**Mercedes Benz Greener Manufacturing**

**Project**

**Made By-**

**Gaurav Pandey**

**Introduction**

In this project we are given two data set, one is training dataset and the other is testing dataset. We are required to reduce the time that car spends 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 testin. Optimal algorithms will contribute to faster testing, resulting in lower carbon dioxide emissions without reducing Mercedes-Benz’s standards.

Following actions should be performed:

1. If for any column(s), the variance is equal to zero, then we need to remove those variables.
2. Check for null and unique values for the train and test data set.
3. Apply Label Encoder
4. Perform dimensionality reduction
5. Predict the test\_df values using XGBoost.

**Explanation**

For solving this problem, I have performed some steps. They are:

1. First of all I have used pandas and numpy for loading the data set in my python.
2. After loading the data set I have checked for the null values in the training as well as the testing data.
3. There was no null values in any of the data set.
4. Then I loaded the data set by using head() function and understood the data.
5. Then I checked for the data types. I found that there were 369 int64 and 8 object and 1 float64 type columns in the training data set where as in test data set we have 369 int64 and 8 object type columns.
6. Then I dropped the dependent variable from the training data and stored it in a new variable.
7. I also dropped the ID variable from both testing and the training data because it was of no use to us for prediction.
8. Since we have to preprocess the data to make it suitable for prediction I separated the numerical and categorical data from both the training as well as the testing data set.
9. There were some columns in the training and the testing data set which have zero variance, we need to remove them because they are of no se to use since they hold same values for all the columns.
10. After that I applied the Label Encoder for the categorical data. This step is done to convert the categorical data into the numerical data.
11. Once we are done with the label encoding part I saw that the values from the numerical data set and the value we get after applying the label encoder has a huge difference.
12. Therefore, I applied the Scaling technique. I used MinMaxScaler from sklearn.preprocessing.
13. After applying the MinMaxScaler on the encoded data I combined the numerical and the newly formed numerical data set which we obtained from the categorical data set for both the training and the testing data.
14. Now we are done with the prepocessing of the training as well as the testing data.
15. Now I applied the PCA (Principal Component Analysis) on both training as well as the testing data in order to reduce the number of columns
16. Here in this case, I have reduced the number of columns to 12.
17. Once we are done with the PCA, now we need to declare our dependent and independent data for training.
18. I have declared my dependent and independent variable for training.
19. Then I used the train\_test\_split() function from the sklean.model\_selection to split my preprocessed training data into the testing and the validation data. 80% of my total data is considered for training the model and the remaining 20% of the data is for validating the model.
20. Once we are done with the splitting of data into the training and the validation part. I used the xgboost to train my model.
21. I used the r2\_score metrics from the sklearn.metrics to evaluate my model.
22. I validated my data.
23. At last, I let my model predict the value on the testing data and I stored the predicted and the id value in a new data frame and the I uploaded that data frame onto a csv file using pandas and saved my output.