A Proposal for Introductory Data Science Lab

Vipin P and Arjun TU
Center for Cybersecurity Systems and Networks, Amrita School of Computing, Amritapuri
vipinp@am.amrita.edu, arjuntu@am.amrita.edu

1. Objectives of the Virtual Lab

Data Science is recognized as an emerging subject as per AICTE (https://www.aicte india.org/sites/default/files/UG_Emerging.pdf) and the cyber security lab is a critical component of the same.

This virtual lab aims to provide a comprehensive introduction to key concepts and practical techniques with a profound understanding of essential topics in the field of Data Science and Machine Learning. This virtual lab covers a wide spectrum of concepts and algorithms which will help students to acquire a strong foundation in machine learning and its applications.

2. Learning Outcomes

- Understand NumPy library: NumPy, NumPy Arrays and Matrices and other operations in NumPy that are used for Data Science

- Understand Pandas Libray: Provide an introduction to the Pandas library in Python to work with structured data.

- Visualise Data using Matplotlib: Provide an introduction to matplotlib library to visualize data and perform EDA.

- Data Preprocessing: Learners will preprocess raw data by applying cleaning, scaling, and encoding techniques to ensure model readiness.

- Understand Machine Learning Fundamentals: Identify the concept of overfitting and effectively apply train/test splits for model evaluation.

- Categorize Types of Machine Learning: Differentiate between Supervised, Unsupervised, and Reinforcement Learning, and recognize their applications in real-world scenarios.

- Apply Bayesian Probability and Naive Bayes: Apply the principles of Bayesian probability and effectively use the Naive Bayes algorithm for classification and probabilistic modeling.

- Build Linear Regression Models: Create and evaluate linear regression models, taking into account model assumptions and regularization techniques such as Lasso, Ridge, and Elastic Net.

- Implement Classification and Regression Algorithms: Implement a range of classification and regression algorithms, including Naïve Bayes, K-Nearest Neighbors, Logistic Regression, Support Vector Machines (SVM), Decision Trees, and Random Forest.

- Evaluate Classification Models: Utilize various metrics to assess classification errors and make informed decisions for model improvement.

- Model Validation and Performance Evaluation: Learners will assess model performance using cross-validation and relevant evaluation metrics to ensure accuracy and robustness.

- Neural Networks and Generalization: Enables the student to design and train neural models for various applications. They will also comprehend the concept of generalization(like Dropout, Activation Functions etc), allowing them to develop neural networks that can effectively make accurate predictions on unseen data.

- Ethics and Bias: Learners will critically evaluate the presence of bias in datasets and apply techniques to mitigate ethical concerns in AI models.

3. List of experiments

Experiment 1: Introduction to NumPy, Arrays and Array Operations

Aim:

Introduce students to NumPy, NumPy Arrays and Matrices and other operations in NumPy that are used for Data Science.

Objectives:

- Learn how to create, print, index, iterate and perform mathematical operations on NumPy Arrays.
- Learn how to use multi-dimensional Arrays(Matrices) and its operations in Numpy such as matmul, dot, inverse etc. in NumPy.
- Learn how to use other common Numpy functions such as linspace, arange argmax etc.

Process:

In this simulation, students will learn to use and implement the basic functions of the NumPy library using the jupyter notebook provided which contains the code.

Simulation:

- Explore the basic arrays in NumPy and operations on it like slicing, indexing and iterating.
- Demonstrate the use of mathematical operations and functions like np.sin, np.sum etc on NumPy Arrays and give sample problems to solve with expected output.
- Similar to above give sample problems on doing all the operations on NumPy matrices.
- Explain NumPy operations like linspace, arange, argmax etc with the need to use them with example codes.

Experiment 2: Introduction to Pandas

Aim:

The aim of this tutorial is to provide an introduction to the Pandas library in Python and the equip learners with the fundamental knowledge to work with structured data effectively using Pandas.

Objectives:

- Familiarize the learners with pandas library.
- Introduce the core Pandas data structures. Eg: dataframe
- Introduce the data exploration,cleaning and preparation in pandas.

Process:

In this simulation, students will learn to use and implement the basic functions of the Pandas library using the jupyter notebook provided which contains the code.

Simulation:

- Loading data from external sources using Pandas ‘read_’ functions.
- Demonstrate how to save data to different file formats using Pandas ‘to_’ functions
- Explore the data using .head(), .tail(), .describe(), and filtering data with .loc and .iloc.
- Explain grouping data with .groupby() and merging data with .merge().

Experiment 3: Data Visualisation using Matplotlib

Aim:

Learn different methods for visualising data using matplotlib package in Python.

