**Mercedes-Benz Greener Manufacturing.**

DESCRIPTION

Reduce the time a Mercedes-Benz spends on the test bench.

Problem Statement Scenario:  
Since the first automobile, the Benz Patent Motor Car in 1886, Mercedes-Benz has stood for important automotive innovations. These include the passenger safety cell with a crumple zone, the airbag, and intelligent assistance systems. Mercedes-Benz applies for nearly 2000 patents per year, making the brand the European leader among premium carmakers. Mercedes-Benz is the leader in the premium car industry. With a huge selection of features and options, customers can choose the customized Mercedes-Benz of their dreams.

To ensure the safety and reliability of every unique car configuration before they hit the road, the company’s engineers have developed a robust testing system. As one of the world’s biggest manufacturers of premium cars, safety and efficiency are paramount on Mercedes-Benz’s production lines. However, optimizing the speed of their testing system for many possible feature combinations is complex and time-consuming without a powerful algorithmic approach.

You are required to reduce the time that cars spend on the test bench. Others will work with a dataset representing different permutations of features in a Mercedes-Benz car to predict the time it takes to pass testing. Optimal algorithms will contribute to faster testing, resulting in lower carbon dioxide emissions without reducing Mercedes-Benz’s standards.

Following actions should be performed:

* If for any column(s), the variance is equal to zero, then you need to remove those variable(s).
* Check for null and unique values for test and train sets.
* Apply label encoder.
* Perform dimensionality reduction.
* Predict your test\_df values using XGBoost.

Solution :

We need to predict the time taken for the cars to pass testing. Since time is a continuous variable, this is a regression problem. We will be using XGBoost Regressor to predict the time taken for cars to pass testing.

STEPS:

1. First we read the train and test datasets.
2. Find the shape of the datasets, print the column names and first 5 rows in the train and test datasets.

Table

Description automatically generated with medium confidence

Table

Description automatically generated

1. Perform EDA.

Chart, histogram

Description automatically generatedWe find that the average time taken for cars to pass testing is around 100.

1. Check for missing values. There are no missing values in the train dataset.

Graphical user interface, text, application

Description automatically generated

1. Check for the dtypes - There are 369 numeric and 8 categorical columns in train dataset.
2. Get the numeric columns. We need to remove the columns with zero variance. This means that columns that are having same values throughout for all rows aren’t much useful in learning. So, these columns can be dropped.

Graphical user interface, text, application

Description automatically generated

1. Get the categorical columns. Convert the categorical columns (string) to numeric, so they are in machine readable format. (Apply label encoder)

A picture containing table

Description automatically generated

1. Before applying PCA, the data needs to be centered and scaled. We use StandardScaler to center and scale the data. Scaled data has a standard deviation close to 1.
2. Perform PCA - Principal Component Analysis. PCA is a method of dimensionality reduction. Its useful in a dataset that has many columns(features). PCA reduces the no.of features, without losing much of information. Features which arent very significant are dropped.

Text

Description automatically generated

1. To understand how much of the variance is explained by each of these principal components, we use explained variance ratio. PCA1's explained variance ratio will be more than PCA2's and so on.
2. After applying PCA, the no.of features are reduced from 364 to 148, without much information loss. PCA improves processing time for very large datasets having many number of columns.

Graphical user interface, text, application, chat or text message

Description automatically generated

1. Now the train dataset is ready, we use XGBoost to calculate time - Dependent variable y. Since time is continuous, we use Regression. (XGBoost Regression)
2. Split the train data into train and test.
3. Fit the X\_train and y\_train.

Graphical user interface, text, application, email

Description automatically generated

1. Predict the y values for X\_test.

Graphical user interface, text, application, email

Description automatically generated

1. Perform error metrics for test: Calculate Mean square error, Root Mean square error, Mean Absolute error and Mean Absolute percentage error.

Graphical user interface, text, application, email

Description automatically generated

1. Perform error metrics for train.

Graphical user interface, text, application, email

Description automatically generated

1. Calculate R squared value.

Text

Description automatically generated

Chart, line chart

Description automatically generated

1. Now the model is ready, we need to predict the y value for test dataset.
2. Read the test dataset(External dataset).
3. Check for missing values. There are no missing values for test dataset.
4. We need to make sure that the no.of features are same for both train and test dataset. Only if the features are same, we can use our already trained model to predict the y values for our test data.(external dataset).

Graphical user interface, text, application

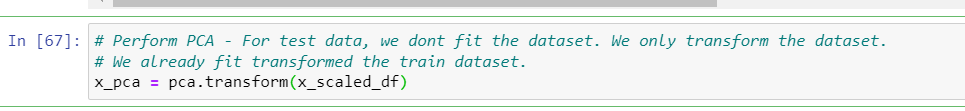
Description automatically generated

1. Drop the features in test data that is not present in train data. This is because the model is not trained for those features. No use in keeping these features in test data. Feature mismatch will occur if there are different no.of features in train and test data.

Graphical user interface, text, application, email

Description automatically generated

1. For test dataset, convert the categorical columns (string) to numeric, so they are in machine readable format. We use label encoder for this.
2. Before applying PCA for test data, the data needs to be centered and scaled. Use StandardScaler to scale the data.
3. Perform PCA - For test data, we dont fit the dataset. We only transform the dataset. We already fit transformed the train dataset. After applying PCA, the no.of features are reduced to 148. The no.of features in test dataset is same as the no.of features in train dataset.



A picture containing text

Description automatically generated

1. Do the prediction on already trained model.

Graphical user interface

Description automatically generated with medium confidence

1. Final Result: Predicted time for cars to pass testing.

Table

Description automatically generated