# L. J Institute of Engineering and Technology
# List of Practical for ML and AML

| | |
|---|---|
| 1. | Prepare decision tree classifier in python using sklearn library for data set diabetes.csv - (pregnant, glucose, bp, skin, insulin, bmi, pedigree, age, label). Use all features except label as independent variable. Use complete dataset for training and limit the depth of tree upto 3 levels and plot using tree.plotTree()method. |
| 2. | Using the diabetes dataset, implement a Decision Tree Classifier and determine the importance of each feature in predicting diabetes. Generate different plots to justify your model. |
| 3. | Implement a Decision Tree Classifier and compare its performance with a SVM Classifier on the iris dataset. Display the accuracy of both models. |
| 4. | Build decision tree classifier for iris data set. One with maximum leaf nodes up to 8 and another one with minimum sample per leaf as 5. Compare accuracy of both models. |
| 5. | Implement a Decision Tree Classifier and compare its performance with a Logistic Classifier on the diabetes dataset. Develop Python code to display the accuracy of both models. |
| 6. | Develop a random dataset using 3 columns and 350 rows. Generate the data frame and apply decision tree classifier to train the model. Compare the accuracy of the model with any 2 other models you can apply on the generated dataset. Generate plot of comparing the accuracy for different models. |
| 7. | Using the diabetes dataset, implement a Decision Tree Classifier and plot ccp_alpha of built tree against any one DT parameter (node count or maximum depth). |
| 8. | Write a Python script that implements the Elbow Method to determine the optimal number of clusters for a generated dataset. (Generate dataset of your choice with 14 points) |
| 9. | Generate a random dataset of urban areas with features such as population density, average income, and amenities, (min. 100 data). Apply K-medoids clustering to identify neighbourhoods with similar characteristics. Visualize the clusters on a map. |
| 10. | Generate a random dataset containing customer information (e.g., age, spending score)(100 rows), apply K-medoids clustering to segment customers into distinct groups. Visualize the clusters and summarize the characteristics of each segment |
| 11. | You are working with an e-commerce company that wants to segment its customers based on their purchasing behaviour. Generate random dataset that includes features like total spending, frequency of purchases, and average basket size. Develop a Python program to perform K-means clustering. Visualize the clusters and summarize the characteristics of each segment. |
| 12. | Use make_blobs to generate 300 samples. Analyse the data and apply KMeans clustering and generate the graph for better representation. Show what happens if you change n_clusters count from 2 to 7(2,3,4,5,6,7)? |
| 13. | A grocery store wants to analyze customer purchase patterns to improve product placements. Generate random transaction data (products purchased together), apply K-means clustering to group similar transactions. |
| 14. | Generate random dataset with 35 points. Apply KMeans clustering and generate the plot with each cluster points with different colours. What is the impact of varying cluster count? Justify your understanding with suitable graphs. |
| 15. | Develop a Python program to implement DBSCAN using the sklearn library. Use a 2D dataset (ex. make_moons) for clustering. |
| 16. | Generate a random dataset with 5 columns and use 3rd and 4th column as data to implement DBSCAN. Consider eps=5 and minPts=5. |
| 17. | Use DBSCAN to cluster customers based on their purchasing behaviour (ex. frequency of purchases, average transaction amount)(Generate random dataset with min. 100 customers). Develop a program that analyses the clusters to identify different customer segments for targeted marketing |

| | |
|---|---|
| 18. | Generate a data frame containing two features(Spending,features) for 100 data. Apply DBSCAN and do parametric analysis for eps value and show the impact on the performance of model via graph. |
| 19. | Generate a random dataset with 4 columns and use 1st and 3rd column as data to implement DBSCAN. What if eps is changed from 1 to 5? How it will affect the performance of the model? What is the impact of minPts on the performance of the model? Justify it with graph by changing minPts. |
| 20. | Generate random dataset for mall_customer dataset with features like Customer, Genre, Age, Annual Imcome, Spending Score. Consider minimum 200 random samples for analysis. Apply DBSCAN as well as agglomerative clustering and generate the dendrogram. Plot points in each cluster for DBSCAN with different colours. |
