**Mercedes-Benz car testing data analysis**

**Sicheng Zhang, 001239226**

**Abstract**

In this project, the main aiming is predicting the time that cars spend on the test bench and evaluate R^2 value, also called the coefficient of determination, by using the dataset provided by Mercedes-Benz. First I do an in-depth Exploratory Data Analysis, which gave us a brief view of what the data looks like and handle the problems in dataset. During this process, I clean the data by check missing values and outliers and plot the distribution of each columns’ data. In the next step-data analyzing, I first reduce dimension by using T-SNE for visualization. And then select important features with supervision by using Random Forest Regressor which shows the importance and correlation coefficients.

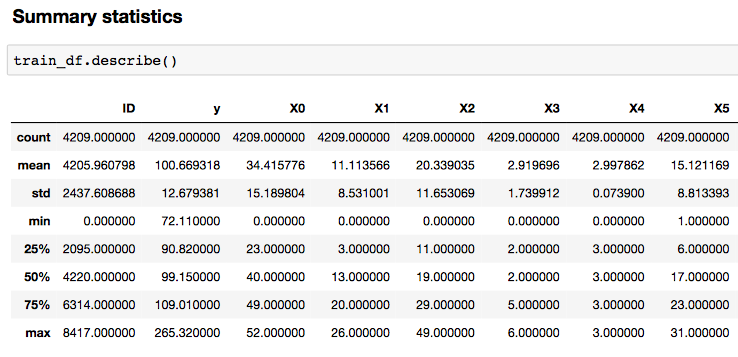
**Introduction**

In this competition, Daimler is challenging Competitors to tackle the curse of dimensionality and reduce the time that cars spend on the test bench. Competitors will work with a dataset representing different permutations of Mercedes-Benz car features to predict the time it takes to pass testing.

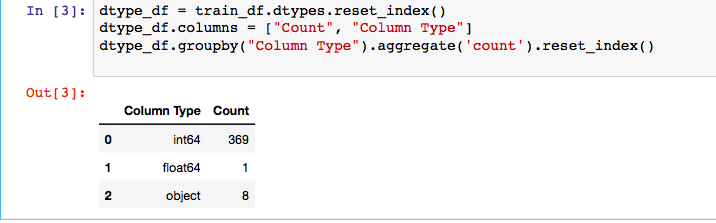
1. **EDA**

In exploratory data analysis, I take a detailed look at our data and do some preprocessing on the data for future use.

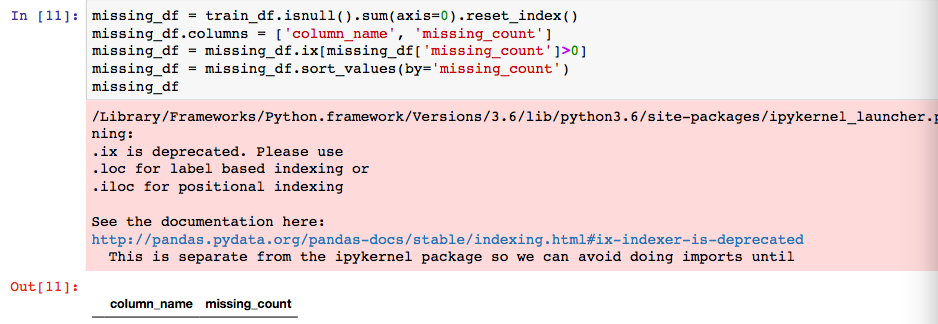
Initially, I use describe() to see the description of a data frame, including max value, min value, mean value and median value of each feature.

