Expt. No. 1	Data Preprocessing with Python Load datasets, handle missing values and perform feature scaling.
Date :	

Purpose:

This experiment aims to provide students with hands-on experience in essential data preprocessing techniques using Python. By working with real-world datasets, students will learn how to prepare data for machine learning models effectively. This preparation includes:

- **Loading Datasets:** Understanding how to import data from different sources and formats into Python.
- **Handling Missing Values:** Learning strategies to manage and mitigate the impact of missing data on model performance.
- **Feature Scaling:** Applying scaling techniques to ensure features are in a comparable range, which is crucial for many machine learning algorithms.

Objectives:

1. **Loading Datasets:**

   Demonstrate how to load various datasets into Python using pandas, including CSV, Excel, and possibly other formats.

2. **Handling Missing Values:**

   Teach students how to identify and address missing values in a dataset to ensure data integrity and model accuracy.

3. **Performing Feature Scaling:**

   Enable students to apply feature scaling techniques to normalize and standardize data, preparing it for machine learning algorithms.

Equipment's /Components / Software:

S.No	Name
1	Collaboratory

Datasets:

A dataset is a structured collection of data, often presented in tabular form, where each row represents an individual record or observation, and each column represents a feature or attribute of the data.

Components of a Dataset

1. **Observations (Rows):**
   - Each row in a dataset typically represents a single instance or record. For example, in a customer dataset, each row might represent a different customer.
2. **Features (Columns):**
   - Each column represents a particular attribute or characteristic of the observations. For instance, in a dataset of houses, columns might include features like Price, Size, Location, etc.
3. **Values:**
   - The cells in the dataset contain the actual data values. For example, a cell might contain the price of a house or the age of a customer.

Steps:

1. Importance of data preprocessing in machine learning
2. Loading Datasets
3. Handling Missing Values
4. Performing Feature Scaling
5. Conclusion

Experimental Procedure:

1. Gmail login
2. Open Collaboratory
3. Download. CSV file and one image file

4. preprocessing on CSV file and do training ready dataset

Methods:

1. **Loading:** Import data from CSV files.

   df = pd.read_csv('file.csv')

2. **Exploring:** Understand the structure and contents of the data.

Display the first few rows of the DataFrame

print(df.head())

Get a summary of the DataFrame

print(df.info())

Describe statistical details of the dataset

   print(df.describe())

3. **Handling Missing Values:** Address gaps with imputation or removal.

Drop rows with missing values

df = df.dropna()

Drop columns with missing values

   df = df.dropna(axis=1)

Student Activity	Load another dataset and perform preprocessing techniques	**1ADPC304_3,1ADPC304_ 4**

Other links :

https://www.datacamp.com/courses/understanding-machine-learning

https://www.geeksforgeeks.org/online-courses-for-machine-learning

https://www.coursera.org/learn/machine-learning

Viva Questions:

Q. No	Question	CO	BL
1.	What is Dataset	1ADPC303_1	K1
2.	What is Data Preprocessing	1ADPC303_2	K1
3.	How to load/fetch a dataset	1ADPC303_1	K1
4.	What is colab ,jupyter	1ADPC303_1	K1
5.	What are types of datasets	1ADPC303_2	K2
6.	What is image	1ADPC303_1	K1
7.	Why we need preprocessing	1ADPC303_2	K2
8.	What is scaling	1ADPC303_1	K2

Expt. No. 2	Implement Linear Regression build a linear regression model, evaluation performance
Date :	

Purpose:

Implementing a linear regression model is a fundamental exercise in machine learning and data analysis. Here's a brief outline of the purpose of this experiment, followed by a basic implementation and performance evaluation steps.

Purpose of the Experiment

- **Understanding Linear Relationships**:
  - Linear regression helps in understanding the relationship between the dependent variable (target) and one or more independent variables (features).
- **Prediction**:
  - Once the model is trained, it can be used to make predictions about the target variable based on new input features.
- **Evaluation Metrics**:
  - Evaluating the performance of the linear regression model helps to understand how well the model generalizes to unseen data and the quality of the predictions.
- **Model Improvement**:
  - By analyzing the performance metrics, you can identify if the model is under fitting or over fitting, and apply techniques to improve it.

Objectives:

- To build a linear regression model that predicts a continuous target variable based on one or more features, and to evaluate its performance. Using historical data to train the model and estimate the parameters (coefficients) that define the linear relationship.

Equipment's /Components / Software:

S.No	Name
1	Collaboratory

Steps:

1. Import Libraries and Load Data,like numpy, pandas, scikit-learn, and possibly matplotlib for visualization.
2. Prepare the Data:

   Features and Target: Separate the dataset into features (X) and the target variable (y).

   Split Data: Divide the data into training and testing sets to evaluate the model's performance
3. Build and Train the Model

   Initialize: Create an instance of the LinearRegression model.

   Train: Fit the model on the training data.
4. Make Predictions

   Predict: Use the trained model to make predictions on the test set.
5. Evaluate Model Performance

   Metrics: Compute performance metrics to evaluate the model's accuracy and fit.

