown of true/false positives and negatives.

- Classification Report: Includes precision, recall, and F1-score for both churned and retained classes.

Note: You can use a real-world dataset such as the Telco Customer Churn Dataset available on Kaggle or create a similar dataset for practice.

Week 7: Support Vector Machines (SVM)

Objective: Explore SVM for linear and non-linear classification.

Experiment:

Build an SVM model to classify handwritten digits using the MNIST dataset.

- Accuracy: Overall percentage of correct predictions.
- Confusion Matrix: Provides a breakdown of correct/incorrect predictions for each digit.
- Classification Report: Detailed precision, recall, and F1-score for each digit.

Note:The MNIST dataset consists of 70,000 grayscale images of handwritten digits (28x28 pixels) representing numbers from 0 to 9.
We can use sklearn.datasets.fetch_openml to load the dataset.

Week 8: K-Means Clustering

Objective: Learn unsupervised learning through clustering.

Experiment:

Apply K-Means clustering to group customers based on purchasing patterns.

- Plot the clusters in 2D space (using scaled features) and mark the centroids.
- Analyze each cluster to identify customer groups based on purchasing patterns, such as:
- High-income, high-spending customers.
- Low-income, low-spending customers

Note: We can use the Mall Customers Dataset. This dataset is available in .csv format, typically containing columns such as CustomerID, Gender, Age, Annual Income , and Spending Score (1-100).

Week 9: Principal Component Analysis (PCA)

Objective: Reduce dimensionality using PCA.

Experiment:

Perform PCA on a high-dimensional dataset and visualize the results.

- Displays the proportion of variance captured by each principal component.
- 2D Plot: Projects data onto the first two principal components.
- 3D Plot: Projects data onto the first three principal components.

Note: We can use the Wine Dataset from the sklearn library. It has 13 features (high dimensionality) related to chemical properties of wine samples, which can be reduced to 2 or 3 principal components for visualization.

Week 10: Neural Networks with TensorFlow/Keras

Objective: Introduce deep learning basics with neural networks.

Experiment:

Build a simple neural network to classify handwritten digits using TensorFlow/Keras.

- The images are flattened from 28x28 matrices into 1D vectors of length 784 to be fed into the neural network.
- Input Layer: The input is a flattened vector of size 784 (28x28).
- Hidden Layer: A dense layer with 128 neurons and ReLU activation. This layer allows the model to learn complex patterns.
- Dropout Layer: A dropout layer with a rate of 0.2, which helps reduce overfitting by randomly setting some of the weights to zero during training.

- Output Layer: A softmax layer with 10 neurons for classifying the digits from 0 to 9.

- Show the Test Accuracy, model's performance over epochs(accuracy plot), loss plot indicates whether the model is converging

Note: We can use MNIST Dataset: The MNIST dataset contains 60,000 training images and 10,000 test images of handwritten digits (0-9).

Week 11: Convolutional Neural Networks (CNN)

Objective: Learn image classification using CNNs.

Experiment:

Implement a CNN model to classify images from the CIFAR-10 dataset.

- Show the Test Accuracy, Training vs Validation Accuracy, Training vs Validation Loss and Predictions

Note: CIFAR-10 Dataset: The CIFAR-10 dataset contains 60,000 32x32 color images in 10 classes. CIFAR-10 dataset can be loaded using cifar10.load_data().

Capstone Application (Any of the application must be deployed on the web and accessible through a browser**)**

Objective: Integrate learned concepts into a real-world application.

Experiment:

Hosting an ML application on the web with a front end that can be run in a browser including creating the machine learning model, developing a web interface (front end), setting up a back-end server, and hosting the entire application on the web.

Create a Web Front-End (User Interface):

Languages: Use HTML, CSS, and JavaScript (or frameworks like React, Angular, or Vue.js) to create the user interface.

Develop the Back-End (Server):

Flask or FastAPI (for Python): These are lightweight web frameworks for serving machine learning models.

Connect the Front-End to the Back-End:

Use AJAX or Fetch API in JavaScript to send user input to the back-end (Flask/FastAPI)

Learning Outcomes:

- Understand and apply various supervised and unsupervised machine learning algorithms.
- Build and evaluate models using Python and its libraries (e.g., NumPy, pandas, scikit-learn, TensorFlow).
- Solve real-world problems using machine learning techniques.

Tools Required:

- Python 3.x/ R
- Libraries: NumPy, pandas, matplotlib, scikit-learn, TensorFlow/Keras, NLTK/Spacy