| 21. | Write a Python code to perform k-fold cross-validation on the dataset using a Support Vector Machine (SVM) classifier. Use the following steps:<br><br>• Import necessary libraries.<br>• Prepare the dataset synthetically using make_classification.<br>• Preprocess the data if necessary (e.g., scaling).<br>• Implement k-fold cross-validation with 10 folds.<br>• Train the SVM classifier on the training set and evaluate it on the test set for each fold.<br>• Print the accuracy for each fold and the average accuracy across all folds.<br><br>Dataset Description:<br>• The dataset consists of 100 samples with 20 features.<br>The target variable is binary (0 or 1). |
| 22. | Write a Python code to perform bootstrap sampling on a dataset using a Random Forest Classifier. The dataset consists of features and labels as specified below.<br><br>**Dataset Description:**<br><br>• The dataset consists of 150 samples with 10 features.<br>• The target variable is categorical with three classes (0, 1, 2).<br>**Instructions:**<br><br>• Import the necessary libraries.<br>• Load or create a synthetic dataset with the specified characteristics.<br>• Implement bootstrap sampling to create multiple samples from the original dataset.<br>• Train a Random Forest Classifier on each bootstrap sample.<br>• Evaluate the model on the original dataset and print the accuracy for each bootstrap sample.<br>• Calculate and print the average accuracy across all bootstrap samples. |
| 23. | Write a Python code to perform Leave-One-Out Cross-Validation (LOOCV) on a dataset using a Decision Tree Classifier. The dataset consists of features and labels as specified below.<br><br>**Dataset Description:**<br><br>• The dataset consists of 100 samples with 5 features.<br>• The target variable is binary (0 or 1).<br><br>**Instructions:** |

| | |
|---|---|
| | • Import the necessary libraries.<br>• Load or create a synthetic dataset with the specified characteristics.<br>• Implement Leave-One-Out Cross-Validation (LOOCV).<br>• Train a Decision Tree Classifier on each training set and evaluate it on the left-out sample.<br>Print the accuracy for each iteration and the average accuracy across all iterations. |
| 24. | Write a Python code to perform Leave-One-Out Cross-Validation (LOOCV) on a dataset using a Decision Tree Classifier. The dataset consists of features and labels as specified below.<br><br>**Dataset Description:**<br><br>• The dataset consists of 150 samples with 10 features.<br>• The target variable is categorical with three classes (0, 1, 2).<br>• Introduce some noise to the dataset to make the classification task more<br><br>**Instructions:**<br><br>• Import the necessary libraries.<br>• Load or create a synthetic dataset with the specified characteristics (make_classification), including noise.<br>• Implement Leave-One-Out Cross-Validation (LOOCV).<br>• Train a Decision Tree Classifier on each training set and evaluate it on the left-out sample.<br>• Calculate and print the accuracy for each iteration, as well as the confusion matrix for the final model.<br>• Calculate and print the average accuracy across all iterations.<br>Print confusion matrix and classificarion report. |
| 25. | Write a Python code to perform Stratified K-Fold Cross-Validation on a dataset using a Random Forest Classifier. The dataset consists of features and labels as specified below.<br><br>**Dataset Description:**<br><br>• The dataset consists of 200 samples with 15 features.<br>• The target variable is categorical with four classes (0, 1, 2, 3).<br><br>**Instructions:**<br><br>• Import the necessary libraries.<br>• Load or create a synthetic dataset with the specified characteristics, ensuring class imbalance.<br>• Implement Stratified K-Fold Cross-Validation with 5 folds.<br>• Train a Random Forest Classifier on each training set and evaluate it on the validation set.<br>• Calculate and print the accuracy for each fold, as well as the average accuracy across all folds.<br>Generate and visualize the classification report for the final model, including precision, recall, and F1-score for each class. |
| 26. | Write a Python code to perform a Grid Search with Cross-Validation on the Iris dataset using a Support Vector Machine (SVM) classifier. The goal is to find the best hyperparameters for the SVM model. |

| | |
|---|---|
| | **Dataset Description:**<br><ul><li>The Iris dataset consists of 150 samples with 4 features.</li><li>The target variable is categorical with three classes (Setosa, Versicolor, Virginica).</li></ul>**Instructions:**<br><ul><li>Import the necessary libraries.</li><li>Load the Iris dataset from sklearn.datasets.</li><li>Split the dataset into features and labels.</li><li>Implement a Grid Search with 5-fold Cross-Validation to find the best hyperparameters for the SVM classifier. Consider the following hyperparameters: C: [0.1, 1, 10, 100] kernel: ['linear', 'rbf'] gamma: ['scale', 'auto']</li><li>Print the best hyperparameters found by the Grid Search.</li><li>Evaluate the best model on the entire dataset and print the accuracy.</li></ul>Generate and visualize the confusion matrix for the best model. |