Objectives:

- Understand the different types of plot and when to use them.
- Learn how to draw the plot for the given data.
- Learn how to gain meaningful insights and observations from the plots.

Process:

In this simulation, students will learn to use and implement the basic functions of the Matplotlib library using the jupyter notebook provided which contains the code.

Simulation:

- Generate synthetic data and use it for making different visualizations like line plot, bar chart, scatter plot, box plot etc.
- Load a sample dataset and make different plots with it while explaining how to read them and gain insights about the data.
- Give the students another sample dataset to work with which they can visualize and gain insights from using which they have to attempt MCQ about the dataset.

Experiment 4: Exploratory Data Analysis (EDA)

Aim:

To introduce students to the process of Exploratory Data Analysis (EDA) and its importance in understanding datasets before applying machine learning algorithms.

Objectives:

- Understand the importance of EDA in the data science workflow
- Learn various techniques for exploring and visualizing data
- Gain insights into data distributions, correlations, and potential outliers
- Identify patterns and trends in the data that can inform further analysis

Process:

In this simulation, students will perform EDA on a given dataset using various Python libraries such as Pandas, Matplotlib, and Seaborn.

Simulation:

- Load a diverse dataset (e.g., Iris dataset or Titanic dataset) using Pandas.
- Perform basic statistical analysis like display summary statistics using describe() and check for missing values and data types
- Visualize data distributions like creating histograms for numerical features and generate box plots to identify potential outliers
- Explore relationships between variables and create scatter plots for pairs of numerical features and generate correlation heatmaps
- Analyze categorical variables. Create bar plots to show frequency distributions and use pivot tables to explore relationships between categorical variables
- Do time series analysis and plot time-based trends and identify seasonality and cyclical patterns
- Generate insights and hypotheses based on the EDA findings

Experiment 5: Data Preprocessing

Aim:

To introduce students to essential data preprocessing techniques that prepare raw data for machine learning algorithms.

Objectives:

- Understand the importance of data preprocessing in the machine learning pipeline
- Learn various techniques for handling missing data, outliers, and inconsistencies
- Gain practical experience in feature scaling, encoding categorical variables, and feature engineering
- Prepare data for both supervised and unsupervised learning algorithms

Process:

In this simulation, students will work with a raw dataset and apply various preprocessing techniques using Python libraries such as Pandas, Scikit-learn, and NumPy. There will be few overlaps in this experiment with Experiment 4 in terms of making plots and drawing conclusions from it. But this overlap is deliberate as both these experiments contains a lot of content and students will benefit from repeated usage of the topics taught in EDA after learning preprocessing.

Simulation:

- Load a raw dataset with various data quality issues (e.g., missing values, outliers, mixed data types).
- Handle missing data and identify missing values using isnull() and sum() and apply techniques like mean/median imputation, forward/backward fill, or dropping rows/columns.
- Detect and handle outliers and use visualization techniques (box plots, scatter plots) to identify outliers and apply methods like IQR (Interquartile Range) or Z-score to detect outliers.
- Use feature scaling and implement normalization using Min-Max scaling and learn to apply standardization (Z-score normalization). Also, explore when to use what.
- Encode categorical variables, perform one-hot encoding for nominal categorical variables and apply ordinal encoding for ordinal categorical variables.
- Create new features based on domain knowledge or data insights and if applicable combine existing features to create meaningful new ones

- Use dimensionality reduction techniques like PCA to reduce the dimensionality of the dataset while retaining most of the variance. Visualize the data in the reduced dimension space.

- Handle imbalanced datasets and Identify class imbalance in the target variables.Also, apply techniques like oversampling, undersampling, or SMOTE.

Experiment 6: Evaluation Metrics, Overfitting/Underfitting and Train/Test Splits

Aim:

Investigate the impact of overfitting and the importance of proper data splitting for model evaluation.

Objectives:

- Understand the concept of overfitting and underfitting.
- Learn how to split data into training and testing sets effectively.
- Observe the impact of overfitting/underfitting on model performance.
- Introduce the students to other evaluation metrics like MAE, MSE, ROC, AUC etc.
- Understand about supervised and unsupervised training methods and models.

Process:

In this simulation, the students will load and train a model with a dataset and model of their choice, and examine the accuracy on both testing and training sets.

Simulation:

- Load and preprocess the dataset.
- Split the data into a training set (e.g., 70%) and a testing set (e.g., 30%).
- Train a dummy machine learning model on the training set.
- Evaluate the model's accuracy, precision, recall, ROC-AUC and F1-score on both the training and testing sets.
- Analyze the results to determine if overfitting is occurring by comparing the performance on the two sets.

