Feature Selection and Tune the Hyperparameters of Random Forest Classification

Written by Tania Natasya Tanuwijaya (1825453)

Random Forest is a machine learning technique that can solve classification and regression problems. Random forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. Therefore, a random forest is suitable for big data. The result of the random forest in a wide diversity that generally results in a better model. Tuning the parameters in Random Forest effectively enhances the model's accuracy. Increasing the number of trees might enhance the accuracy. However, lowering the number of estimators might speed up the model. Furthermore, scaling and selecting the data features is also important to enhance the accuracy of the model. Random forest improves bagging because it decorates the trees by introducing splitting on a random subset of features. It means that at each split of the tree, the model considers only a small subset of features rather than all the model's features.

Approach – Random Forest consists of a set of unpruned classification-based trees. It works by selecting random samples in the given training dataset. It uses bagging and features randomness when building each tree to create an uncorrelated forest of trees which prediction is more accurate than that of any individual tree. A decision tree was constructed for each random sample and got the prediction for each decision tree. Before tuning the hyperparameters, the data is normalized to enhance accuracy. Finding the optimal hyperparameters for tuning is important since it will influence the model performance to train the data. However, by calculating the Mean Absolute Error (MAE), the training data is slightly overfitting (Figure 1). Dimension Reduction and Features Selection are performed to reduce the risk of this overfitting. We first find the number of components and important features in the data. Then, re-fitting it to the Random Forest model for the training data to enhance the accuracy.

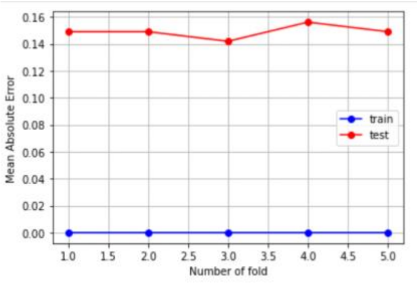

Figure 1: The K-fold cross-validation graph

Results – Compared to the other models, Random Forest with the default parameter shows the best accuracy, around 83-85%. This number is 10%, 10%, and 17% higher than the Logistic Regression, the SVM, and the K-Nearest Neighbor respectively. After the GridSearch has applied to the Random Forest to search for the best parameters, it increases the accuracy to 85.8%, with the 91.2% of F1 score. Furthermore, it is checked that the training data is a little overfitting the prediction. Thus, dimension reduction is applied to the model due to reducing the model complexity. However, the accuracy drops to 80%. Therefore, Feature Selection is chosen and re-fitting to the Random Forest model. The accuracy increases to 86.5%, while the F1 score also gains 0.5% higher (Figure 2). Hence, the Random Forest with Feature Selection is a slightly better model for predicting this dataset. Nevertheless, the F1 score interprets the incorrect classified cases better than the Accuracy Metric. In most real-life classification problems, imbalanced class distribution exists, and thus F1-score is a better Metric to evaluate the model (Huilgol, 2019).

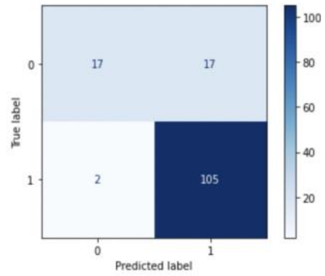

Figure 2: Confusion Matrix of the training dataset with 86%

Pros and Cons

1. Pro: Random Forest produces good predictions. It consists of multiple single trees based on a random training data sample. They are typically more accurate than single decision trees (Deng, 2018).
2. Pro: The random forest algorithm provides a higher level of accuracy in predicting outcomes. It also has a higher true and false positive rate as the number of explanatory variables increases in a dataset (Deng, 2018).
3. Con: Random Forest requires more resources for computation. These include node size, the number of trees, and the number of features sampled. In addition, the Random Forest needs more resources to store these data to handle large datasets and its computation (IBM, 2020).
4. Con: It consumes much time for training. Random Forest is slow in generating the predictions because it combines multiple decision trees. Whenever it makes a prediction, all the trees in the forest must predict the same input and then vote on it (Navlani, 2018).

Contributor Thought – Random Forest reaches a quite good accuracy and F1 scores, even with the default parameters only, though it takes more effort and time to train the data (2-4 minutes in this dataset).

Conclusion – Random Forest is one of the models in machine learning that solve classification problems. This model builds multiple decision trees and merges them to get a more accurate and stable prediction. Although it requires more time to train, this model enhances accuracy effectively by tuning the hyperparameter. Furthermore, Feature Selection is also important to reduce the data complexity. From the test dataset, Random Forest can predict whether the startups succeed or fail in about 85% accuracy.

Appendix

Deng, H. (2018) *Why random forests outperform decision trees*. Available at: https://towardsdatascience.com/why-random-forests-outperform-decision-trees-1b0f175a0b5 (Accessed: 24 December 2021).

Huilgol, P. (2019) *Accuracy vs. F1 score*. Available at: https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2 (Accessed: 24 December 2021).

IBM, (2020) *Random forest*. Available at: https://www.ibm.com/cloud/learn/random-forest (Accessed: 24 December 2021).

Navlani, A. (2018) *Understanding Random Forests Classifiers in Python*. Available at: https://www.datacamp.com/community/tutorials/random-forests-classifier-python (Accessed: 24 December 2021).

Probst, P., Wright, M. N., & Boulesteix, A. L. (2019). 'Hyperparameters and tuning strategies for random forest.' *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(3), e1301. doi: 10.1002/widm.1301.