| 27. | Write a Python program to apply Principal Component Analysis (PCA) on the Iris dataset and reduce its dimensionality to 2 dimensions. You should use sklearn's PCA implementation. After reducing the dimensions, visualize the transformed data in a 2D scatter plot, coloring the points based on their respective classes.<br><br>**Dataset Description:**<br><ul><li>The Iris dataset consists of 150 samples with 4 features.</li><li>The target variable is categorical with three classes (Setosa, Versicolor, Virginica).</li></ul>**Instructions:**<br><ul><li>Import the necessary libraries.</li><li>Load the Iris dataset from sklearn.datasets.</li><li>Separate the features and the target variable.</li><li>Standardize the features (mean = 0, variance = 1).</li><li>Apply PCA to reduce the dimensionality of the dataset to 2 dimensions.</li></ul>Visualize the transformed data in a 2D scatter plot, using different colors for each class. |
| 28. | Write a Python program that demonstrates the curse of dimensionality using the k-Nearest Neighbors (k-NN) classifier with k=5. The program should create a synthetic dataset with varying dimensions and evaluate the performance of the k-NN classifier as the number of dimensions increases. Specifically, you should:<br><ul><li>Generate a synthetic dataset with a fixed number of samples (e.g., 1000) and varying dimensions (e.g., 2, 5, 10, 20, 50, 100).</li><li>Use the k-NN classifier to classify the data.</li><li>Measure and print the accuracy of the classifier for each dimensionality.</li></ul>Visualize the accuracy as a function of dimensionality. |
| 29. | You are using a Support Vector Machine (SVM) classifier for a dataset that has 3 features and 500 data points. |

| | |
|---|---|
| | If you increase the dimensionality from 3 features to 12 features, explain how the volume of the feature space changes. What are the potential challenges that arise from this increase in dimensionality?<br><br>**Instructions:**<br>• Provide detailed answers to both questions, incorporating concepts from machine learning and geometry.<br>• Use diagrams or examples where appropriate to illustrate your points.<br>Discuss the implications of high-dimensional spaces on model performance, including overfitting, computational complexity, and the curse of dimensionality. |
| 30. | Create a synthetic dataset using make_blobs with the following specifications:<br><br>• The dataset should have 4 features and 4 centers (clusters).<br>• Initialize the PCA object with n_components=3 to reduce the dataset to 3 dimensions.<br>• Use Matplotlib to plot the original 4D dataset in a 3D scatter plot and the PCA-reduced 3D dataset in another 3D scatter plot side by side.<br>**Instructions:**<br><br>• Import the necessary libraries (numpy, matplotlib, and sklearn).<br>• Generate the synthetic dataset using make_blobs.<br>• Apply PCA to reduce the dataset to 3 dimensions.<br>• Create a figure with two subplots:<br>• The first subplot should display the original 3D representation of the dataset (you can use any three of the four features for this plot).<br>• The second subplot should display the PCA-reduced 3D dataset.<br>• Ensure that the points in both plots are color-coded based on their respective cluster labels.<br>Add appropriate titles and labels to the plots for clarity. |
| 31. | Use the Breast Cancer dataset to train a random forest classifier. The model will be trained to accurately classify tumors as benign or malignant. The dataset is imbalanced, with a significant majority of benign samples. Evaluate the model's performance using accuracy, precision, recall, and F1-score. generate complete problem statement |
| 32. | Use the Wine dataset, to develop a k-nearest neighbors (k-NN) classifier to classify different types of wine based on their chemical properties. You will evaluate the model's performance using various metrics, including accuracy, precision, recall, and F1-score.<br>• Load the Wine dataset from a CSV file<br>• checking for missing values, understanding the distribution of classes, and visualizing the features.<br>• Normalize or standardize the features if necessary to improve the performance of the k-NN classifier<br>• split the dataset into training and testing sets (e.g., 80% training and 20% testing)<br>• Train the classifier on the training dataset.<br>• Use the trained model to make predictions on the test dataset.<br>• Calculate and print the following performance metrics:<br>    o Accuracy<br>    o Precision (for each class)<br>    o Recall (for each class)<br>    o F1-score (for each class) |

| | |
|---|---|
| 33. | Apply Principal Component Analysis (PCA) to the Iris dataset to reduce its dimensionality from 4 to 2 dimensions. You will visualize the PCA-reduced dataset in a 2D scatter plot using Matplotlib.<br>**instructions:**<br><br>Load the Iris dataset<br>Briefly explore the dataset to understand its structure. Print the shape of the dataset and the first few rows to get an overview of the features and target variable.<br>Initialize the PCA object to reduce the dataset to 2 dimensions.<br>Fit the PCA model to the Iris dataset and transform the dataset<br>Use Matplotlib to create a scatter plot of the PCA-reduced dataset.<br>Color the points based on their respective species (Setosa, Versicolor, Virginica) to visualize how well the PCA has separated the different classes.<br>Add appropriate titles and labels to the axes of the plot.<br>Include a legend to indicate which colors correspond to which species. |