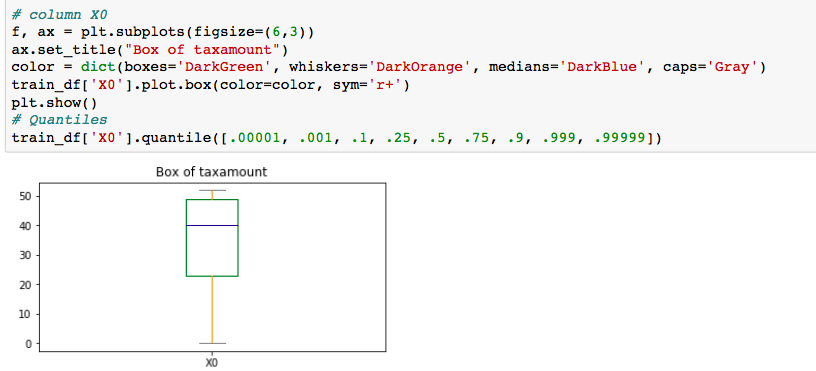


Then, I check the data type of each column, and encode categorical columns to numerical. Meanwhile, I find some features that are constants which means have little influence on target y, so I remove them.

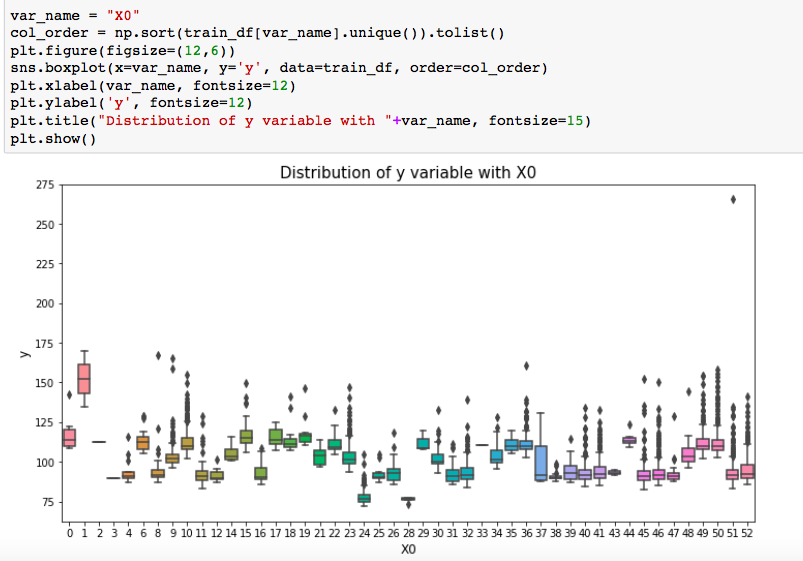


Next, I find there are no missing value and outlier.





Finally, I plot distribution of target y variable with every single columns.

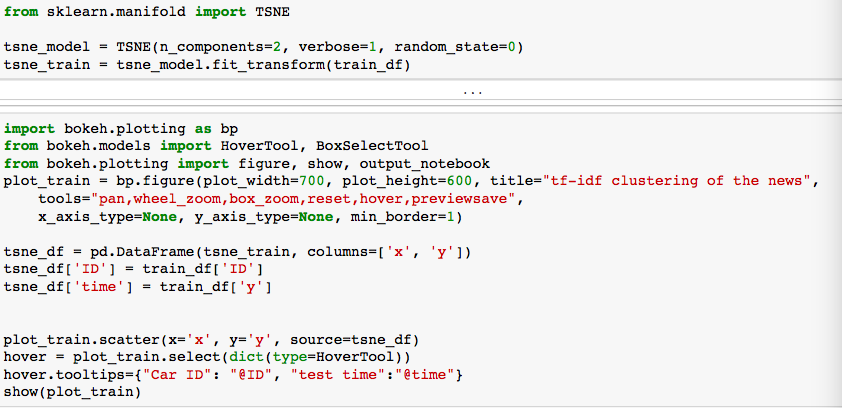


1. **Feature selection**

During this part, I used both supervised and unsupervised feature selection methods.

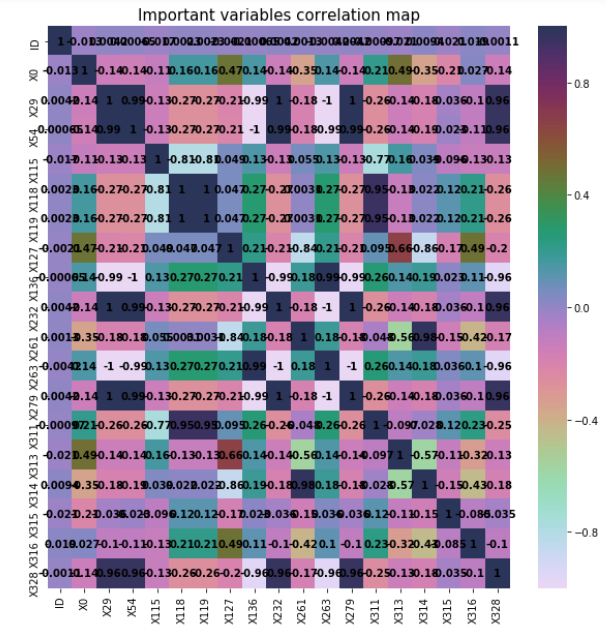
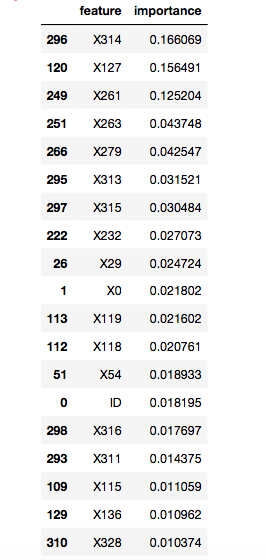
* Supervised

Reduce dimension by using T-SNE and interactive frontend visualization by using Bohek

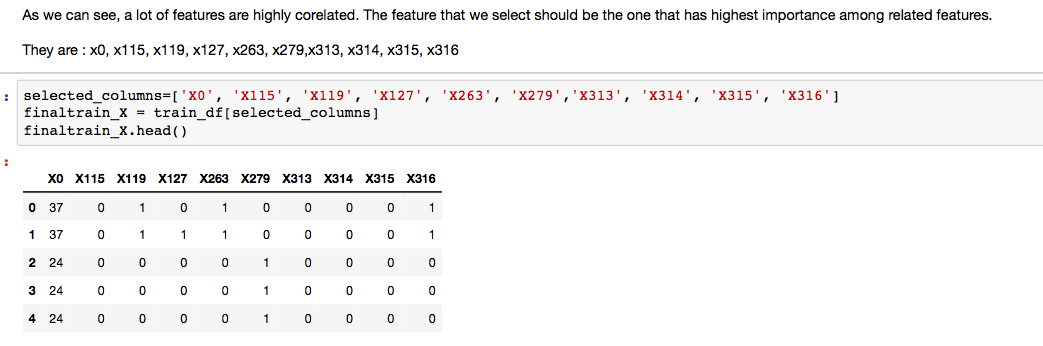


* Unsupervised

Use random forest to calculate importance and correlated coefficient, then manually select features according to both importance and correlated coefficient.

