**MSBD 6000b** **Li Meng**
Project 1 20477168

## Abstract

This project is divided into two processes, data preprocessing and classifier. I adopted quantile transformer to preprocessing the data by project them into the same range or distribution and reduce the influence of outliers effectively. For classification, Extremely Randomized Trees is used to train the model through the training data. To evaluate the accuracy of the model, cross validation model is adopted.

## 1 Preprocessing

In general, learning algorithms is usually benefit from standardization of the data set. In this part, I have tried different normalizers, scalers, transformers to achieve a better result.
The easiest way to standard the dataset is to transform the data to center it by removing the mean value of each feature, then scale it by dividing non_constant features by their standard deviation. But in this way, the distribution of the dataset are often been ignored. Beside data standardization, the feature can also be scaling into a range between a given minimum and maximum value. However, it will destroy the sparseness of the data. To keep the sparseness and reduce the impact of outliers, quantile transformer is adopted to preprocessing the data.
Quantile transformer puts each feature into the same distribution. through performing a rank transformation, it smooths out unusual distribution and is less influenced by outliers than above algorithms.

## 2 Classification

### 2.1 Decision tree

Decision trees are a non_parametric supervised learning method used for classification. A decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label. The top node is the root node. An attribute selection measure is a heuristic for selecting the splitting criterion the "best" separates a given data partition of class-labeled training tuples into individual classes. Information gain is a most used attribute selection measure. It's based on the information theory work by Shannon. The attributes are ranked by their information gain. The expected information (entropy) needed to classify a tuple in $D$ is:

$$Info(D) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

Information needed to classify $D$ is

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{D} * Info(D_j)$$

Information gained by branching on attribute A is

$$Gain(A) = Info(D) - Info_A(D)$$

Decision trees are simple to understand and interpret. But decision tree learners can create over-complex tree that do not generalize the data well.Some times trees can be unstable because small variations in th data might results in a completely different tree being generated. To overcome above problems, decisions trees within an ensemble are used.

### 2.2 Extremely Randomized Trees

In random forests, every tree in the ensemble is built from a sample drawn with replacement(i.e., a bootstrap sample) from the training set. When splitting a node during the construction of the tree, instead of choosing the best split among all features, the split is picked among a random subset of the features. Further more, in extremely randomized trees, randomness goes one step further in the way splits are computed. Instead of looking for the most discriminative thresholds such as random forests do, thresholds are drawn at random for each candidate feature and the best of these randomly-generated thresholds is picked as the splitting rule.

## 3 Evaluation

In this project, I apply k-fold cross-validation to evaluate different models. First I divided the training dataset and label into 20 sub-samples. In each run, use one distinct sub-sample as testing set and the remaining 19 sub-samples as training set. Finally, I evaluate the model using the average of the $k$ run. This method effectively reduces the randomness of training set/test set.

## 4 Experiments and results

### 4.1 Preprocessing

In this part, I test different preprocessing algorithms with Extremely Randomized Trees to see their performance.

| | Standard Scaler | Min Max Scaler | Robust Scaler | Quantile Transform |
|---|---|---|---|---|
| score | 0.952761 | 0.952141 | 0.952141 | 0.955872 |

### 4.2 Classification

In this part, several classification methods are applied to train the dataset after preprocessed by quantiles transform.

| | Logistic Regression | SVM | Random Forest | Extremely Randomized Trees |
|---|---|---|---|---|
| score | 0.940353 | 0.948104 | 0.9440825 | 0.955872 |

From the above tables we can see that quantiles transform and extremely randomized suits the dataset best. So I adopt them to predict the test dataset.

## References

[1] Scikit-learn: Machine Learning in Python, Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and

Brucher, M. and Perrot, M. and Duchesnay, E., Journal of Machine Learning Research, 12, 2825–2830, 2011

[2] Data mining: concepts and techniques, Jiawei Han, Micheline Kamber, and Jian Pei and Pete Barnum Morgan Kaufmann; 3 edition (July 6, 2011)