| 34. | Apply Linear Discriminant Analysis (LDA) to the Iris dataset to reduce its dimensionality from 4 to 2 dimensions. You will visualize the LDA-reduced dataset in a 2D scatter plot using Matplotlib.<br>**instructions:**<br><br>Load the Iris dataset<br>Briefly explore the dataset to understand its structure. Print the shape of the dataset and the first few rows to get an overview of the features and target variable.<br>Initialize the PCA object to reduce the dataset to 2 dimensions.<br>Fit the LDA model to the Iris dataset and transform the dataset<br>Use Matplotlib to create a scatter plot of the LDA-reduced dataset.<br>Color the points based on their respective species (Setosa, Versicolor, Virginica) to visualize how well the LDA has separated the different classes.<br>Add appropriate titles and labels to the axes of the plot.<br>Include a legend to indicate which colors correspond to which species. |
| 35. | Apply Linear Discriminant Analysis (LDA) to the Wine dataset to reduce its dimensionality and classify the types of wine based on their chemical properties. You will visualize the LDA-reduced dataset in a 2D scatter plot and evaluate the classification performance.<br><br>Dataset: You will use the Wine dataset, which consists of 178 samples of wine, each described by 13 features representing different chemical properties. The target variable indicates the type of wine, which can take on one of three classes (1, 2, or 3). |
| 36. | apply Linear Discriminant Analysis (LDA) to the Iris dataset using the Scikit-learn library. You will preprocess the data using label encoding, perform LDA to reduce the dimensionality of the dataset, and visualize the results in a 2D scatter plot. |
| 37. | Objective: The goal of this practical exam is to apply Linear Discriminant Analysis (LDA) to the Breast Cancer dataset using the Scikit-learn library. You will reduce the dimensionality of the dataset from multiple features to 1 dimension and visualize the results using Matplotlib.<br>• load the Breast Cancer dataset<br>• initialize the LDA to reduce the dataset to 1 dimension.<br>• Fit the LDA model to the Breast Cancer dataset and transform the dataset to obtain the LDA-reduced representation.<br>Visualization |

| | |
|---|---|
| 38. | Dataset: You will use the Iris dataset, which consists of 150 samples of iris flowers, each described by 4 features (sepal length, sepal width, petal length, and petal width). The target variable indicates the species of the iris flower (Setosa, Versicolor, or Virginica). apply both Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) to the Iris dataset. You will reduce the dimensionality of the dataset using both techniques and visualize the results in 2D scatter plots. You will then compare the effectiveness of LDA and PCA in terms of class separability. |
| 39. | develop a linear regression model to predict house prices based on various features. You will use a dataset that contains information about houses, including features such as the number of bedrooms, square footage, and location. You will evaluate the model's performance using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared. Show performance of same model when PCA reduced data set is used. |
| 40. | apply Principal Component Analysis (PCA) to a house price prediction dataset to reduce its dimensionality and then use a regression model to predict house prices. You will evaluate the model's performance using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared ($R^2$). |
| 41. | Consider Social_Netwok_Ads.csv dataset - (UserID, Gender, Age, EstimatedSalary, Purchased). Use Age and EstimatedSalary as input features and Purchased as target feature. Split test data set 30% of complete dataset. Build two models of support vector classifier in python using sklearn library, one for linear and another for RBF kernel with C and gamma parameters set. Predict test labels and print test accuracy. |
| 42. | Develop a Support Vector Classifier to predict whether a tumor is malignant or benign based on 30 features computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. The dataset contains 569 samples with binary labels indicating tumor type. (breast_cancer dataset of sklearn) Load the Dataset from sklearn.Split the dataset into training and testing sets.Train the SVC Model using the training data. Visualize the Results. |
| 43. | Write a Python program to implement SVM classification for breast cancerprediction with the following requirements: Data preprocessing using StandardScaler use kernel parameters for tuning. |
