Car Popularity Prediction
Goldman Sachs Codesprint

Problem Statement
A car company has the data for all the cars that are present in the market. They are planning to introduce some new ones of their own, but first, they want to find out what would be the popularity of the new cars in the market based on each car's attributes.

We will provide you a dataset of cars along with the attributes of each car along with its popularity. Your task is to train a model that can predict the popularity of new cars based on the given attributes.

Solution

In order to solve the problem we can view it as a classification problem rather than viewing it as a regression problem, the reason it can be considered as a classification problem is the predictions for 'popularity' of a car can be in range from 1 to 4.

So to solve such a classification problem with a large dataset of 10,000 records my approach was to use Light GBM (i.e. Light Gradient Boosting Method). As we can analyse from the dataset all values for any property of a car falls in a specific range. Thus the first and foremost approach that comes in mind is to use 'Decision Tree' based models. But seeing such a huge dataset it won't be a reliable model.

Thus to overcome the issue of large datasets we can use LGBM classifier, the GBM (boosted trees) has been around for really a while, and there are a lot of materials on the topic. LGBM is used for supervised learning problems, where we use the training data (with multiple features) xi to predict a target variable yi. Light GBM is a gradient boosting framework that uses tree based learning algorithm. As per the results obtained in past tend to be very promising and the current problems dataset possess the properties which allows us to apply the method LGBM to obtain predictions.

Light GBM grows tree vertically while other algorithm grows trees horizontally meaning that Light GBM grows tree **leaf-wise** while other algorithm grows level-wise. It will choose the leaf with max delta loss to grow. When growing the same leaf, Leaf-wise algorithm can reduce more loss than a level-wise algorithm.

Libraries Used
1. Sklearn (scikit-learn)
2. Seaborn
3. MatplotLib
4. Numpy
5. Pandas
6. XGBoost
7. Jupyter Notebook

Using The Code To Obtain Prediction.csv
1. Run the jupyter notebook file submitted as a solution using 'jupyter'
2. Store the test.csv and train.csv in the same directory as that of jupyter notebook under 'data/' folder.
3. Create a csv named update.csv in the same directory as that of jupyter notebook with a single column 'popularity' or use the one provided in .zip of solution.
4. Step by step execute each step of jupyter notebook.

5. After completion you will get a 'prediction.csv' as output in same directory.

Note - The program being executed in the directory should have access to write and read data.

References
1. https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc