K BEST ALGORITHM ASSIGNMENT

Chi-Square

The Chi-Square (χ^2) test is a statistical test commonly used in **feature selection** for machine learning tasks, particularly in the context of classification problems. It helps to **identify the features** that are most relevant or important for **predicting the target variable.**

Confusion matrix

The main purpose of a confusion matrix in machine learning is to provide a detailed breakdown of the performance of a classification model. It helps evaluate the model's accuracy beyond just the overall classification accuracy

```
def selectkbest(indep_X,dep_Y,n):
        test = SelectKBest(score_func=chi2, k=n)
        fit1= test.fit(indep_X,dep_Y)
        selectk_features = fit1.transform(indep_X)
        return selectk_features
```

indep_X: The independent variables (features)

dep_Y: The dependent variable (target)

n: The number of features to select.

SelectKBest-->The function creates an instance of the SelectKBest class, which is a feature selection technique that selects the k best features based on a scoring function.

In this case, the chi2 (chi-square) scoring function is used.

fit1= test.fit(indep_X,dep_Y)--> get the values of k ex(k=4) i/p as 4 columns and o/p as 1.get into fit1.

selectk_features = fit1.transform(indep_X)--> fit1 of k value transform to indep_X.

return selectk_features--> Mainly for select the K feature

```
def split_scalar(indep_X,dep_Y):
        X_train, X_test, y_train, y_test = train_test_split(indep_X, dep_Y, test_size = 0.25, random_state = 0)
        sc = StandardScaler()
        X_train = sc.fit_transform(X_train)
        X_test = sc.transform(X_test)
        return X_train, X_test, y_train, y_test
```

split_scalar--> performs data splitting and standard scaling on the independent variables.

training and testing sets using **train_test_split from sklearn.model_selection**

It splits indep_X and dep_Y into training and testing sets (X_train, X_test, y_train, y_test) with a test size of 25% and a random state of 0 for reproducibility.

StandardScaler -->object sc from sklearn.preprocessing to standardize the features. It then applies fit_transform to standardize the training set X_train and transform to standardize the testing set X_test.

```python
def cm_prediction(classifier,X_test):
    y_pred = classifier.predict(X_test)

       # Making the Confusion Matrix
    from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(y_test, y_pred)

    from sklearn.metrics import accuracy_score
    from sklearn.metrics import classification_report

    Accuracy=accuracy_score(y_test, y_pred )

    report=classification_report(y_test, y_pred)
    return  classifier,Accuracy,report,X_test,y_test,cm
```

classifier: the original classifier object

Accuracy: the calculated accuracy score

report: the generated classification report

X_test: the original test data

y_test: the true labels of the test data

cm: the confusion matrix

This function can be useful for evaluating the performance of a machine learning classifier on a test dataset. By calling this function with a trained classifier and test data, you can obtain the **confusion matrix, accuracy score, and classification report, which provide insights into the classifier's performance and help in assessing its effectiveness.**

```
def selectk_Classification(acclog,accsvml,accsvmnl,accknn,accnav,accdes,accrf):

    dataframe=pd.DataFrame(index=['ChiSquare'],columns=['Logistic','SVMl','SVMnl','KNN','Navie','Decision','Random'])
    for number,idex in enumerate(dataframe.index):
```

The purpose of using **enumerate()** in this context is to allow accessing both the **index position and the index label** within the loop, which can be useful for various operations on the DataFrame