**DS USING PYTHON LAB**

**EXPERIMENT: 06**

**AIM:** Implementation of Classification modeling

**PROBLEM STATEMENT:**

1. Choose a classifier for the classification problem.
2. Evaluate the performance of the classifier.

**THEORY:**

**K-Nearest Neighbors (KNN)**

K-Nearest Neighbors (KNN) is a simple and popular machine learning algorithm used for classification and regression problems. It is a non-parametric method, which means that it does not make any assumptions about the underlying data distribution.

The basic idea behind KNN is to classify a new data point based on the class of its K nearest neighbors. The distance between the new data point and the existing data points in the training set is calculated, and the K nearest neighbors are selected based on this distance measure. The most common class among these K neighbors is then assigned as the class label for the new data point.

For regression problems, instead of assigning a class label to the new data point, the algorithm calculates the mean or median of the K nearest neighbors' target values and uses that as the predicted value.

The choice of K, the number of neighbors to consider, is an important hyperparameter that can greatly affect the performance of the algorithm. A small value of K may lead to overfitting, while a large value of K may lead to underfitting.

One potential disadvantage of KNN is that it can be computationally expensive, especially for large datasets. Additionally, the algorithm may be sensitive to the choice of distance metric used and the scaling of the features.

To implement KNN, we need to follow these basic steps:

1. Collect and preprocess the training data.
2. Define the KNN model and select the appropriate value for K.
3. Train the KNN model using the training data.
4. Evaluate the performance of the KNN model using the test data.
5. Make predictions on new data points using the trained KNN model.

**Naive Bayes**

Naive Bayes Classifier is a probabilistic machine learning algorithm used for classification problems. It is based on Bayes' theorem and the assumption of independence between features.

The basic idea behind Naive Bayes Classifier is to calculate the probability of each class label given a set of features. The class label with the highest probability is then assigned to the new data point.

The "naive" assumption made in Naive Bayes Classifier is that all features are independent of each other, given the class label. This means that the presence or absence of one feature does not affect the probability of another feature being present or absent.

There are three main types of Naive Bayes Classifier:

* Gaussian Naive Bayes: This is used for continuous data and assumes that the data follows a Gaussian distribution.
* Multinomial Naive Bayes: This is used for discrete data such as text data, and assumes that the features follow a multinomial distribution.
* Bernoulli Naive Bayes: This is used for binary data, where the features take on only two values (0 or 1).

To implement the Naive Bayes Classifier, we need to follow these basic steps:

1. Collect and preprocess the training data.
2. Compute the prior probabilities of each class label.
3. Compute the likelihood probabilities of each feature given each class label.
4. Compute the posterior probabilities of each class label given the new data point.
5. Choose the class label with the highest posterior probability as the predicted class label for the new data point.

**Support Vector Machines (SVMs)**

Support Vector Machines (SVMs) are a popular type of supervised learning algorithm used for classification and regression problems. They work by finding a hyperplane that maximally separates the data points into different classes.

In binary classification problems, the hyperplane is a line that separates the two classes. In multi-class classification problems, multiple hyperplanes are used to separate the different classes.

SVMs are effective in high-dimensional spaces and can handle large datasets. They also work well with both linearly separable and non-linearly separable data by using a technique called kernel trick.

Here are the basic steps to implement SVM for binary classification:

1. Collect and preprocess the training data.
2. Define the SVM model and select the appropriate kernel function.
3. Train the SVM model using the training data.
4. Evaluate the performance of the SVM model using the test data.
5. Make predictions on new data points using the trained SVM model.