Experimental Procedure:

1. Gmail login
2. Open Collaboratory
3 Download. CSV file and one image file
4 preprocessing on CSV file and do training ready dataset
5 Call Liner regression
6 Test the target

7Evaluatethemodelperformance

Methods:

Evaluation methods for linear regression machine learning model:

1. Mean squared error
2. R2.score

Student Activity: Find out real time problem, collect dataset and apply linear regression on problem

Viva Questions:

Q. No	Question	CO	BL
1.	What is regression problem	1ADPC303_1	K1
2.	Define some real time example of regression where you apply the regression algorithm	1ADPC303_1	K1
3.	How to load/fetch a dataset	1ADPC303_2	K1
4.	What is colab ,jupyter	1ADPC303_1	K1
5.	What is R2 score	1ADPC303_2	K2
6.	What is Random state variable	1ADPC303_1	K1
7.	What are the evaluation technics for machine learning model	1ADPC303_3	K2
8.	What is linear regression	1ADPC303_1	K2
9	Explain linear regression equation	1ADPC303_5	K3

Student Activity	Load another dataset and perform Implement Linear Regression build a linear regression model, evaluation performance	**1ADPC304_3,1ADPC304_ 4**

Other links :

https://www.datacamp.com/courses/understanding-machine-learning

https://www.geeksforgeeks.org/online-courses-for-machine-learning

https://www.coursera.org/learn/machine-learning

Expt. No. 3	Classification with Logistic Regression and KNN Implement logistic regression and KNN, compare their performance
Date:	

Purpose:

This experiment aims to determine which classification algorithm—Logistic Regression or KNN—performs better in terms of accuracy and efficiency on a given dataset. By understanding their strengths and weaknesses, we aim to provide insights into their suitability for different classification tasks.

Objectives:

The primary objective of this experiment is to evaluate and compare the performance of Logistic Regression and K-Nearest Neighbors (KNN) in solving classification problems. This comparison will be based on performance metrics such as accuracy, precision, recall, F1-score, and computational efficiency.

Equipment/Components / Software:

S.No	Name
1	Collaboratory

Steps:

3. Gmail login
4. Open Collaboratory
3 Download. CSV file
5 preprocessing on CSV file and do training ready dataset
5 Call Liner regression
6 Test the target

7Evaluatethemodelperformance

Experimental Procedure:

1. Dataset Selection

- Choose a dataset suitable for classification. Ensure it has a mix of features and is well-suited for comparison.

- Example datasets: Iris dataset, Wine dataset, or any other dataset with labeled categorical outcomes.

2. Preprocessing

- Data Cleaning: Handle missing values, remove outliers, and address inconsistencies.
- Feature Scaling: Normalize or standardize features, especially important for KNN, as it is sensitive to the scale of data.
- Splitting Data: Divide the dataset into training and testing sets (e.g., 80% training and 20% testing).

3. Model Implementation

- Logistic Regression:
    - Implement the logistic regression model using a library such as scikit-learn in Python.
    - Fit the model on the training data.
    - Tune hyperparameters if necessary (e.g., regularization strength).
- K-Nearest Neighbors (KNN):
    - Implement the KNN model using scikit-learn.
    - Choose an appropriate value for k (number of neighbors), possibly through cross-validation.
    - Fit the model on the training data.

4. Model Evaluation

- Performance Metrics:
    - Accuracy: Calculate the ratio of correctly predicted instances to the total instances.
    - Precision: Measure the accuracy of positive predictions.
    - Recall: Measure the ability to find all positive instances.
    - F1-score: The harmonic mean of precision and recall.
    - Confusion Matrix: Analyze the true positives, true negatives, false positives, and false negatives.
- Computational Efficiency:
    - Training Time: Measure the time taken to train each model.
    - Prediction Time: Measure the time taken to make predictions on the test set.

5. Comparison

- Compare the performance metrics of Logistic Regression and KNN.
- Discuss the results based on accuracy, precision, recall, F1-score, and computational efficiency.
- Analyze which model performed better overall and why.

6. Discussion

- Interpret the results in the context of the dataset and the specific problem.
- Discuss the strengths and weaknesses of each model.
- Provide recommendations on which model might be preferred for similar classification problems.

Methods:

- Logistic Regression:
  - Use logistic function to model the probability of class membership.
  - Optimize using gradient descent or other optimization techniques.
- K-Nearest Neighbors (KNN):
  - Classify an instance based on the majority class among its k-nearest neighbors.
  - Experiment with different values of k and distance metrics (e.g., Euclidean, Manhattan).

Viva Questions:

Q. No	Question	CO	BL
1.	How will you evaluate the performance of the Logistic Regression model	1ADPC303_3	K3
2.	What performance metrics will you use to assess Logistic Regression (e.g., accuracy, precision, recall, F1-score)?	1ADPC303_3	K3
3.	How will you implement the K-Nearest Neighbors model	1ADPC303_4	K4
4.	How will you choose the value of k	1ADPC303_1	K2
5.	What distance metric will you use for KNN	1ADPC303_2	K3
6.	What are the performance metrics for KNN on the test data	1ADPC303_3	K2
	What are the training and prediction times for both models	1ADPC303_3	

7.			K2
8.	Which model performed better based on the evaluation metrics? Why?	1ADPC303_1	K2

Student Activity	Load another dataset and perform Classification with Logistic Regression and KNN Implement logistic regression and KNN, compare their performance	**1ADPC304_3,1ADPC304_ 4**

Other links:

https://www.datacamp.com/courses/understanding-machine-learning

https://www.geeksforgeeks.org/online-courses-for-machine-learning

https://www.coursera.org/learn/machine-learning

Expt. No. 4	Detecting Spam mails using Support Vector Machines
Date:	

Purpose:

To detect spam emails using Support Vector Machines (SVM) in Python, and to evaluate the model's performance on a labeled email dataset.

Objectives:

1. To develop a machine learning model using SVM for binary classification of emails into "spam" and "non-spam" categories.
2. To preprocess email data for feature extraction.
3. To train and evaluate the SVM model on the preprocessed data.
4. To assess the model's performance using accuracy, precision, recall, and F1 score

Equipment/Components / Software:

S.No	Name
1	Collaboratory

Steps

1. Gmail login
2. Open Collaboratory
3 Download. CSV file named Spam.CSV
4. Call SVM
5 Test the target

Experimental Procedure:

1. **Data Collection:**

- Obtain a labeled dataset of emails (e.g., the Enron email dataset or any other publicly available dataset).
- Each email should be labeled as "spam" or "ham" (non-spam).

2. **Model Training:**

- **Support Vector Machine:** Use the SVM algorithm to train the model on the training data.
- **Parameter Tuning:** Experiment with different kernels (linear, RBF, etc.) and hyper parameters (C, gamma) to find the best model.