| 44. | The breast cancer dataset is available in the sklearn.datasets module and can be loaded using load_breast_cancer(). implement a Naive Bayes Classifier on the Breast Cancer Dataset using python's sklearn library. Assume that all features of datasets are continuous variables and must be used for building model. Perform following task. load data set, Split data for training and testing, Build a model and Predict labels of test data. |
| 45. | Implement a Gaussian Naive Bayes classifier to predict whether a patient has diabetes based on various health metrics of PIMA.csv dataset. The dataset consists of information from 768 female Pima Indians aged 21 and older, initially gathered by the National Institute of Diabetes and Digestive and Kidney Diseases. Target variable: Diabetes (binary, 0 or 1) Attributes: Pregnancies, OGTT (Oral Glucose Tolerance Test), Blood pressure, Skin thickness, Insulin, BMI (Body Mass Index), Age, Pedigree diabetes function.Load the Dataset from a CSV file. Provide summary statistics for the dataset. Split the data into training and testing sets (80% train, 20% test). Instantiate a Gaussian Naive Bayes model and fit it on the training data. Predict diabetes status for the test set. Discuss any biases in the dataset and how they may affect model performance |
| 46. | Build a Naive Bayes classifier to analyze the sentiment of movie reviews (positive or negative). The IMDB_Dataset.csv typically contains two main columns: Review: The text of the movie review. Sentiment: A label indicating the sentiment of the review, usually categorized as 'positive' or 'negative'. Load the dataset and inspect the structure. Clean the text data by removing punctuation, converting to lowercase, and tokenizing the reviews.Use CountVectorizer or TfidfVectorizer to |

| | |
|---|---|
| | convert the text data into a numerical format suitable for model training. Split the dataset and Create a Multinomial Naive Bayes model and fit it on the training data. Predict the sentiment of the test reviews. |
| 47. | Use sklearn.datasets.fetch_20newsgroups() dataset (all sets) to classify newsgroup documents into their respective categories using a Multinomial Naive Bayes classifier. Load the dataset and explore the categories available. Preprocess the text data using CountVectorizer to convert text documents into a matrix of token counts. Split the dataset and Create a Multinomial Naive Bayes model. Fit the model on the training data and print accuracy. |
| 48. | Imagine a telecommunications company that wants to predict whether a customer will churn (leave the service) based on various features such as age, account length, and monthly charges. The company has historical data on customers, including whether they churned or not.<br>Age: (in years), Account_Length: (in months), Monthly_Charges: (in dollars)<br>Churn: Target variable (1 if the customer churned, 0 otherwise)<br>'Age': [25, 34, 45, 29, 50, 38, 42, 35, 48, 55],<br>'Account_Length': [12, 24, 36, 18, 48, 30, 42, 24, 36, 60],<br>'Monthly_Charges': [70, 90, 80, 100, 60, 80, 90, 70, 80, 100],<br>'Churn': [0, 1, 0, 1, 0, 1, 0, 1, 0, 1]<br><br>build a logistic regression model using python that can predict the probability of a customer churning by spliting 80% of given data for training. Predict targets on trained model of test data. Print accuracy |
| 49. | build a logistic regression model that can predict whether a customer is likely to churn based on the features in file named customer_churn.csv.<br>customer_id: Unique customer ID, age: Customer age,<br>gender: Customer gender (male/female)<br>account_length: Length of the customer's account (in months)<br>international_plan: Whether the customer has an international plan (yes/no)<br>voice_mail_plan: Whether the customer has a voice mail plan (yes/no)<br>number_vmail_messages: Number of voice mail messages<br>total_day_calls: Total day calls<br>total_night_calls: Total night calls<br>total_intl_calls: Total international calls<br><br>churn: Whether the customer churned (yes/no) |
| 50. | Use iris dataset for creating a binary classification problem that predicts whether a flower is of the species "Iris-Virginica" or not. Dataset Features: sepal_length, sepal_width, petal_length, petal_width (in cm) species: Species of the iris flower (Iris-setosa, Iris-versicolor, Iris-virginica). Make Predictions and Print Probabilities with Alter Threshold to 0.6. |
| 51. | Consider Salary.csv , with years_of_experience and salary. Write a python code for fitting best fit simple linear regression with independent variable years_of_experience and dependent variable salary. Use complete dataset to train model. Plot :fitted model along with trained data points , residual plot. |

| | |
|---|---|
| 52. | Consider a dataset HousePrices.csv with columns square_feet and price. Write a Python code to fit a polynomial regression model with square_feet as the independent variable and price as the dependent variable. Use the complete dataset to train the model. Plot the fitted polynomial model along with the trained data points and create a residual plot. |
