**SCRIPT EXPLANATION**

**SELECT\_K\_ALGORITHM:**

This Python script is a classification workflow for comparing different machine learning algorithms. It takes a dataset, preprocesses it, selects important features, trains various classifiers, evaluates their performance, and presents the results in a DataFrame.

Here’s a step-by-step explanation:

**1. Import Libraries**

The script imports essential libraries like:

* pandas for data manipulation.
* scikit-learn for preprocessing, feature selection, and machine learning models.
* matplotlib for visualization (though not used in this script).

**2. Functions Overview**

**a. selectkbest**

This function uses the Chi-Square test (chi2) to select the n most important features from the independent variables (indep\_X). The output is a reduced dataset with the selected features.

**b. split\_scalar**

* Splits the dataset into training and testing sets (X\_train, X\_test, y\_train, y\_test) using an 80-20 split.
* Scales the features using StandardScaler to standardize the data (mean = 0, variance = 1).

**c. cm\_prediction**

* Predicts the target values using a classifier.
* Calculates and returns:
  + **Confusion Matrix (cm)**
  + **Accuracy Score**
  + **Classification Report**

**d. Classification Algorithms**

The script defines functions for training and evaluating several classifiers:

1. **Logistic Regression** (logistic)
2. **Support Vector Machine - Linear Kernel** (svm\_linear)
3. **Support Vector Machine - Non-Linear Kernel** (svm\_NL)
4. **Naive Bayes** (Navie)
5. **K-Nearest Neighbors** (knn)
6. **Decision Tree** (Decision)
7. **Random Forest** (random)

Each function fits a model to the training data, predicts on the test set, and returns evaluation metrics.

**e. selectk\_Classification**

* Creates a summary DataFrame with accuracy scores for each algorithm.
* Rows represent feature selection methods (here, only ChiSquare is used).
* Columns represent different classifiers.

**3. Data Loading and Preprocessing**

* **Dataset Loading**: dataset1=pd.read\_csv("prep.csv") loads the dataset from a CSV file.
* **Encoding Categorical Variables**: pd.get\_dummies converts categorical variables into dummy/indicator variables for machine learning compatibility.
* **Splitting Features and Target**:
  + indep\_X: Independent variables (features).
  + dep\_Y: Dependent variable (target).

**4. Feature Selection**

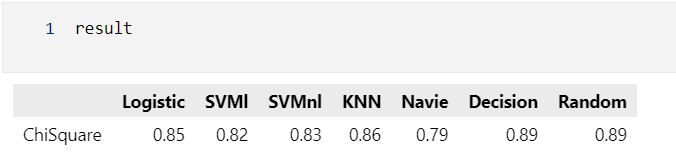
* selectkbest(indep\_X, dep\_Y, 5) selects the top 5 features using Chi-Square.
* The reduced dataset (kbest) is used in the subsequent steps.

**5. Model Training and Evaluation**

1. **Data Splitting**:
   * split\_scalar(kbest, dep\_Y) splits the dataset and scales the features.
2. **Classifier Training**:
   * Each classifier is trained using its respective function (e.g., logistic, svm\_linear).
   * The accuracy scores are stored in corresponding lists (e.g., acclog, accsvml).

**6. Results Compilation**

The results are compiled into a DataFrame using selectk\_Classification, with the following structure:



* Each cell shows the accuracy of a specific classifier.
* This provides a comparative analysis of classifier performance.

The final result DataFrame displays the accuracy scores of all classifiers using the Chi-Square-selected features.

**RECURSIVE\_FEATURE\_ELIMINATION:**

The code performs **feature selection and model evaluation** on a dataset, aiming to compare the accuracy of different machine learning models after Recursive Feature Elimination (RFE) has been applied to select the top n features. Here’s a step-by-step explanation:

**1. Dataset Preparation**

* The dataset prep.csv is loaded using pandas.read\_csv().
* Dummy encoding is applied to categorical variables with pd.get\_dummies(), and the classification\_yes column is treated as the dependent variable (dep\_Y).
* The independent features are stored in indep\_X.

**2. Recursive Feature Elimination (RFE)**

**Function: rfeFeature(indep\_X, dep\_Y, n)**

* Uses RFE with four estimators: Logistic Regression, SVM with a linear kernel, Random Forest, and Decision Tree.
* Each model iteratively eliminates less important features to select the top n features (here, n=3).
* The reduced feature sets are stored in rfelist.

**3. Train-Test Split and Feature Scaling**

**Function: split\_scalar(indep\_X, dep\_Y)**

* Splits the data into training and testing sets (75% train, 25% test) using train\_test\_split.
* Applies StandardScaler to normalize the features.

**4. Classification Models**

Each model follows this process:

* **Fit the Model**: Train the classifier on the training set.
* **Predict**: Use the trained model to make predictions on the test set.
* **Evaluate**: Calculate accuracy, confusion matrix, and classification report.

The models evaluated include:

* Logistic Regression
* Support Vector Machine (linear kernel)
* Support Vector Machine (non-linear kernel, RBF)
* K-Nearest Neighbors
* Naive Bayes
* Decision Tree
* Random Forest

**5. Model Performance Comparison**

**Function: rfe\_classification(acclog, accsvml, accsvmnl, accknn, accnav, accdes, accrf)**

* Takes accuracy values from all models for each RFE-transformed feature set.
* Rows represent the different RFE models (Logistic, SVM, Random Forest, Decision Tree).
* Columns represent the accuracy scores for each classifier applied to the selected features.

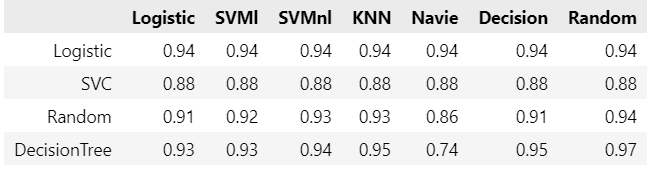
**6. Accuracy Calculation**

For each RFE-transformed feature set (rfelist):

* The code trains each classifier on the reduced feature set and appends the accuracy to corresponding lists (acclog, accsvml, etc.).

**7. Result**

The final result DataFrame contains accuracy scores for each model and feature selection combination.



**Purpose of the Code**

1. **Feature Selection**: Identifies the most important features using RFE.
2. **Model Evaluation**: Compares the performance of multiple classifiers using selected features.
3. **Decision Making**: Allows the user to choose the best combination of feature selection method and classifier for their dataset.