**Decision Tree**

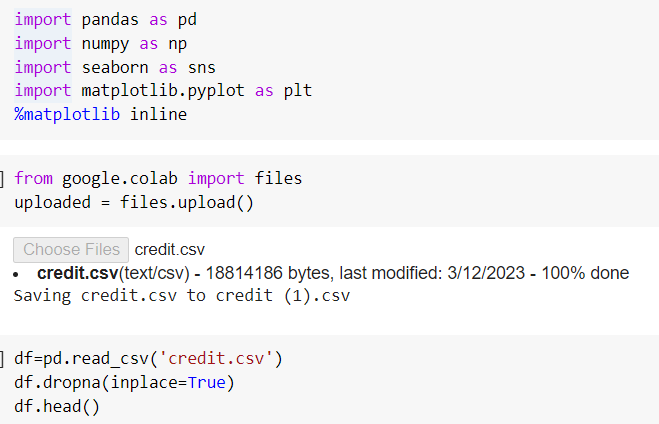
A decision tree is a popular type of supervised learning algorithm used for classification and regression tasks. It works by recursively splitting the data based on the values of the input features to create a tree-like model of decisions and their possible consequences.

Here are the basic steps to implement a decision tree for classification:

1. Collect and preprocess the training data.
2. Define the decision tree model and select the appropriate hyperparameters.
3. Train the decision tree model using the training data.
4. Evaluate the performance of the decision tree model using the test data.
5. Make predictions on new data points using the trained decision tree model.

IMPLEMENTATION:

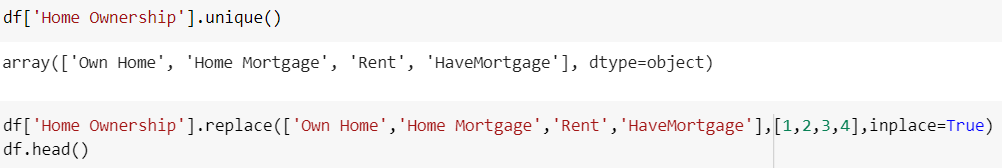
1. Loading the dataset and importing libraries



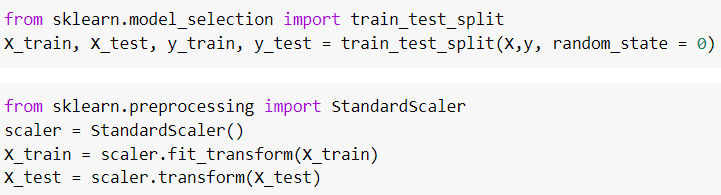
1. Determining the independent features as X and dependent features as y



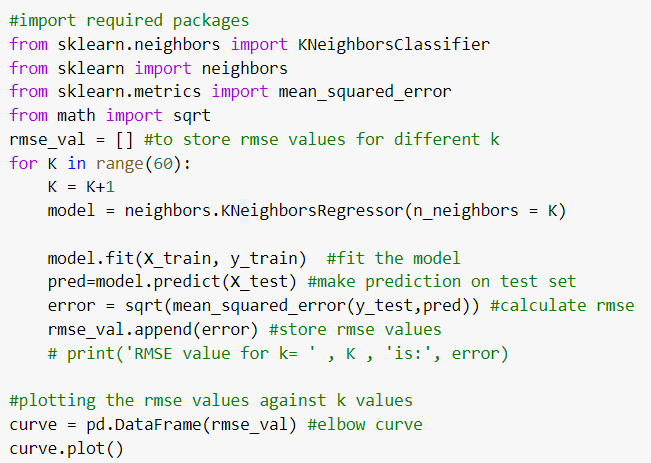
1. Giving unique values to each category

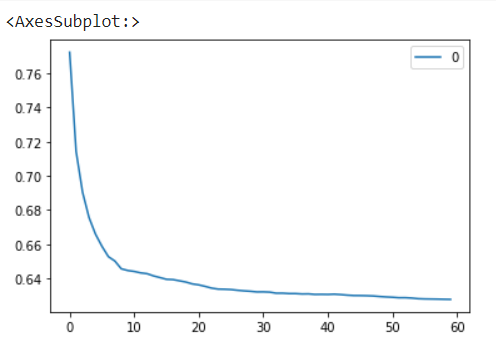


1. Splitting the dataset into train and test datasets and then standardizing the values

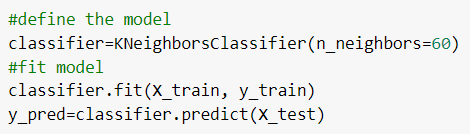


**K-Nearest Neighbor(KNN) Algorithm**

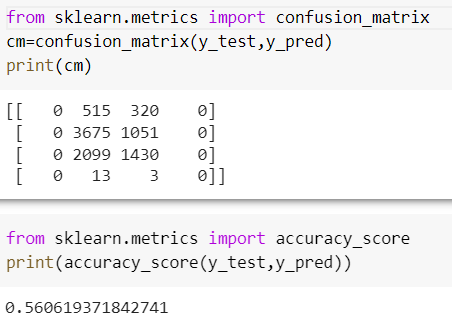
1. Finding the best value of n\_neighbour