| 53. | Consider a dataset CarPrices.csv with columns age, mileage, and price. Write a Python code to fit a multiple linear regression model with age and mileage as independent variables and price as the dependent variable. Use the complete dataset to train the model. Plot the actual prices against the predicted prices and create a residual plot. |
| 54. | Consider a dataset HousePrices.csv with features such as size (in square feet) and a target column price (in dollars). Write a Python code to implement Linear Regression using Gradient Descent to predict price based on size. Use the complete dataset to train the model. Plot the regression line along with the training data points. |
| 55. | Consider a dataset CarPrices.csv with features such as horsepower, age, and a target column price. Write a Python code to fit a linear regression model to predict price based on the features and evaluate the model using MSE, $R^2$, and MAE. |
| 56. | Consider a dataset HouseData.csv with features such as num_rooms, square_feet, and location, and a target column house_price. Write a Python code to fit a linear regression model to predict house_price based on the features and evaluate the model using Mean Squared Error (MSE), $R^2$, and Mean Absolute Error (MAE). |
| 57. | Consider a dataset EmployeeData.csv with features such as years_of_experience, education_level, and a target column salary. Write a Python code to fit a linear regression model to predict salary based on the features and evaluate the model using Mean Squared Error (MSE), R-squared ($R^2$), and Mean Absolute Error (MAE) |
| 58. | Consider a dataset HousingData.csv with columns num_rooms, area, age, and a target column price. Write a Python code to fit a Lasso regression model using num_rooms, area, and age as independent variables to predict price. Use the complete dataset to train the model. Plot the coefficients of the features and create a residual plot. |
| 59. | Consider a dataset Diabetes.csv with various medical attributes and a target column diabetes_progression. Write a Python code to fit a Ridge regression model using all available features to predict diabetes_progression. Use the complete dataset to train the model. Plot the coefficients of the features and create a residual plot. |
| 60. | Consider a dataset WineQuality.csv with various chemical properties of wine and a target column quality. Write a Python code to fit a Lasso regression model using all available features to predict quality. Use the complete dataset to train the model. Plot the coefficients of the features and create a residual plot. |
| 61. | Consider a dataset HealthData.csv with features such as age, bmi, blood_pressure, and a target column health_score. Write a Python code to fit a Lasso regression model using all available features to predict health_score. Use the complete dataset to train the model. Plot the coefficients of the features and create a residual plot. |
| 62. | Write a python snippet to demonstrate use of PCA on 100 samples with 3 features generated randomly. |
| 63. | Write a python snippet to demonstrate use of PCA on dataset of your choice. Also print explained variance of all principle components. |
| 64. | Create synthetic Dataset using make_blobs with 3 features and 3 centers initialize the PCA object with n_components=2 to reduce the dataset to 2 dimensions. use matplotlib to plot the original 3D dataset and the PCA-reduced 2D dataset side by side. |

| | |
|---|---|
| 65. | Use load_iris from sklearn.datasets to load the Iris dataset. initialize the PCA object with n_components=2 to reduce the dataset to 2. use matplotlib to plot the PCA-reduced dataset in 2D. |
| 66. | Use load_wine from sklearn.datasets to load the wine dataset. initialize the LDA object with n_components=2 to reduce the dataset to 2. use matplotlib to plot the LDA-reduced dataset in 2D represented by a different color. |
| 67. | Use breast cancer from sklearn.datasets to load the cancer dataset. initialize the LDA object with one component to reduce the dataset to 1D. use matplotlib to plot the LDA-reduced dataset. |
| 68. | Use load_digits() from sklearn.datasets to load the hand written numbers dataset. initialize the LDA object with two component to reduce the dataset. Use matplotlib to plot the LDA-reduced dataset. |
| 69. | Use load_iris from sklearn.datasets to load the Iris dataset. initialize the LDA object with n_components=2 to reduce the dataset to 2. use matplotlib to plot the LDA-reduced dataset in 2D. |
| 70. | Implement an SVC model to classify iris flowers into three species (Setosa, Versicolor, and Virginica) based on their sepal and petal dimensions. The dataset contains 150 samples with four features: sepal length, sepal width, petal length, and petal width. Load the Dataset from sklearn.Split the dataset into training and testing sets.Train the SVC Model using the training data. Visualize the Results. |