3. **Model Evaluation:**

- **Predict on Test Data:** Use the trained model to classify the emails in the test set.
- **Performance Metrics:** Evaluate the model using accuracy, precision, recall, and F1 score.
- **Confusion Matrix:** Visualize the confusion matrix to understand the classification performance.

4. **Model Optimization (Optional):**

- **Cross-Validation:** Use cross-validation techniques to ensure the model's robustness.
- **Feature Selection:** Apply feature selection methods to improve model efficiency.

Methods:

Support Vector Machines (SVM):

- A supervised learning model that classifies data by finding the optimal hyper plane that separates different classes. Various kernels like linear, polynomial, and RBF can be used depending on the data characteristics

Q. No	Question	CO	BL
1.	What is the main objective of this experiment	1ADPC303_1	K1
2.	What does SVM stand for	1ADPC303_1	K1
3.	Explain how SVM is used for email classification	1ADPC303_3	K2
4.	Describe the difference between spam and non-spam emails in the context of this experiment.	1ADPC303_3	K2
5.	What is a hyperplane in the context of SVM	1ADPC303_3	K3
6.	Explain the basic working principle of an SVM.	1ADPC303_1	K3

7.	How does SVM handle non-linearly separable data	1ADPC303_4	K3
8.	Difference between a linear SVM and a non-linear SVM.	1ADPC303_1	K4

Student Activity	Load another dataset and perform Detecting Spam mails using Support Vector Machines	**1ADPC304_3,1ADPC304_ 4**

Other links:

https://www.datacamp.com/courses/understanding-machine-learning

https://www.geeksforgeeks.org/online-courses-for-machine-learning

https://www.coursera.org/learn/machine-learning

https://www.ibm.com/topics/support-vector

https://www.sciencedirect.com/topics/computer-science/support-vector-machine

Expt. No. 5	Model Evaluation with Cross –Validation implement cross-validation techniques to evaluate models
Date:	

Purpose:

To create a synthetic dataset, apply a linear regression model, and evaluate its performance using cross-validation techniques of linear regression for predicting a continuous target variable based on one or more features.

Objectives:

1. To generate a synthetic dataset that follows a linear relationship between the feature(s) and the target variable.
2. To train a linear regression model on the generated dataset.
3. To evaluate the model's performance using the Mean Squared Error (MSE) metric.
4. To assess the reliability and generalization of the model using cross-validation techniques.

Equipment/Components / Software:

S.No	Name
1	Collaboratory

Steps:

1. Gmail login
2. Open Collaboratory
3 Generate Dataset
4. Train the Linear Regression Model
5 Evaluate the Mode
6 Perform Cross-Validation

Experimental Procedure:

1.**Dataset Creation:**

- Generate a synthetic dataset where the target variable is linearly dependent on one or more features with added Gaussian noise.
- Convert the data into a suitable format (e.g., pandas DataFrame) for easy manipulation and analysis.

2. **Data Splitting:**

- Split the dataset into training and testing sets (e.g., 80% training, 20% testing) to allow for model evaluation on unseen data.

3. **Model Training:**

- Initialize and train a linear regression model on the training data.
- Use the training set to fit the linear regression model, estimating the relationship between the feature(s) and the target variable.

4. **Model Evaluation:**

- Predict the target variable for the test set using the trained model.
- Calculate the Mean Squared Error (MSE) between the predicted and actual target values on the test set to evaluate the model's performance.

5. **Cross-Validation:**

- Perform k-fold cross-validation (e.g., 5-fold) to assess the model's performance across different subsets of the data.
- Calculate the MSE for each fold and average the scores to obtain a more reliable performance metric.

Viva Questions:

1.	What is cross-validation in model evaluation	1ADPC303_1	K1
2.	What is the purpose of using cross-validation in model evaluation		K1
3.	Explain how k-fold cross-validation works		K2
4.	Why is cross-validation important in preventing overfitting		K2

5.	Define the terms: coefficient, intercept, and residual in the context of linear regression		K1
6.	What is the formula for the linear regression model		K1
7.	How does the Ordinary Least Squares (OLS) method work in linear regression		K3
8.	Define the terms: accuracy, precision, and recall.		K1
9.	What is a confusion matrix		K1
10	Explain the difference between precision and recal		K1
11	What is ROC-AUC		K1

Student Activity	Load another dataset and perform Model Evaluation with Cross –Validation implement cross-validation techniques to evaluate models	**1ADPC304_3,1ADPC304_ 4**

Other links:

https://www.datacamp.com/courses/understanding-machine-learning

https://www.geeksforgeeks.org/online-courses-for-machine-learning

https://www.coursera.org/learn/machine-learning

https://www.google.com/search?q=Machine+learning+evaluation+techniques+in+brief&rlz=1C1CHBD_enIN1122IN1122&oq=Machine+learning+evaluation+techniques+in+brief&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIHCAEQIRigATIHCAIQIRigATIHCAMQIRiPAjIHCAQQIRiPAtIBCTE0MjcwajBqN6gCCLACAQ&sourceid=chrome&ie=UTF-8

Expt. No. 6 Date:	**Dimensionality Reduction with PCA** Perform PCA on a dataset and interpret the results.

Purpose:

The primary goal of this experiment is to understand and apply **Principal Component Analysis (PCA)** as a dimensionality reduction technique to simplify a dataset while retaining most of the variance. By doing so, you can reduce the complexity of the data and make it more manageable for further machine learning tasks, such as classification or regression.

Objectives:

- Understand the concept of **Dimensionality Reduction** and the role of **PCA** in reducing the dimensions of a dataset.
- Apply PCA on a dataset and identify the **principal components** that capture the most significant variance.
- Visualize the data in reduced dimensions.
- Analyze and interpret how much variance is explained by each principal component.
- Evaluate the performance of PCA in preserving data characteristics after dimensionality reduction.