- Train another regression model and evaluate its performance using MAE and MSE.
- Supervised and Unsupervised models and the difference between them.

Experiment 7: Movie Review Sentiment Analysis using Naïve Bayes

Aim:

To build a spam email filter using the Naive Bayes classification algorithm.

Objectives:

- Understand the concept of spam email detection.
- Preprocess email text data for classification.
- Train a Naive Bayes classifier to distinguish between spam and non-spam emails.
- Assess the accuracy of the spam filter.

Process:

In this simulation, the students will learn to train and understand the working of naive bayes by using it as a binary classifier.

Simulation:

- Obtain a dataset of movie reviews with sentiment labels (positive/negative).
- Preprocess the text data, including removing punctuation and stopwords.
- Split the dataset into training and testing sets.
- Train a Naive Bayes classifier on the training data.
- Apply the model to classify movie reviews as positive or negative.
- Calculate and display sentiment analysis metrics like accuracy, precision, recall, and F1 score.

Experiment 8: Predicting Student Performance Based on Study Hours using Linear Regression

Aim:

The aim of this basic linear regression experiment is to examine the relationship between the number of study hours and students' academic performance to determine whether a linear model can predict a student's final exam score based on their study time.

Objectives:

- Collect data on the study hours and final exam scores of a sample of students.
- Establish a linear regression model that predicts a student's final exam score using study hours as the independent variable.
- Evaluate the model's performance and accuracy in predicting student performance.

Process:

In this simulation, the students will work on the Linear regression model, the dependent/independent variables and the concept of Mean Squared Error.

Simulation:

- Collect data on the study hours and final exam scores of a random sample of students. Ensure the data is accurate and representative of the student population.
- Clean the data by handling missing values and removing outliers.
- Explore the data through summary statistics and visualizations to understand the distribution of study hours and exam scores.
- Establish a simple linear regression model where the dependent variable (Y) is the final exam score, and the independent variable (X) is the study hours.
- Split the dataset into a training set and a testing set.
- Train a Linear Regression model on the training data.
- Evaluate the model's performance on the testing data using appropriate metrics such as MSE or RMSE.

Experiment 9: Predicting Student Performance Based on Study Hours: A Comparison of Linear, Lasso, and Ridge Regression

Aim:

The aim of this experiment is to explore and compare the effectiveness of linear, Lasso, and Ridge regression models for the problem statement in Experiment 6.

Objectives:

- To determine if regularization techniques (Lasso and Ridge) improve the prediction accuracy and model interpretability compared to simple linear regression

Process:

In this simulation, the students will work on regularization.i.e adding a penalty term (L1 or L2 norm) to the cost function of a linear regression model, which reduces the magnitude of the coefficients or weights.

Simulation:

- The starting steps are the same as the previous experiment.
- To implement Lasso regression, add L1 regularization to the linear model. This helps in feature selection by setting some coefficients to zero. The Lasso equation is similar to the linear regression equation with an additional regularization term.
- To implement Ridge Regression Model, add L2 regularization to the linear model. Ridge regression discourages large coefficients and helps improve model stability.
- Train the linear regression, Lasso regression, and Ridge regression models on the training data.
- Evaluate each model's performance on the testing data using metrics like Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2).
- Compare the results to determine which model provides the best prediction accuracy and robustness.

Experiment 10: Predicting Credit Card Fraud using Support Vector Machine

Aim:

The aim of this experiment is to develop and evaluate a Support Vector Machine (SVM) model for the detection of credit card fraud. The objective is to investigate the effectiveness of SVM in distinguishing between legitimate and fraudulent credit card transactions.

Objectives:

- Preprocess and prepare the data for SVM modeling, including data normalization and feature engineering (if necessary).
- Build both a Linear SVM and a Non-linear SVM classifier capable of identifying fraudulent transactions.
- Assess the performance of these SVM models in terms of precision, recall, and accuracy.
- Explore the potential of SVM, with different kernel functions(linear, polynomial, radial basis function).

Process:

In this simulation, the student will work on a credit card fraud detection dataset with different SVM kernels and tune the hyperparameters for each model for optimal performance.

Simulation:

- Acquire a dataset containing historical credit card transaction data
- Preprocess the data as discussed in the above experiments.
- Split the dataset into a training set and a testing set.
- Perform Feature engineering like dimensionality reduction or feature selection(if necessary).
- Implement a Linear SVM classifier and a Non-linear SVM classifier with different kernel functions (polynomial or radial basis function).
- Tune hyperparameters for each model for optimal performance (eg: regularization parameter (c), kernel coefficient (γ)).
- Train the Linear SVM and Non-linear SVM (RBF kernel) models on the training dataset
- For each kernel version, evaluate the performance.
- Create plots or visualizations that compare the performance of these different kernel versions.