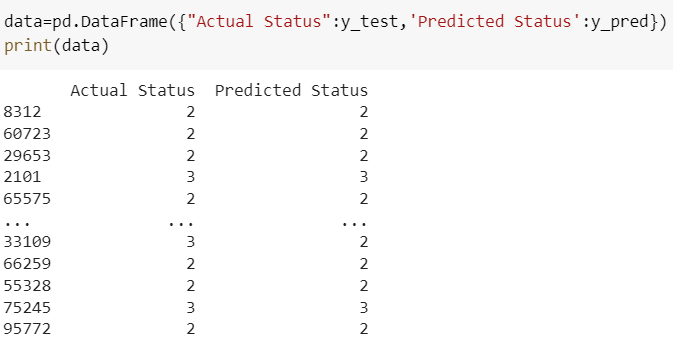


1. Model implementation

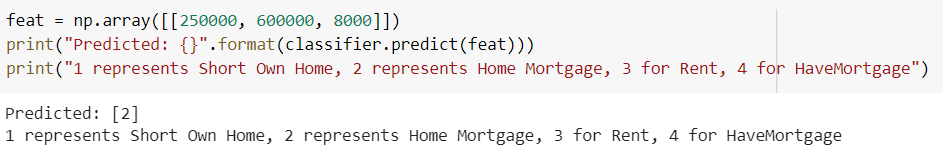


1. Model evaluation



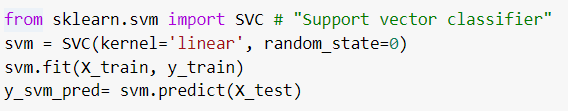


1. Prediction

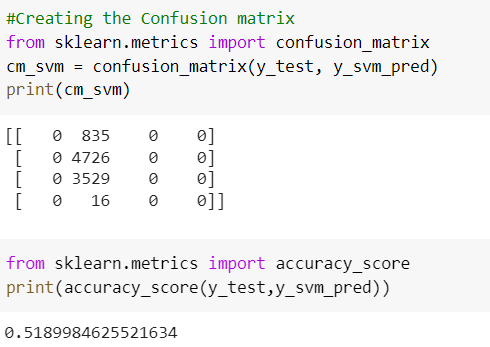


**Support Vector Machine (SVM)**

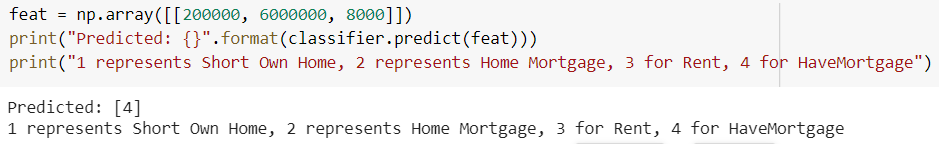