You can use one of the following datasets:

- **Iris dataset** (Common for demonstrating dimensionality reduction)
- **Wine dataset** (Another popular dataset with multiple features)
- **MNIST dataset** (If dealing with images and high-dimensional data)

Example: Let's consider the Iris dataset:

- Features: Sepal Length, Sepal Width, Petal Length, Petal Width
- Target: Species (Setosa, Versicolor, Virginica)

Methods:

Principal Component Analysis (PCA):

- PCA is a linear transformation technique that projects data onto a new set of axes (the principal components), ordered by the amount of variance they capture from the original dataset.

Steps:

1. **Standardize the Dataset:** PCA is sensitive to the scale of the features. Ensure that all features are standardized (mean = 0, variance = 1) to avoid features with larger scales dominating the principal components.
2. **Compute the Covariance Matrix:** This matrix represents the variance between each pair of features in the dataset.
3. **Compute the Eigenvalues and Eigenvectors:** The eigenvectors of the covariance matrix represent the principal components, and the corresponding eigenvalues indicate the amount of variance captured by each component.
4. **Select Principal Components:** Sort the eigenvalues in descending order, and select the top **k components** (where k is the number of dimensions you wish to reduce to).
5. **Transform the Data:** Project the original dataset onto the new principal component axes.

Interpretation of Results:

1. **Explained Variance:**
    - The explained_variance_ratio_ tells you how much variance each principal component explains. For example, if the first two components explain 95% of the variance, you can reduce the dataset to two dimensions without losing much information.
2. **Cumulative Variance:**
    - Use the cumulative sum of the explained variance to determine how many principal components are sufficient for your analysis. For example, if the first

two components explain 95% of the variance, you can safely reduce your dataset to two dimensions.

3. **Visualization:**
    - The 2D scatter plot helps you visualize how the data has been transformed. You should observe clusters corresponding to the different classes (species) of the Iris dataset. PCA might help reveal some structure or pattern in the data that was less obvious in the higher-dimensional space.
4. **Dimension Reduction Impact:**
    - If you have reduced the dimensions from 4 to 2 (for visualization), assess whether the patterns in the data are still meaningful and if the reduction captured the main variability.

Conclusion:

- PCA helps in simplifying datasets by reducing the number of features while retaining most of the data's variability.
- By visualizing the results, we can see that the principal components can often help in identifying patterns or structures in the data, even after dimensionality reduction.
- In this case, the PCA applied to the Iris dataset captures the majority of the variance with just two components, and this is reflected in the visualization where the species are still distinguishable.

Viva Questions:

Sr.No	Question	CO	Knowledge Level (K)
1	What is Dimensionality Reduction? Why is it needed in Machine Learning?	1ADPC303_1	K2
2	What is Principal Component Analysis (PCA)? Explain its core concept.	1ADPC303_1	K2
3	Describe the steps involved in performing PCA on	1ADPC303_2	K3

	a dataset.		
4	Perform PCA on a given dataset using scikit-learn and interpret the explained variance.	1ADPC303_2	K3
5	How do you determine the optimal number of principal components to retain in PCA?	1ADPC303_3	K4
6	Compare the performance of a machine learning model before and after applying PCA.	1ADPC303_3	K4
7	Apply PCA to reduce the dimensionality of a real-world dataset and evaluate the model's accuracy post-PCA.	1ADPC303_4	K6
8	How can PCA be used to solve the curse of dimensionality problem in high-dimensional data?	1ADPC303_4	K6
9	Discuss the limitations of PCA. How would you decide when PCA is not suitable for a dataset?	1ADPC303_5	K4
10	How does PCA compare to other dimensionality reduction techniques like t-SNE or LDA?	1ADPC303_5	K4

Student Activity	Load another dataset and perform **Dimensionality Reduction with PCA** Perform PCA on a dataset and interpret the results.	**1ADPC304_3,1ADPC304_ 4**

Other links:

https://www.google.com/search?q=pca+in+machine+learning&rlz=1C1YTUH_enIN1059IN1059&oq=PCA+&aqs=chrome.1.69i57j0i512j0i433i512j0i512l7.3395j0j7&sourceid=chrome&ie=UTF-8

https://www.geeksforgeeks.org/principal-component-analysis-pca

https://www.datacamp.com/tutorial/principal-component-analysis-in-python

Expt. No. 7	Implement Image Recognition using **MLP**
Date:	

Purpose:

1. Implement a simple neural network (MLP) for image classification.
2. Demonstrate how MLP can be applied for recognizing and classifying images, particularly in small to medium-sized datasets.
3. Explore the performance of MLP in comparison to other deep learning models, such as CNNs, for image-based tasks.
4. Analyze the accuracy, loss, and efficiency of MLP on an image dataset.

Objectives

1. Understand the architecture of a Multilayer Perceptron (MLP) and its role in image recognition tasks.
2. Preprocess an image dataset (e.g., MNIST or CIFAR-10) for use with MLP models.
3. Train an MLP using backpropagation and gradient descent to classify images.
4. Evaluate the performance of the trained MLP model in terms of accuracy, precision, and recall.
5. Compare the results with other models such as CNN (optional extension).

Dataset

For this experiment, you can use any publicly available dataset. Two common choices for beginners are:

1. **MNIST Dataset**:
    - Consists of 70,000 grayscale images of handwritten digits (0–9), with each image size of 28x28 pixels.
    - Training set: 60,000 images
    - Test set: 10,000 images
    - Each image is labeled with the corresponding digit.

2. **CIFAR-10 Dataset**:
    - Contains 60,000 32x32 color images in 10 classes, with 6,000 images per class.
    - Training set: 50,000 images
    - Test set: 10,000 images
    - Classes include airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

Methods

1. **Preprocessing**:

- Normalization: Scale pixel values to the range [0, 1].
- Flattening: Convert each image into a 1D array (e.g., 28x28 pixels becomes a vector of length 784).
- One-Hot Encoding: Encode class labels as one-hot vectors (e.g., the digit "5" becomes [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]).