- Observe how different kernels handle the credit card fraud detection task and determine which one performs the best.

Experiment 11: Predicting Customer Churn in a Telecommunications Company using Decision Tree and Random Forest

Aim:

The aim of this experiment is to develop and evaluate Decision Tree & Random Forest models for predicting customer churn in a telecommunications company. The objective is to investigate the effectiveness of these models in predicting the customer churn in a Telecom Company.

Objectives:

- Tune hyperparameters in decision tree models (criterion(gini impurity, entropy), max_depth etc).
- Understand the various hyperparameters used in Random Forest (eg: n_estimators etc)
- Utilize techniques like pruning to prevent overfitting by limiting the tree's depth and complexity.
- Understand and interpret the similarities and differences between Random Forest and Decision Tree.

Process:

In this simulation, the student will be training and comparing the differences between Decision Tree and Random Forest and also interpreting the impurity art each node.

Simulation:

- Acquire a dataset containing information about telecommunications customers, including historical churn data.
- Preprocess the data by handling missing values, scaling numerical features, and encoding categorical features.
- Split the dataset into a training set and a testing set.
- Tune hyperparameters for the Decision Tree model, such as the maximum depth of the tree, minimum samples required for a split, and minimum samples required for a leaf.

- Configure hyperparameters for the Random Forest, including the number of trees (estimators), feature selection strategies, and tree depth.
- Train the Decision Tree and Random Forest models on the training dataset
- Assess the performance of both the Decision Tree and Random Forest models on the testing dataset using various evaluation metrics.
- Interpret the output, including Gini impurity or entropy at each node.

Experiment 12: Clustering wines using K-Means

Aim:

The aim of this experiment is to apply an Unsupervised machine learning algorithm - K-Means clustering to a dataset of wine characteristics and group wines into clusters based on their similarities.

Objectives:

- Explore the data and determine the optimal number of clusters for K-Means.
- Apply K-Means clustering to segment the wines into clusters.
- Visualize the clusters and assess the quality of the clustering.

Process:

In the simulation, the student will perform EDA on wine dataset and create visualizations aftering choosing the optimal 'k'.

Simulation:

- Gather a dataset containing attributes of wines, such as alcohol content, acidity, hue etc.
- Preprocess the data by handling any missing values, scaling the features, and ensuring data quality.
- Perform EDA and identify relevant features for clustering.
- Apply the K-Means algorithm to cluster the wines into the determined number of clusters.
- Initialize the centroids and iterate until convergence.
- Create visualizations to represent the clusters.
- Evaluate the homogeneity and separation of clusters.
- Analyze the characteristics of wine within each cluster.

Experiment 13: Model Validation and Performance Evaluation

Aim:

To introduce students to robust techniques for validating machine learning models and evaluating their performance across different scenarios.

Objectives:

- Understand the importance of proper model validation in machine learning
- Learn various cross-validation techniques and when to apply them
- Gain practical experience in implementing different performance metrics
- Understand how to interpret model performance results and make informed decisions

Process:

In this simulation, students will work with multiple datasets and model types to apply various validation techniques and performance metrics.

Simulation:

- Discuss the limitations of simple train-test splits and explain the concept of overfitting and how proper validation helps detect it.
- Implement k-fold cross-validation and explore variations like stratified k-fold and leave-one-out cross-validation.
- Train multiple models (e.g., logistic regression, decision tree, SVM) on a dataset and apply cross-validation to each model and compare their performance.
- Implement Grid Search and Random Search for hyperparameter optimization techniques using scikit-learn's GridSearchCV and RandomizedSearchCV.
- Use cross-validation results to select the best performing model and introduce basic ensemble methods (e.g., voting classifier) and validate their performance.

Experiment 15: Building a Handwritten digit classifier using Neural Networks

Aim:

Understand the basic theory, working and training process behind a Neural Networks

Objectives:

- Learn the basics of neural networks and its theory such as constructing a feed forward neural network, activation functions, back-propagation and gradient descent.
- Understand how to read the data, preprocess it and train a neural network.

Process:

In this simulation, the students will build a simple neural network in tensorflow.

Simulation:

- Download the MNIST Handwritten Digit classifier Dataset.
- Load the dataset and preprocess it in a way that we give it as input to the Neural Network.
- Split the dataset into training, testing and validation sets.
- Build a Feed-forward neural network using Tensorflow library.
- Train the neural network on the training set and evaluate it on the validation set.
- Visualize the model input and prediction from the test set.