1. Model Implementation



1. Model Evaluation

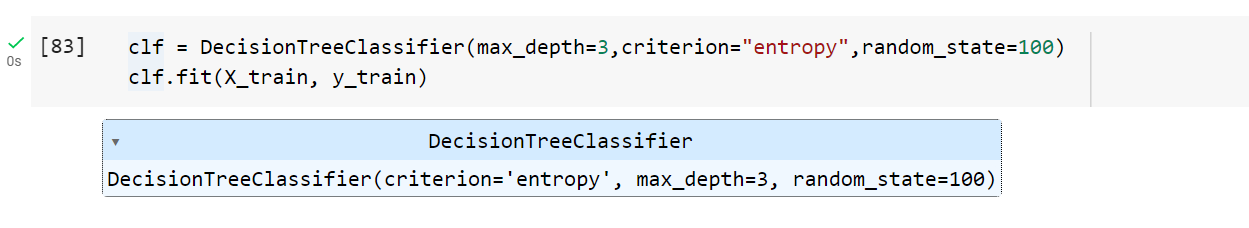


1. Prediction

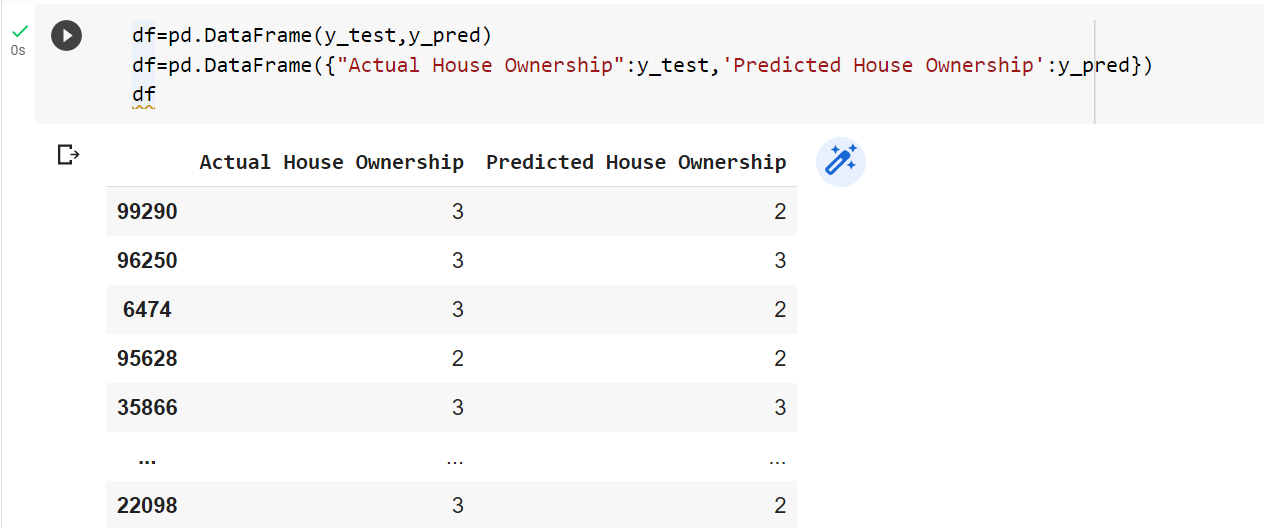


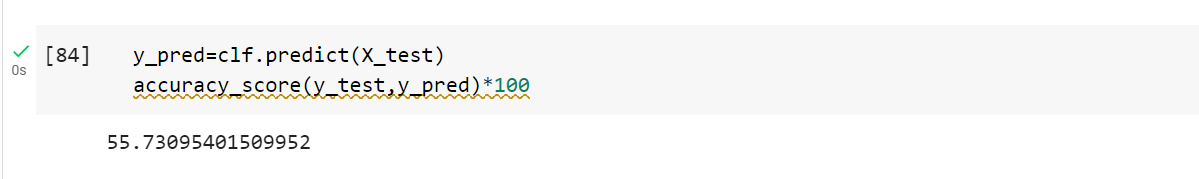
**Decision Tree Classifier**

1.Model implementation