2. Model Architecture:

- Input Layer: Accepts the flattened image input (e.g., 784 nodes for MNIST).
- Hidden Layers: 1-2 hidden layers with fully connected neurons using ReLU activation.
- Output Layer: The number of neurons equals the number of classes (10 for MNIST, 10 for CIFAR-10), with softmax activation to predict probabilities for each class.

3. **Training**:

- Loss Function: Categorical Cross-Entropy for multi-class classification.
- Optimizer: Adam optimizer with a learning rate of 0.001.
- Metrics: Accuracy to monitor model performance.

4. **Evaluation**:

- Evaluate the model using the test set to compute accuracy, loss, and confusion matrix.

5. **Optional**:

- Hyperparameter tuning (e.g., number of hidden layers, learning rate).
- Visualization of training history (loss, accuracy over epochs)

Conclusion

1. Have built and trained an MLP for image recognition tasks.
2. Gained insight into the strengths and limitations of using MLPs for image classification compared to more complex models like CNNs.
3. Be able to analyze and interpret the performance of MLP on image datasets.

Q. No	Question	CO	BL
1	What are the basic concepts of image recognition and how is MLP used for this task in machine learning?	1ADPC303_1	K2
2	Explain how a Multi-Layer Perceptron works in image recognition tasks.	1ADPC303_1	K2
3	Implement an MLP model for recognizing handwritten digits using the MNIST dataset in Python with a library of your choice.	1ADPC303_4	k3
4	Write a Python program to preprocess image data before feeding it into an MLP for training	1ADPC303_2	K3
5	How would you evaluate the performance of an MLP model in recognizing images? Describe different metrics you would use.	1ADPC303_3	K4
6	Analyze the effect of increasing the number of hidden layers in an MLP on image recognition accuracy.	1ADPC303_3	K4
7	Develop an MLP-based solution for classifying images from a custom dataset and explain the process in detail.	1ADPC303_4	K6

8	Propose a strategy to improve the accuracy of your MLP model for image recognition in a real-world application.	**1ADPC303_4**	K6
9	Discuss how advanced techniques like dropout and batch normalization can be used in MLPs for image recognition tasks.	**1ADPC303_5**	**K4**
10	Explore how transfer learning can be applied in combination with MLP for real-time image recognition applications.	**1ADPC303_5**	**K4**

Student Activity	Load another dataset and perform Implement Image Recognition using **MLP**	**1ADPC303_3,1ADPC303_ 4**

Other links:

https://www.altexsoft.com/blog/image-recognition-neural-networks-use-cases

https://www.datacamp.com/tutorial/multilayer-perceptrons-in-machine-learning

https://www.geeksforgeeks.org/multi-layer-perceptron-learning-in-tensorflow

Expt. No. 08 Date:	**Association Rule Mining with Apriori** Generate association rules using the Apriori algorithm.

Purpose

The purpose of this experiment is to implement the Apriori algorithm for mining association rules within a dataset. Association Rule Mining (ARM) is a technique widely used in data mining to uncover interesting relations, patterns, or associations among data items in large datasets.

Objectives:

- **Understand the Apriori Algorithm**: Learn the working principles of the Apriori algorithm, including how it finds frequent itemsets and generates association rules.
- **Apply Association Rule Mining**: Use the Apriori algorithm to discover frequent itemsets in a dataset and interpret the generated association rules.
- **Optimize the Mining Process**: Experiment with support, confidence, and lift thresholds to filter and optimize the quality of association rules.
- **Explore Use Cases**: Analyze the usefulness of association rules in practical applications, such as market basket analysis, recommendation systems, or healthcare.

Dataset:

For this experiment, a dataset with categorical attributes is ideal. One commonly used dataset is the **Market Basket Analysis (MBA)** dataset, which contains records of transactions from a retail store or e-commerce site. Each transaction includes a unique transaction ID and items purchased in that transaction.

Alternatively, datasets from other fields, such as medical data or product usage logs, could also be utilized depending on the application. The dataset should include:

- **Transactions**: Unique IDs representing each transaction.

- **Items**: A list of categorical items or product identifiers included in each transaction.

For example, a subset of the dataset might look like this:

Transaction ID	Items
1	Bread, Milk
2	Bread, Diapers, Beer, Eggs
3	Milk, Diapers, Beer, Cola
4	Bread, Milk, Diapers, Beer
5	Bread, Milk, Diapers, Cola

Methods:

1. **Data Preprocessing**:
    - Load the dataset and clean it by removing duplicate transactions, handling missing data, and formatting items in a standard way.
    - Convert the transaction data into a suitable format, such as a transaction matrix, where each row represents a transaction, and columns represent items.
2. **Frequent Itemset Generation with Apriori Algorithm**:
    - Use the Apriori algorithm to scan the dataset and identify frequent itemsets based on a minimum support threshold. The algorithm iteratively generates candidate itemsets and eliminates non-frequent itemsets to reduce the search space.
    - Select an appropriate minimum support threshold that balances the number of generated itemsets and computational efficiency.
3. **Association Rule Generation**:
    - After finding frequent itemsets, generate association rules by calculating support, confidence, and lift for each potential rule.
    - Apply minimum confidence and lift thresholds to filter out weak or less meaningful rules.

4. **Evaluation**:
    - Evaluate the generated rules by analyzing their support, confidence, and lift values. This can help in determining the relevance and strength of associations.
    - Experiment with different threshold values to see how they affect the quantity and quality of rules.
5. **Analysis and Interpretation**:
    - Interpret the results by analyzing patterns within the rules. For instance, in a retail dataset, rules like {Bread} => {Milk} may indicate that customers who buy bread often also buy milk.
    - Consider applications of discovered patterns, such as cross-selling, inventory optimization, and targeted marketing.