Experiment 16: Data Ethics and Bias in Machine Learning

Aim:

To introduce students to the ethical considerations in data science and machine learning, with a focus on identifying and mitigating bias in datasets and models.

Objectives:

- Understand the importance of ethics in data science
- Learn to identify different types of bias in datasets and machine learning models
- Explore techniques to mitigate bias and promote fairness in machine learning
- Discuss the broader societal implications of AI and data-driven decision making

Process:

In this simulation, students will work with datasets that contain potential biases and learn how to identify and address these issues.

Simulation:

- Discuss key ethical principles in data science (e.g., privacy, fairness, transparency) and present case studies of ethical issues in real-world AI applications
- Load a dataset with known biases (e.g., a hiring dataset with gender bias) and use data visualization and statistical techniques to identify potential biases
- Discuss different types of bias (e.g., sampling bias, measurement bias, algorithmic bias)
- Implement preprocessing techniques to reduce bias (e.g., resampling, reweighting)
- Explore fairness-aware machine learning algorithms and Demonstrate the use of fairness metrics (e.g., demographic parity, equal opportunity)
- Present students with ethical dilemmas in data science scenarios and Discuss trade-offs between model performance and fairness, Hence, Developing guidelines for responsible AI development
- Explore the societal implications of biased AI systems and Discuss the role of data scientists in promoting ethical AI practices.

4. Learning Component

These experiments provide valuable hands-on experience in essential data science techniques such as supervised and unsupervised learning, using different packages for Data handling and visualizations, different algorithms for regression, classification and clustering, enabling learners to apply their knowledge to a variety of problem statements. After completing these experiments, the learner will be well versed in different types of approaches in ML and training models for different tasks.

5. Syllabi of Introductory Data Science and ML lab at various universities

1. IITK - CS771A Introduction To Machine Learning
https://cse.iitk.ac.in/pages/CS771.html
2. BITS - CSF320 Foundation of Data Science
https://nitinvinayak.github.io/public/pdfs/CS_F320_2266.pdf
3. VIT - CBS3006 Machine Learning
https://vit.ac.in/sites/default/files/scope/B.Tech_CSE_BS_2021_2022.pdf
4. IITG - CS 361 Machine Learning
https://www.iitg.ac.in/acad/CourseStructure/Btech2018/CS/CS361.html

6. Target Group

UG (3rd Year)

7. Mapping with AICTE course

Course name and Code - Machine Learning (PCC CS-603)

8. Student Feedback and Learning

- How will you collect feedback and use them?
    - ❖ Through online feedback forms with questions pertaining to the experiments, we will be using the collected data to improve upon the experiments. There will also be quizzes along with the experiments which will allow us to objectively know if the experiments were successful in imparting the necessary knowledge.

- What is the actual learning component provided by the Virtual Lab?
    - ❖ To gain practical, hands-on experience by conducting various data science experiments.
    - ❖ To develop problem-solving skills and the ability to interpret the results of data analysis.

- After the Virtual Lab experience, would the student be able to perform the experiment in the real lab?
    - ❖ The Virtual Lab will provide valuable skills and knowledge, with additional guidance and mentoring the student will be able to perform the experiment in the real lab tackling real world problems in a way that would benefit an Indian undergrad student.

9. Required Components for the Lab

1. Step-by-step procedure – Will be part of the description of each lab experiment.

2. Online manual – will be provided as part of the lab and can be accessed through the lab URL.

3. Pre-test I – A set of 5 objective questions to check whether the student has the prerequisite knowledge needed to attempt the lab. Students are expected to score 100% for this pretest to be able to proceed to the simulation. In case the student is unable to score this, the corresponding concepts will be displayed for student review before their next attempt at the pre-test.

4. Pre-test II - A set of 5 objective questions to gauge the students' understanding of the concepts which will be shown in the lab. The student is not expected to score anything here and this is just used in the final feedback process.

5. Simulator – For each experiment an online Python interpreter is designed for students to explore. Based on the experiment, the simulator may either visualize the algorithm/architecture or expect student input/configuration to assist in better understanding of the topic.
6. Post-test – A set of 5 questions are provided in each topic to assess student understanding. An 80% result is required to pass this assessment. A 100% score displays a congratulatory message to the student. An 80% result also shows the errors made by the student alongside the correct answer and corresponding review material. A score less than 80% directs the student to review the material/lab once again.
7. Related resources – will be provided as textual/ web data.