Name: R Sankeerthan Reddy

Email address: ratnasankeerthan@gmail.com

Contact number: +91 8374776349

Any desk address: **Bhupalpally, Telangana,506169**

Years of Work Experience: **1 Year (internship only)**

Date: **22nd Jul 2020**

Self-Case Study -1: Mercedes Benz Greener Manufacturing

Overview

1. Mercedes-Benz has stood for important automotive innovations. 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 each and every unique car configuration before they hit the road, Daimler's engineers have developed a robust testing system. But, optimizing the speed of their testing system for so many possible feature combinations are complex and time-consuming without a powerful algorithmic approach. As one of the world's biggest manufacturers of premium cars, safety and efficiency are paramount on Daimler's production lines.

2. The main of the project is to reduce the testing time and reduce the cost delivery time of the vehicle with the standard of the vehicle.

3. The performance metrics for this competition is R^2 error for the evaluation of the model.

Research-Papers/Solutions/Architectures/Kernels

1. https://docs.interpretable.ai/stable/examples/mercedes/#Optimal-Feature-Selection

   Objective: The main objective is to predict and minimize the cost of failure.

   - In the above document, they have used the Optimal Feature Selection. i.e. the relationship between the model performance and the number of features selected.
   - They have considered only few features and trained the model and checked the model performance.
   - They used the second method as trained the entire model on data set with elastic net.
   - Elastic Net is a middle ground between Ridge Regression and Lasso Regression. The regularization term is a simple mix of both Ridge and Lasso's regularization terms, and you can control the mix ratio r. When r = 0, Elastic Net is equivalent to Ridge Regression, and when r = 1, it is equivalent to Lasso Regression.
   - When we compare the both the models, we can observe that the elastic net peaks before 100 features. that both methods reach the same level of out-of-sample performance of around 0.62, but there is a significant difference in the number of features needed to attain this performance. As seen earlier, Optimal Feature Selection reaches this level with 6-8 features, whereas the elastic net needs close to 80.

   - This gives strong evidence to reinforce the earlier point that elastic net and lasso are not designed for feature selection, but rather for robustness. They give sparser solutions than other regularization approaches, but these solutions are not as sparse as they could be. Using Optimal Feature Selection to solve the feature selection problem to optimality gives us the ability to truly explore the tradeoff between model performance and sparsity exactly.

   This gave me a clear idea about the new technique that should be used in dimension reduction using the Elastic Net.

2. https://satacroteam.github.io/Mercedes-Benz-Greener-Manufacturing/

    - In the above document they have use 4 different types of models. They have used the stacking algorithm for the measurement of the model performance
    - They have used all the features and they have just removed the outliers of the data only

ML model	
Xgboost ICA PCA TSVD GRP SRP Stack Optimization	0.57066
Xgboost ICA PCA TSVD GRP SRP Stack	0.56831
Xgboost ICA PCA TSVD GRP SRP	0.56784
Xgboost ICA PCA	0.55983

    - Here they used the combination of Xgboost and dimensional reduction technique
    - But the model with the Xgboost and FastICA, PCA this results in the r^2 error as 0.5598.

3. http://rstudio-pubs-static.s3.amazonaws.com/491189_05618809272041428036915 1c791cf9e.html#17_interaction_between_outcome_and_covariates

    Here they are used the deep EDA on the data set. they have performed all types of visualization on the data set for the better understanding.

    They have converted the categorical data into one hot encoding and moved on to next process

    According to the document they did not remove any columns

    The categorical variables (X0-X8) were converted by one-hot-encoding. These along with the binary variables (X10-X385) were tested for their significance as described in section 1.7 and only those features with p-values less than 0.01 are selected. Also, the new variable cluster is converted by one-hot-encoding and added to the train and test features set.

Extreme Gradient Boosting tree-based method is used for building model. The competition evaluation metric was coefficient-of-determination R^2. However, the model fitted using root-mean-square-error (RMSE) as performance metric is found to be more accurate than that by mean-absolute-error (MAE) and R^2.

This clearly ideates that we can use RMSE as the performance metric for the custom implementation

4. https://mc.ai/self-case-study-on-mercedes-benz-greener-manufacturing-competition/

From the above blog we can understand that

- In categorical features

Feature X0: There are total 47 categories in X0. In some categories their moderate outlier point's which can be ignored. There are few categories which contains very few points.

Feature X1: There are total 27 categories in X1. But the corresponding target values are not well separated and distributed.

Feature X2: There are total 43 categories in the feature, few categories have very few data point's hence after splitting the dataset into train and test, it may so happen that some categories are not included in train and only in test. This may create unseen category error. Hence, we will need to treat this further.

Feature X3: Total Categories 7 and has very little variance between categories. This may not be good for a Tree based algorithm.

Feature X4: Total categories 7 again the same problem of unseen category error may pop-up.

Feature X5: Total categories in X5 are 29.

Feature X6: Total categories in X6 are 12.

Feature X8: Total categories in X8 are 25. There is little variance between the features Convert these features into one hot encoding for the categorical data and for the numerical data they have applied the feature engineering technique.

- For Featurization we are using Singular Value Decomposition (SVD) and Principal Component Analysis (PCA), this decomposition techniques will be applied to numerical features. The features are so decomposing that maximum variance is restored.

On further they have applied the random forest, decision trees for training of the model and turned for the better performance of the model

I can conclude that use these categorical features and some numerical features and we can build the machine learning model.

5. https://www.kaggle.com/sudalairajkumar/simple-exploration-notebook-mercedes

   From this kernel I understood that there are some categorical and numerical data that is used for the cal. Of analysis od the testing time data. by using some basic eda we can do the feature engineering

   From the above Kaggle kernel we can conclude that the we can use label encoding and find the feature importance of the features.

   After getting the features use the features that has more importance.

   Consider those features and build the model using the random forest.

First Cut Approach

1. **Download the dataset**
2. **ML problem and Performance Metric:**

   a. This is a Regression problem.

   b. Performance metric is R^2 error.

3. **Train Test Split**:

   Since this is not time series data, I will split the Train data randomly in to 70% train and 30% test. Stratified splitting will be done.

4. **EDA**

   I will perform univariate analysis like box plot on each feature.

   I will check the features and remove the features least important features and I will remove them.

   Plotting boxplot for categorical features.

   Create a new feature called is Zero which is set to 1 if the original feature is zero.

   Use DT to convert numerical feature to categorical feature

5. **Model selection & hyperparameter tuning.**

   Here I am planning to use all the features and use dimensionality reduction such that I don't want to lose the data

   I am planning to use Random Forest/Xgboost. Also, I want to experiment with stacking models, dimensional reduction technique.

Will perform hyper parameter tuning, select the best parameters and fit the model and compare cost of all models.

Will select the model which gives less cost and high f1 score.

6. **Build the data pipeline**
7. **Deploy the model in the cloud using flask or steamlit**

Other references:

https://towardsdatascience.com/ridge-lasso-and-elastic-net-regularization-7861ca575c64