ML Record for Task-1 to Task-4

Task-1: Demonstrate the Data Pre-Processing Techniques by Taking Real Datasets.

AIM: To implement the Data Pre-Processing Techniques by Taking Real Datasets.

Description:

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. Real-world or raw data usually has inconsistent format, human errors, and can also be incomplete. Data preprocessing resolves such issues and makes datasets more complete and efficient to perform data analysis. The main objective of data preprocessing step is to ensure that input data is accurate, consistent, reliable, missing values are handled and check the quality of data before applying any Machine Learning Model.

Procedure:

Step1: Importing necessary Libraries as Pandas, Numpy, Scikit-learn, matplotlib etc..

Step2: Loading the dataset as sale dataset using read_csv function.

Step3: Exploratory Data Analysis(EDA)

- Identify structure and data types of each column using describe() function.
- Check for missing values using isnull().
- Visualize distributions and relationships between variables using plots.

Step4: Handling Missing values

- Removing Rows: If the percentage of missing values is small using dropna() function
- Imputation**:** Filling missing values with mean, median, or mode using fillna() function.

Step5: Managing Categorical Features

- One-Hot Encoding: Converts categorical variables into binary columns using get_dummies() function.
- Label Encoding**:** Assigns a unique integer to each category using LabelEncoder() function

Step6: Feature Scaling

- Min-Max Scaling: Scales features to a range between 0 and 1 using MinMaxScaler() function.
- Standardisation: Centers features around zero with a standard deviation of one using StandardScaler() function.

Program:

Output:

Task-2(a): Implement dimensionality Reduction (PCA) technique.

AIM: To implement the dimensionality Reduction technique as PCA without using Scikit-learn and with using Scikit-learn

Description: Dimensionality reduction techniques can be used to filter only a limited number of significant features needed for training. Principal component analysis (PCA) is unsupervised learning algorithm, used for the dimensionality reduction that enables you to identify correlations and patterns in a data set so that it can be transformed into a data set of significantly lower dimension without loss of any important information. On finding a strong correlation between different variables, a final decision is made about reducing the dimensions of the data in such a way that the significant data is still retained.

Procedure:

Step1: Importing necessary Libraries as Pandas, Numpy, Scikit-learn, matplotlib etc..

Step2: Loading the dataset as diabetes dataset using read_csv function.

Step3: Standardization of the data: Standardization is carried out by subtracting each value in the data from the mean and dividing it by the overall deviation in the data set.

$$Z = \frac{Variable\ value - mean}{Standard\ deviation}$$

Step4: Computing the covariance matrix using Cov() function.

Step5: Calculating the eigenvectors and eigenvalues of covariance matrix using linalg.eig() function

Step6: Sort eigenvalues and corresponding eigenvectors in descending order.

Step7: Computing the Principal Components by Selecting the top n_components eigenvectors

Step7: Project the original data onto the new lower-dimensional space using dot product of standardized data with principal components.

Step8: Plot the first two principal components using scatter plot.

Program:

Output:

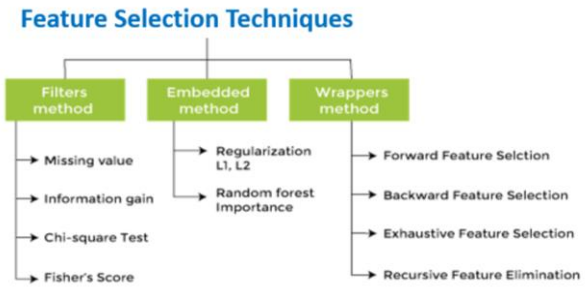
Task-2(b): Demonstrate Feature subset selection Techniques.

AIM: To implement Feature subset selection techniques as Filter, Wrapper and Embedded methods.

Description:

Feature selection is an important process in machine learning . It involves selecting a subset of relevant features from a larger set of available features. The primary objective of feature selection is to identify and retain the most informative and relevant features while discarding or ignoring the irrelevant or redundant ones. By doing so, it improves the performance of models by focusing on the most meaningful information and avoiding noise or unnecessary complexity.

Feature selection reduces the dimensionality of the data, making it easier for the model to learn and reducing the risk of overfitting. By reducing the number of features, feature selection can also help to reduce training time and computational costs. Different feature selection techniques, including filter, wrapper, and embedded methods, can be used depending on the type of data and the modeling approach.



Procedure:

Step1: Importing necessary Libraries as Pandas, Numpy, Scikit-learn, matplotlib etc..

Step2: Loading the dataset as diabetes dataset using read_csv function.

Step3: Separating features and target class using iloc function.

Step4: Filter Method: Perform Information gain using mutual_info_classif() function to select best features.

Step5: Wrapper Method: Perform Forward feature Selection using Sequential FeatureSelector() function to select best features.

Step6: Embedded Method: Perform Regularization technique using Lasso() function to select best features.

Program:

Output:

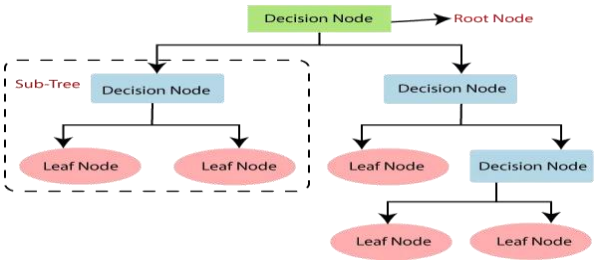
Task-3: Write a program to demonstrate the working of the Decision Tree-Based Id3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.

AIM: To implement the Decision Tree-Based Id3 algorithm on appropriate data set for building the decision tree and apply this knowledge to classify a new sample.

Description:

Decision Tree is a supervised learning technique that can be used for classification. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset.

ID3 (Iterative Dichotomiser 3) decision tree uses Entropy function and Information gain as metrics.



Procedure:

Step1: Importing necessary Libraries as Pandas, Numpy, Scikit-learn, matplotlib etc..

Step2: Loading the dataset as Iris.

Step3: Separating features and target class.

Step4: Split dataset into training (80%) and testing (20%) sets using train_test_split() function.

Step5: Train and fit Decision Tree Model using Entropy on training set using DecisionTreeClassifier() function.

Step6: Predict on the test set using predict() function.

Step6: Compute performance metrics using classification_report() function

Program:

Output:

Task-4: Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering a few test data sets.

AIM: To implement the Naïve Bayesian classifier for a sample training data set and compute the accuracy of the classifier, considering a few test data sets.

Description: Naïve Bayes algorithm is a supervised learning algorithm, based on conditional probability. Naïve Bayes is based on Bayes theorem and used for solving classification problems. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. It is mainly used in text classification that includes a high-dimensional training dataset.

Bayes' theorem is used to determine the probability of a hypothesis with prior knowledge.

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

- P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
- P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
- P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
- P(B) is Marginal Probability: Probability of Evidence.

Procedure:

Step1: Importing necessary Libraries as Pandas, Numpy, Scikit-learn, matplotlib etc..

Step2: Loading the dataset as Iris.

Step3: Separating features and target class.

Step4: Split dataset into training (80%) and testing (20%) sets using train_test_split() function.

Step5: Train and fit Naïve Bayes model on training set using GaussianNB() function.

Step6: Predict on the test set using predict() function.

Step6: Compute performance metrics using classification_report() function.

Program:

Output: