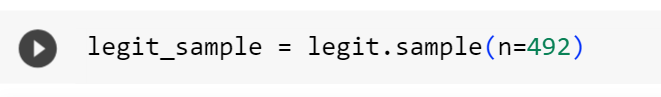
**Credit Card Fraud Detection**

**Phase 4: Development Part 2**

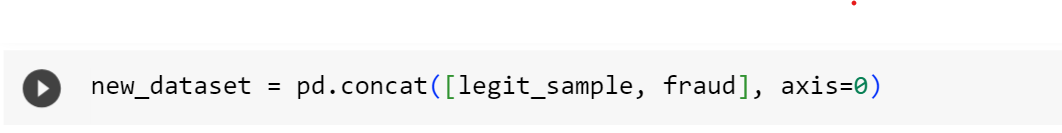
**1.FEATURE ENGINEERING:-**

**Under-Sampling and balancing the data**

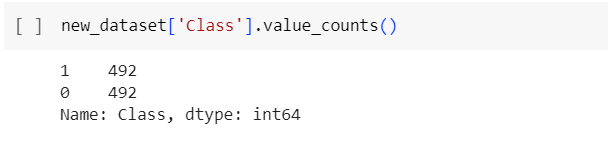
We first resample the data to balance the classes by selecting a random subset of "legit" transactions and then combines it with the "fraud" transactions. This new dataset is prepared for machine learning by separating the features (X) and labels (Y). It's important to note that balancing the classes is a crucial step when dealing with imbalanced datasets to prevent the model from being biased towards the majority class.



In this line, a random sample of 492 records (transactions) is taken from the "legit" subset of the dataset. This step is intended to balance the number of "legit" and "fraud" transactions in the new dataset.

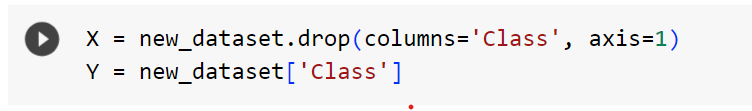


Here, the previously sampled "legit" transactions (492 of them) are concatenated with the original "fraud" transactions. The axis=0 argument indicates that the concatenation should be done along rows. As a result, "new\_dataset" now contains both the randomly sampled "legit" and "fraud" transactions**.**



This line counts the occurrences of each unique value in the "Class" column of the new dataset. It essentially shows the distribution of the two classes, which should now be more balanced due to the previous sampling.

**Splitting the data into Features & Targets**

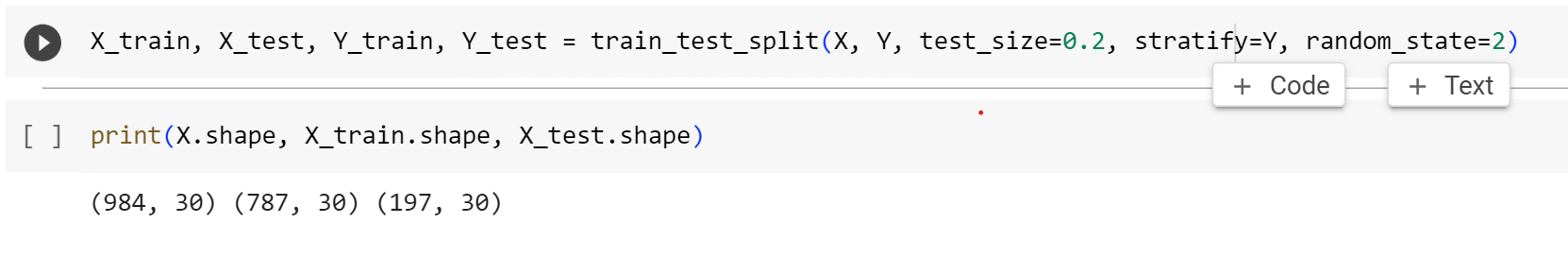


The "Class" column is dropped from the "new\_dataset" to create a feature matrix "X." This matrix contains all the input features that will be used for training the model. The target vector "Y," which contains the corresponding labels indicating whether each transaction is "legit" (0) or "fraud" (1). It's the variable that the model aims to predict.

**2.MODEL TRAINING**

**Split the data into Training data & Testing Data**

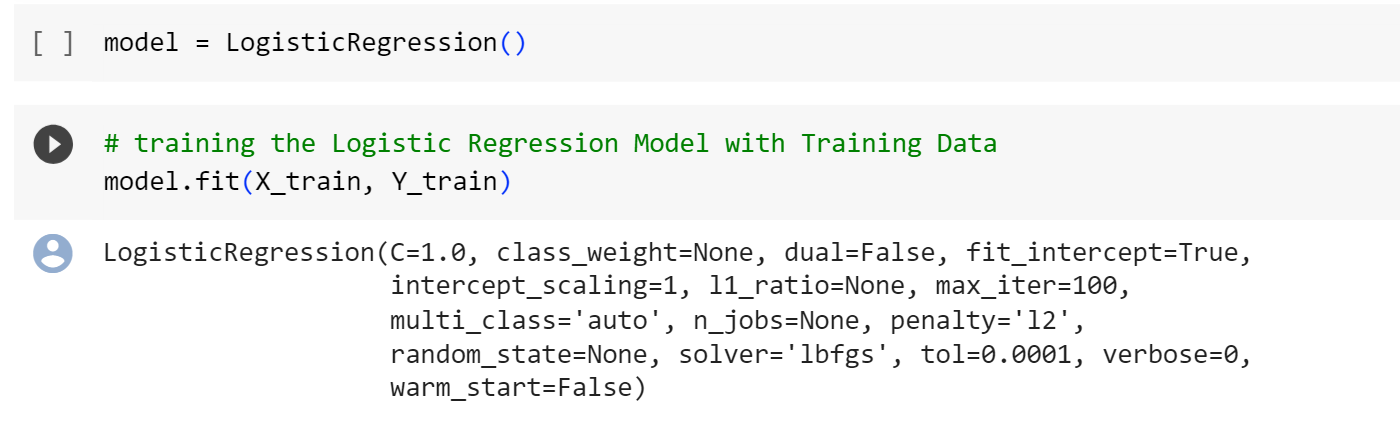
It is used to split a dataset into training and testing subsets using the train\_test\_split function.



These are typically used to represent the feature matrix (**X**) and the target vector (**Y**). **X** represents the features or attributes of your dataset, and each row corresponds to a data point, while each column represents a different feature. **Y** represents the target variable or labels, which you want your model to predict based on the features in **X**.

**Logistic Regression**

This creates an instance of the logistic regression model. In scikit-learn, Logistic Regression is a class used to create logistic regression models for binary classification tasks.



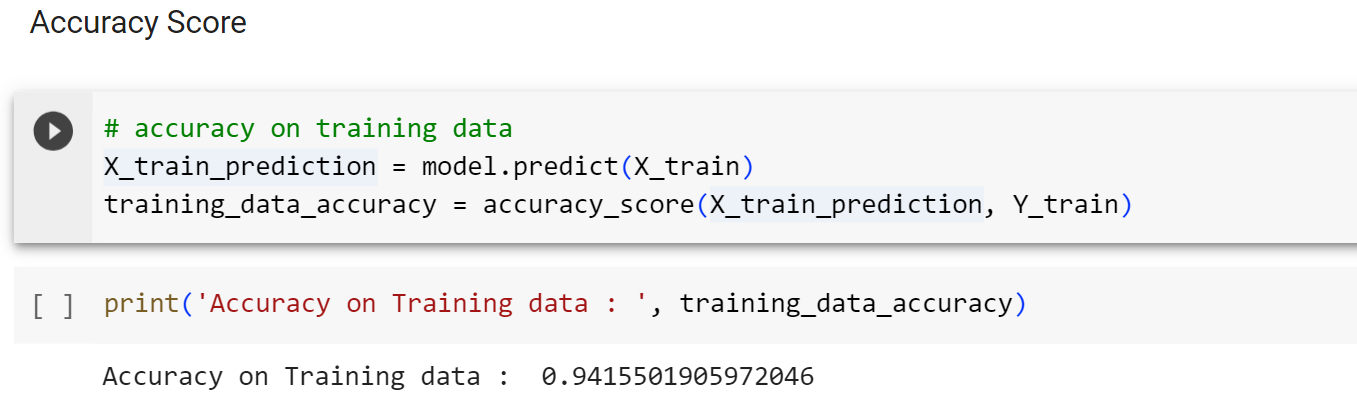
In summary, the code accomplishes the following:

* Creates a logistic regression model.
* Trains the model using the training data (**X\_train** and **Y\_train**).
* During training, the model learns the decision boundary that distinguishes the two classes based on the features in the training data.
* The trained model can then be used to make predictions on new, unseen data.

**3.MODEL EVALUATION**

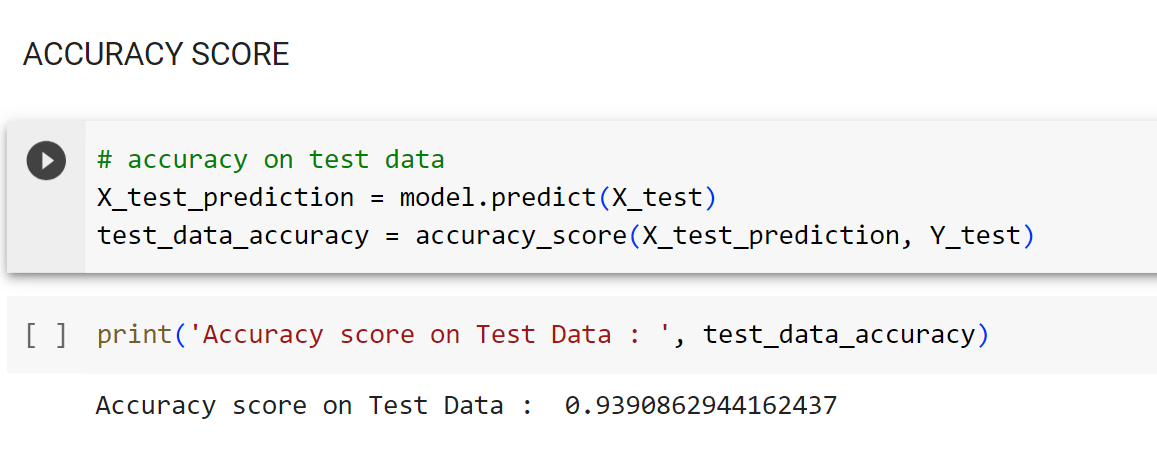
# Accuracy on training data :

* It calculates the accuracy of the logistic regression model on the training data.



It calculates the accuracy of the logistic regression model on the training data by comparing the model's predictions to the actual labels. The accuracy score represents the proportion of correct predictions made by the model on the training dataset, indicating how well the model is performing on data it has already seen during training.

Accuracy on testing data **:**

* This below code calculates the accuracy of the logistic regression model on the test data.
* The accuracy score reflects the proportion of correct predictions made by the model on a dataset it has never seen during training.

# 