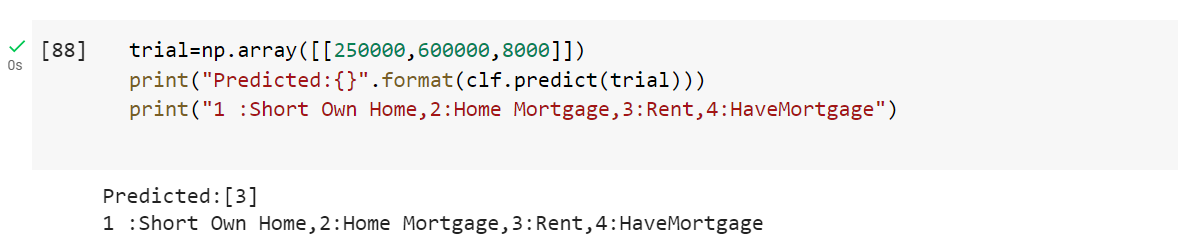
****

2.Model Evaluation

****

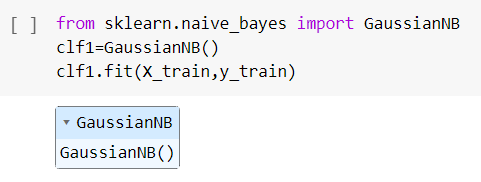
****

3.Prediction

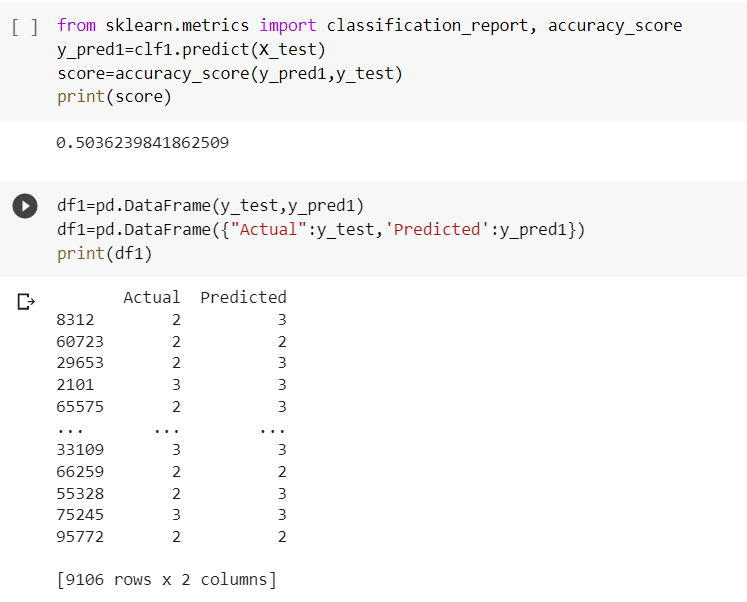
****

**Naive Bayes**

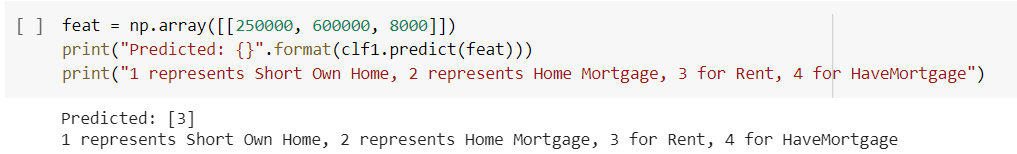
1. Gaussian Naive Bayes
   1. Model Implementation