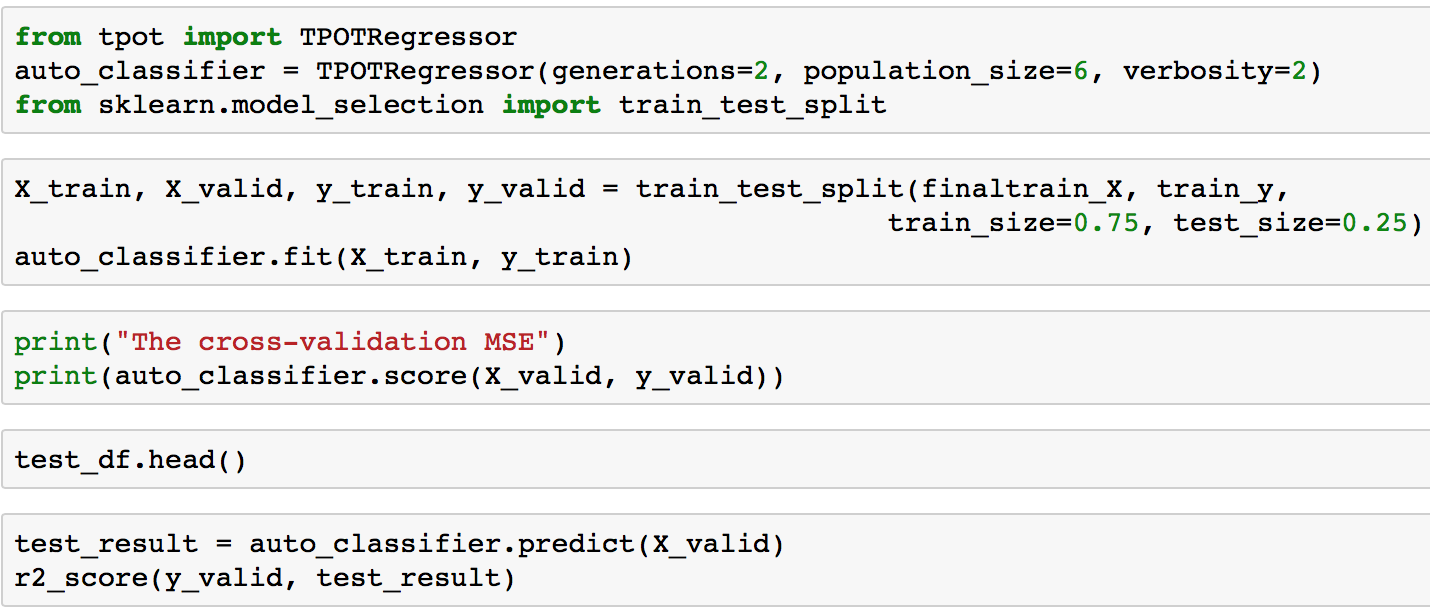


* output

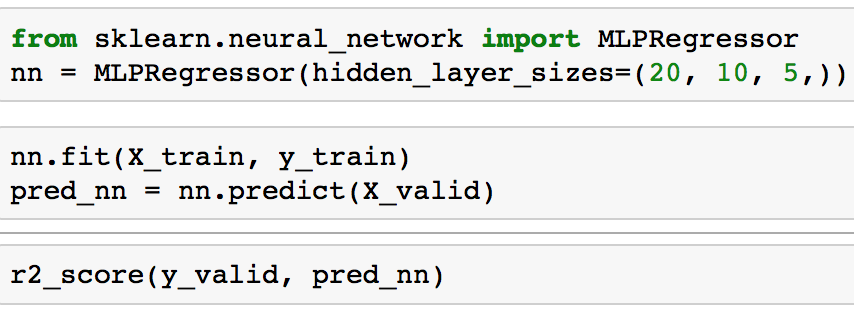


1. **Data training**

Now I train the model with TPOT and neural network and evaluate the models with R^2, the coefficient of determination.



TPOT code



Neural Network code

**Results**

I calculate R^2 (coefficient of determination) as the result. The result of TPOT is 0.59, while the neural network gets 0.57.

**Discussion**

I talk about the results from two part, the feature selection process and the training part.

I use two methods of feature selection. One is manually select important features based on the co-relations of the features. The other one is using dimension reduction algorithms, such as t-SNE.

The advantage of dimension reduction algorithms is its simplicity. I don’t have to do much and just let the algorithm do the selection. The multi-dimensional features are embedded into a loIr dimensional vector which contains the truncated components and use them as the features.

While calculating importance and co-relation and manually selecting can visualize the whole process. It takes more time and labor, and may be not that accurate or mathematically rigorous compared to dimension reduction algorithms. Sometimes it may lose some information from the data even if it might be minor.

Reference

<http://pandas.pydata.org/pandas-docs/stable/>

<https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding>

[http://scikitlearn.org/dev/modules/generated/sklearn.ensemble.RandomForestRegressor.html](http://scikit-learn.org/dev/modules/generated/sklearn.ensemble.RandomForestRegressor.html)

<http://www.kenrockwell.com/tech/bokeh.htm>

[http://scikitlearn.org/stable/modules/generated/sklearn.neural\_network.MLPRegressor.html#sklearn.neural\_network.MLPRegressor](http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html#sklearn.neural_network.MLPRegressor)

<https://github.com/EpistasisLab/tpot/issues/257>