Conclusion: Summarize the findings and discuss the effectiveness of the Apriori algorithm for ARM. Suggest potential improvements or next steps, such as using other ARM algorithms (e.g., FP-Growth) or exploring additional datasets for further experimentation

Viva Questions :

Q.No	Question	CO	K Level
1	What is Association Rule Mining, and how does it apply to machine learning?	**1ADPC303_1**	**K2**
2	Define the Apriori Algorithm and its primary purpose in association rule mining.	**1ADPC303_1**	**K2**
3	Explain key terms used in Association Rule Mining, such as support, confidence, and lift.	**1ADPC303_1**	**K2**
4	How do association rules differ from classification rules in machine learning?	**1ADPC303_1**	**K2**
5	Describe the steps involved in implementing the Apriori Algorithm.	**1ADPC303_2**	**K3**

6	Which Python libraries can be used to implement association rule mining with the Apriori Algorithm?	**1ADPC303_2**	**K3**
7	Demonstrate how to set minimum support and minimum confidence thresholds in an Apriori implementation.	**1ADPC303_2**	**K3**
8	Show how you would preprocess a dataset to be used with the Apriori Algorithm.	**1ADPC303_2**	**K3**
9	What are the key factors that affect the performance of the Apriori Algorithm?	**1ADPC303_3**	**K4**
10	How do support, confidence, and lift influence the selection of association rules?	**1ADPC303_3**	**K4**
110	Compare the performance and computational cost of Apriori with other association rule mining algorithms.	**1ADPC303_3**	**K4**
12	Explain how the results of association rule mining can be evaluated and interpreted.	**1ADPC303_3**	**K4**
13	How would you use the Apriori Algorithm to solve a real-world problem, such as market basket analysis?	**1ADPC303_4**	**K6**
14	Describe how association rule mining with Apriori can be applied to recommend products in an e-commerce site.	**1ADPC303_4**	**K6**
15	Identify challenges you might face when applying the Apriori Algorithm in a high-dimensional dataset.	**1ADPC303_4**	**K6**
16	Discuss any limitations of the Apriori Algorithm and current advancements that address these issues.	**1ADPC303_5**	**K4**
17	Explain how Apriori is used in emerging fields such as healthcare or social media analysis.	**1ADPC303_5**	**K4**
18	What are some recent improvements or modifications to the Apriori Algorithm in the field of machine learning?	**1ADPC303_5**	**K4**

Student Activity	Load another dataset and perform **Association Rule Mining with Apriori** Generate association rules using the Apriori algorithm.	**1ADPC303_3,1ADPC303_ 4**

Other links:

https://www.youtube.com/watch?v=guVvtZ7ZClw

https://medium.com/image-processing-with-python/apriori-algorithm-in-associate-rule-mining-dc9404caffd1

Expt. No. 09	**Implement Recommender System** for Movie Recommendations
Date:	

Purpose:

The purpose of this experiment is to develop a recommender system for movie recommendations using machine learning techniques. Recommender systems are crucial in personalized content delivery, enhancing user experience by suggesting items (such as movies) based on user preferences, past behaviors, or similarity with other users.

Objectives:

1. **Understand Recommendation System Types**: Explore and understand the different types of recommendation systems, including collaborative filtering, content-based filtering, and hybrid methods.
2. **Build a Collaborative Filtering Model**: Use collaborative filtering (both user-based and item-based) to make recommendations based on user-movie interactions.
3. **Develop a Content-Based Recommendation Model**: Implement content-based filtering to suggest movies based on movie attributes (e.g., genre, actors, director) similar to the user's past preferences.
4. **Evaluate the Model's Performance**: Assess the effectiveness of the recommendation system using metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), or precision and recall, depending on the evaluation approach.
5. **Deploy a Hybrid Model (Optional)**: Experiment with a hybrid recommender model combining collaborative and content-based approaches to improve recommendation quality.

Dataset

For this experiment, a popular dataset is the **MovieLens** dataset, which includes ratings, movie details (e.g., genres, title, release year), and user profiles. This dataset contains:

- **User IDs**: Unique identifiers for each user.
- **Movie IDs**: Unique identifiers for each movie.

- **Ratings**: Ratings given by users to movies (usually on a 1–5 scale).
- **Movie Attributes**: Details about each movie, including title, genre, and release year.

Results:

- Present the recommended movies for selected users and analyze the recommendations based on user preferences.
- Provide evaluation metrics and discuss how they indicate the performance of the recommender system.

Conclusion:

- Summarize the experiment's findings, noting any challenges faced (e.g., data sparsity) and potential improvements.
- Suggest future work such as incorporating implicit feedback, using deep learning for embeddings, or building more complex hybrid systems.

Viva Questions :

Q. No	Question	CO	K Level
1	What is a recommender system, and why is it important in machine learning applications?	1ADPC303_1	K2
2	Describe the types of recommendation techniques used in recommender systems.	1ADPC303_1	K2
3	Explain the difference between collaborative filtering and content-based filtering.	1ADPC303_1	K2
4	Which Python libraries are typically used for implementing a recommender system, and why?	1ADPC303_2	K3
5	How would you use the scikit-learn library to perform collaborative filtering for movie recommendations?	1ADPC303_2	K3
6	How do you preprocess data for a movie recommendation system?	1ADPC303_2	K3
7	What metrics would you use to evaluate the effectiveness of a movie recommendation system?	1ADPC303_3	K4
8	How would you interpret the performance metrics of a recommendation model?	1ADPC303_3	K4

9	Describe how you would implement a hybrid recommender system combining collaborative and content-based methods.	1ADPC303_4	K6
10	What are some of the real-world challenges when implementing a movie recommendation system?	1ADPC303_4	K6
11	Discuss some of the recent trends in recommender systems, such as deep learning-based approaches.	1ADPC303_5	K4
12	How does matrix factorization work in collaborative filtering, and what are its advantages?	1ADPC303_5	K4
13	In what ways can recommendation systems be improved to handle new user and item issues (cold start problem)?	1ADPC303_5	K4
14	Can you explain how content-based filtering can help in recommending movies based on genre or director similarity?	1ADPC303_1	K2
15	What is the role of similarity measures (e.g., cosine similarity) in collaborative filtering?	1ADPC303_2	K3
16	How would you use the pandas library to manage and filter a dataset of movie ratings?	1ADPC303_2	K3
17	How can you ensure that your recommendation model generalizes well to new data?	1ADPC303_3	K4
18	Explain how you would handle missing data or sparse data in the user-item interaction matrix.	1ADPC303_3	K4
19	Describe how you would approach hyperparameter tuning for a recommendation model.	1ADPC303_4	K6
20	What are some advanced techniques like transfer learning or neural networks used in recommender systems?	1ADPC303_5	K4

Other links:

Student Activity	Load another dataset and perform **Implement Recommender System** for Movie Recommendations	**1ADPC303_3,1ADPC303_ 4**

https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie

https://www.upgrad.com/blog/create-your-own-movie-recommendation-system-using-python

Expt. No. 10	**Hyperparameter Tuning** Use grid search and random search for model optimization.
Date:	

Purpose:

The purpose of this experiment is to understand and apply hyperparameter tuning techniques, specifically grid search and random search, for optimizing machine learning models. Hyperparameters are parameters that govern the behavior of a model but are not learned from the training data.

Proper tuning of these hyperparameters can significantly enhance model performance. This experiment focuses on implementing both grid search and random search to identify the optimal set of hyperparameters for a chosen model.

Objectives

1. **Understand Hyperparameter Tuning**: Learn about the importance of hyperparameter tuning in model optimization and how it differs from parameter tuning.
2. **Implement Grid Search for Tuning**: Apply grid search to exhaustively search for the best combination of hyperparameters within a specified range.
3. **Apply Random Search for Tuning**: Use random search to randomly sample hyperparameter combinations within a range, offering a faster alternative to grid search.
4. **Compare Performance**: Evaluate the effectiveness of grid search and random search and compare their performance in terms of accuracy, computation time, and efficiency.
5. **Analyze and Select the Best Model**: Select the model with the optimal hyperparameters and analyze how these parameters affect model performance.

Methods:

Implement Grid Search:

- Define a grid of hyperparameter values. Grid search will exhaustively search through each combination of these hyperparameters to find the best model.

- For each combination, train the model and evaluate its performance using cross-validation.
- Select the hyperparameter set that yields the highest performance.

Implement Random Search:

- Specify ranges for each hyperparameter, allowing the algorithm to randomly sample a subset of combinations.
- Use cross-validation to evaluate performance for each randomly selected combination.
- Random search continues for a predefined number of iterations or until a set time limit is reached.
- Identify the combination of hyperparameters that yields the highest performance.

Results:

- Present the best hyperparameters found by grid search and random search, along with corresponding cross-validation scores.

- Provide a comparison table or plot for both methods, showing accuracy and computation time.
- Highlight the test set performance of the final model with optimized hyperparameters.

Conclusion:

- Summarize the experiment's findings, highlighting the importance of hyperparameter tuning for model optimization.
- Suggest future improvements, such as using Bayesian optimization or other advanced search methods for even more efficient tuning.
- Discuss the general applicability of hyperparameter tuning across different machine learning models and problem types

Viva Questions :

Q No.	Question	CO	K Level
1	What is the role of hyperparameters in machine learning models, and how does hyperparameter tuning impact model performance?	1ADPC303_1	K2
2	Can you explain the difference between grid search and random search in the context of hyperparameter optimization?	1ADPC303_1	K2
3	How would you decide which hyperparameters to tune in a machine learning algorithm like Random Forest or SVM?	1ADPC303_2	K3
4	What are the common metrics used to evaluate the performance of machine learning models, and how are they affected by hyperparameter tuning?	1ADPC303_3	K4
5	Can you explain the impact of choosing too many or too few hyperparameter values during grid search on the model's performance and computation?	1ADPC303_3	K4
6	In what scenarios would you prefer using random search over grid search for hyperparameter tuning?	1ADPC303_4	K6
7	How does the computational cost of grid search compare to random search, and how does this influence the selection of a hyperparameter tuning method?	1ADPC303_2	K3

8	How would you implement hyperparameter tuning in Python using GridSearchCV and RandomizedSearchCV?	1ADPC303_2	K3
9	Can you explain the concept of cross-validation in hyperparameter tuning and its importance in preventing overfitting?	1ADPC303_3	K4
10	How can hyperparameter tuning improve a machine learning model's performance in a real-world application like fraud detection or recommendation?	1ADPC303_4	K6
11	How do hyperparameter optimization methods like grid search and random search contribute to the field of machine learning, especially in real-time applications?	1ADPC303_5	K4
12	Explain how you would handle hyperparameter tuning for deep learning models and how it differs from traditional machine learning algorithms.	1ADPC303_5	K4

Student Activity	Load another dataset and perform **Hyperparameter Tuning** Use grid search and random search for model optimization	**1ADPC303_3,1ADPC303_ 4**

Other links:

https://www.youtube.com/watch?v=G-fXV-o9QV8

https://machinelearningmastery.com/hyperparameter-optimization-with-random-search-and-grid-search/

Expt. No. 11	**Choose best machine learning algorithm** to implement online fraud detection
Date:	

To implement an **Online Fraud Detection** system using machine learning, you'll need to design an experiment with clear **purpose, objectives, dataset, and methods**. Here's a comprehensive guide:

1. Purpose:

The primary purpose of this experiment is to build and evaluate machine learning algorithms for detecting fraudulent transactions in online environments. The aim is to reduce false positives while maximizing the detection rate of fraudulent activities.

2. Objectives:

- To analyze and preprocess an online transaction dataset to identify important features related to fraud.
- To compare various machine learning algorithms and evaluate their performance in detecting fraudulent transactions.
- To select the best-performing algorithm based on evaluation metrics like accuracy, precision, recall, F1-score, and AUC-ROC.
- To implement real-time fraud detection for online platforms using the chosen machine learning algorithm.

3. Dataset:

You'll need a dataset that includes both fraudulent and non-fraudulent transactions, with relevant features like transaction amount, timestamp, user demographics, and transaction details.

- **Public Datasets for Online Fraud Detection**:
  - **Kaggle's Credit Card Fraud Detection Dataset**: A popular dataset with anonymized credit card transactions labeled as fraudulent or non-fraudulent.
  - **IEEE-CIS Fraud Detection Dataset**: Contains features related to transactions, including device and card details, fraud labels, and more.
  - **PaySim**: A simulated mobile money transfer dataset mimicking real financial systems.

Features:

  - Transaction ID, timestamp
  - Transaction amount

- Location (IP, country)
- Payment method (credit card, bank transfer)
- Merchant information
- User demographics (age, gender, account history)
- Flag indicating if the transaction was fraudulent (target variable)

4. Methods:

The process of selecting and implementing the best machine learning algorithm can be broken down into the following steps:

A. Data Preprocessing:

- **Handling missing values**: Identify and handle missing or inconsistent data.
- **Feature engineering**: Create new features from existing ones (e.g., transaction frequency, user behavior patterns).

- **Data scaling and normalization**: Normalize data to ensure algorithms perform optimally, especially for models like SVM.
- **Class imbalance handling**: Fraud data is typically highly imbalanced (more legitimate transactions than fraudulent ones). Techniques like SMOTE (Synthetic Minority Over-sampling Technique), under sampling, or weighted loss functions may be needed.

B. Algorithms:

To select the best algorithm, you can experiment with several classifiers, including:

1. **Logistic Regression**:
   - A simple, interpretable baseline model.
   - Works well with well-separated classes but may struggle with imbalanced data.
2. **Random Forest**:

- A powerful ensemble method based on decision trees.
- Works well with complex data patterns and is robust to overfitting.

3. **Gradient Boosting Machines (GBM)**:
   - Includes methods like XGBoost, LightGBM, and CatBoost.
   - Often outperform many other algorithms on tabular data like fraud detection due to their ability to handle complex interactions between features.
4. **Support Vector Machine (SVM)**:
   - Effective for small- to medium-sized datasets with clear margins between classes.
   - Can be used with different kernels (linear, RBF) but may need balancing techniques.
5. **Neural Networks (Deep Learning)**:
   - Useful for handling complex patterns in large-scale datasets.
   - Variants like LSTM or GRU can be applied to sequential transaction data for temporal fraud detection.
6. **Anomaly Detection Algorithms**:
   - **Isolation Forest**: Designed for anomaly detection, focusing on outliers like fraud.
   - **Autoencoders**: Unsupervised neural networks that can identify fraud as anomalies.

C. Evaluation Metrics:

Since fraud detection is a binary classification problem with imbalanced data, standard accuracy is insufficient. Instead, you should focus on:

- **Precision**: The ratio of true fraud cases among the cases predicted as fraud.
- **Recall (Sensitivity)**: The ratio of true fraud cases detected by the model.
- **F1-score**: Harmonic mean of precision and recall.
- **AUC-ROC (Area Under the Curve - Receiver Operating Characteristic)**: Measures how well the model separates fraud from non-fraud.
- **Confusion matrix**: To better understand the number of false positives and false negatives.

D. Cross-Validation:

Use **cross-validation techniques** (e.g., k-fold cross-validation) to ensure the model generalizes well to unseen data. Stratified k-fold cross-validation is preferred to maintain the proportion of fraud cases in each fold.

E. Hyperparameter Tuning:

Perform hyperparameter tuning using techniques like **Grid Search** or **Random Search** to find the best **combination of hyperparameters for each algorithm.**

F. Deployment:

Once the best algorithm is selected, it can be deployed in a real-time environment using tools like:

- **Flask or FastAPI**: To expose the model as an API for real-time fraud detection.
- **Stream processing platforms**: Such as **Apache Kafka** for real-time transaction monitoring.
- **Model Monitoring**: Set up monitoring systems to track the performance of the model in real-world use cases and retrain it periodically to adapt to changing patterns of fraud.

5. Conclusion:

In this experiment, you'll explore different machine learning techniques to select the best model for online fraud detection. The final model should be scalable and able to detect fraud in real-time with high precision and recall to minimize both false positives and false negatives.

Viva Questions :

Q. No	Question	CO	BL
1	What are the key characteristics of online fraud detection systems?	1ADPC303_1	K1
2	Explain the difference between supervised and unsupervised machine learning approaches in the context of fraud detection.	1ADPC303_1	K2
3	What are some popular algorithms used for detecting	1ADPC303_2	K2

	online fraud?		
4	Demonstrate how to implement an online fraud detection model using Random Forest in Python.	1ADPC303_2	K3
5	How do you use scikit-learn to implement a fraud detection model with a Decision Tree?	1ADPC303_3	K3
6	Compare the performance of Random Forest and Logistic Regression in online fraud detection.	1ADPC303_3	K4
7	Evaluate the precision, recall, and F1 score of an online fraud detection model using cross-validation.	1ADPC303_4	K4
8	How would you design an ensemble-based system to improve fraud detection accuracy?	1ADPC303_4	K6
9	Discuss the impact of using advanced techniques like Neural Networks for online fraud detection.	1ADPC303_5	K4

Other links:

https://www.sciencedirect.com/science/article/pii/S2772662223000036#:~:text=Random%20forest%20is%20the%20most,favour%20the%20majority%20class%20solely.

https://sqream.com/blog/fraud-detection-machine-learning

https://trustdecision.com/resources/blog/5-new-machine-learning-algorithms-for-fraud-detection

Student Activity	Load another dataset and perform **Choose best machine learning algorithm** to	**1ADPC304_3,1ADPC304_ 4**

	implement online fraud detection	


Viva Questions


Viva Questions