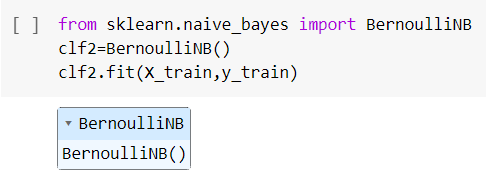
* 1. Model Evaluation



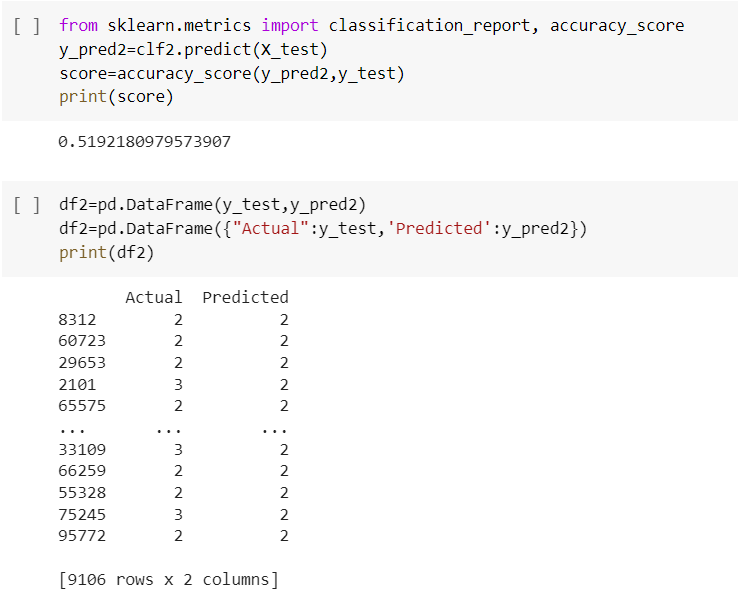
* 1. Prediction



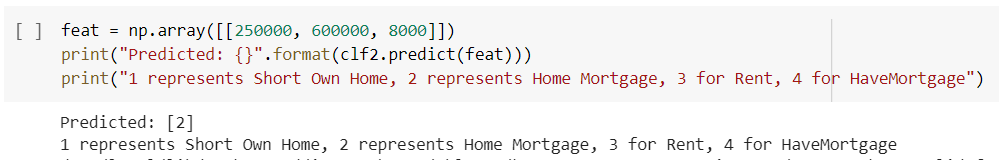
1. Bernoulli Naive Bayes
   1. Model Implementation



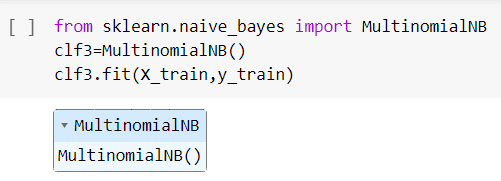
* 1. Model Evaluation



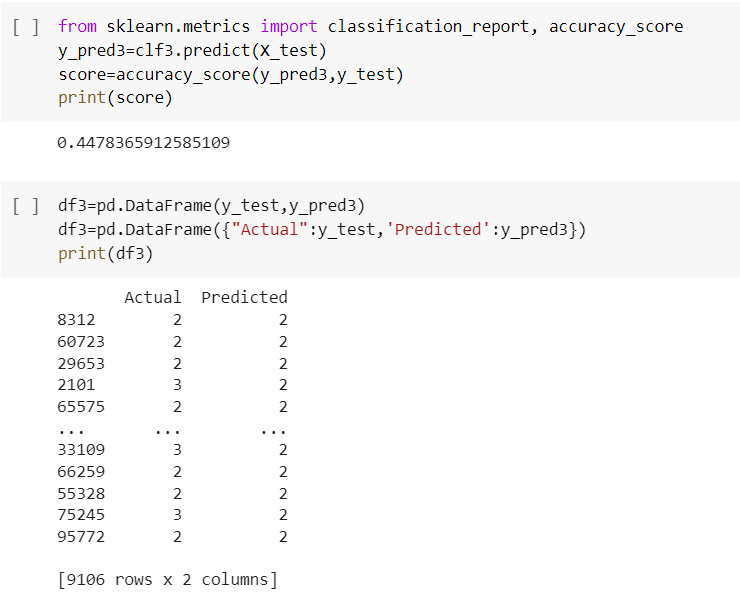
* 1. Prediction



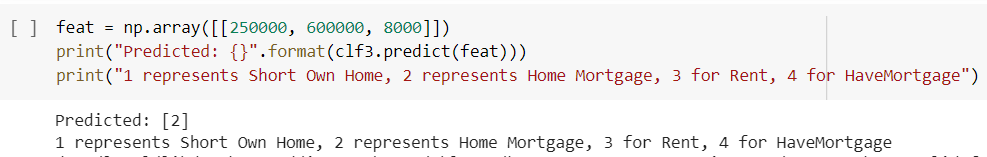
1. Multinomial Naive Bayes
   1. Model Implementation



* 1. Model Evaluation



* 1. Prediction



**CONCLUSION:**

In this experiment, we have predicted ‘Home Ownership’ with the help of different features like ‘Current Loan Amount, Annual Income and Monthly Debt’. This is done with the help of Supervised Machine Learning through Classification Algorithms like KNN, Naive Bayes, SVM and Decision Tree. All the algorithms have different algorithm scores.