# INM431 Machine Learning Coursework Supplementary Material

By J. Leplat, December 2020

## Introduction

The coursework’s main goal is to compare, contrast and critically evaluate two ML models. The final two models selected for comparison are Naïve Bayes and Random Forest. These two methods address the problem of machine learning in different ways, hence the comparison should provide some interesting insights.

When comparing methods it is interesting to find out how the performance varies, as parameters are gradually changed. In this way it is possible to find the optimal parameter values, and have a reasonable confidence that the parameters do not overfit the training data.

The approach taken is iterative. The plan was to compare decision trees to Naïve Bayes, time allowed the random forest method to be attempted also. The poster makes no reference to the decision tree results. The comparison of Naïve Bayes and random forest is more interesting: Naïve Bayes and Decision Trees are a white box model, Random Forest is a black box model [1], but decision trees and random forest methods have much more common than Naïve Bayes in terms of its probabilistic approach to classification.

## Approach

The dataset is split into a training set and test set. The test set is kept aside for the final test. In this way it is possible to evaluate how good the trained model is at generalization. The dataset was normalized in the range 0 to 1, this did not make any difference in the performance. Using the z-score standardization [2] Naïve Bayes was found to train in a shorter time, this is different normalization algorithms have a strong effect on the sensitivity of the classification [2].

### Bootstrapping

Bootstrapping is used, this is recommended for large datasets [3]. When bootstrapping, careful consideration was given as to how the tree is created anew on each fold iteration: a separate random number generator is used to the one used for taking a random train and test set. In this way, for 5 folds, as an example, the model is created using the same random seed in order to ensure that it is the data that changes and not the model. Bootstrapping is a method that is used to reduce the variance for ensemble methods [4].

### Grid Search

The model selection process uses a hyperparameter search. The results for different parameter configurations are gathered in a csv file, and plotted. In this way it is possible to visually identify results that are good candidates for the final model. The visually identified results are looked up in the results csv file to gain a better understanding of the performance, hyperparameter values used and why the performance is either better or worse than other readings.

### Stopping Criteria

The stopping criteria selected is to stop when the performance results degrade noticeably. In this way it is possible to gain an understanding of how sensitive the model is to small changes in the hyperparameter values.

### Performance Metrics

The metrics used to evaluate the performance are train loss, test loss, accuracy, precision, recall, F1 [5], [6] time to train and predict, time to predict. For all the methods and models, it was found that accuracy, precision, recall, and F1 tend to move together for different hyperparameter values. Accuracy is 1 – training loss, both are kept as they are arrived at separately, in this way it is possible to confirm that there is no error in the calculations.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | Figure Train/Test Loss Split Criteria  Split criteria comparison. Deviance can be seen to consistently have a slightly lower loss than the other criteria. |
|  | Figure Train/Test Loss by row number in results table  The plot shows that training and test loss tend to move together. The results with lower loss are identified and can be looked up in the results table. | | |
|  | | Figure Result Metrics by results table row number  The plot shows that accuracy, precision, recall and F1 tend to be grouped together. The results with the highest metric values are identified and looked up in the results table. | |

By looking at the plots, the best performing rows of the results table are identified, and analyzed, to perform a grid search: disp(treeSelectedFeatureResultsNormalised.resultsTable([55, 110, 164, 166, 192, 199],:));

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | numberOfFolds | minLeafSize | minParentSize | maxNumSplit | splitCriterion | avgTrainLoss | avgTestLoss | | 5 | 5 | 10 | 8000 | gdi | 0.1021 | 0.1681 | | 2 | 1 | 6 | 8000 | twoing | 0.0459 | 0.1459 | | 2 | 1 | 6 | 4000 | deviance | 0.0388 | 0.1468 | | 5 | 1 | 6 | 4000 | deviance | 0.0405 | 0.1391 | | 3 | 1 | 6 | 8000 | deviance | 0.0397 | 0.1403 | | 5 | 1 | 10 | 8000 | deviance | 0.0682 | 0.1521 | |
| |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | avgAccuracy | avgPrecision | avgRecall | avgF1 | entryCount | elapsedTime | predictTime | randomSeed | | 0.833 | 0.8317 | 0.8333 | 0.8325 | 3200 | 2.0313 | 0.0031 | 54 | | 0.8542 | 0.8538 | 0.8543 | 0.8541 | 3200 | 0.9063 | 0.0859 | 109 | | 0.8527 | 0.8525 | 0.8525 | 0.8525 | 3200 | 0.625 | 0 | 163 | | 0.8612 | 0.8605 | 0.8617 | 0.8611 | 3200 | 1.7031 | 0.0031 | 165 | | 0.8596 | 0.8594 | 0.86 | 0.8597 | 3200 | 0.9844 | 0 | 191 | | 0.8476 | 0.8476 | 0.8487 | 0.8481 | 3200 | 1.5781 | 0.0031 | 198 | |

Training the tree using the hyperparameters with: minLeafSize = 1, maxParentSize = 6, maxNumSplit = 8000, random seed = 192, deviance split criterion; are selected for the final model.

### Confusion Matrix

|  |  |
| --- | --- |
|  | A confusion matrix is calculated on the final models. This is used to establish how the correct and incorrect classifications are distributed among the classes. Which classes tend to be most confused (I and J for example), which are the least confused (X, X and M for example), and also to establish whether any one class is poorly predicted. |

## Conclusion

Random forest is the best in terms of performance with 0.96 accuracy, decision trees came in second with 0.88 accuracy, and Naïve Bayes came in last with 0.75 accuracy. Different datasets can lead to different outcomes, but in general the random forest method performs better than Naïve Bayes and decision trees. This result is in line with the conclusion based on the comparison of 10 machine learning methods on 11 different datasets [7].

## Glossary

**Attribute, Example, Feature** an example is one sample of data that is composed of the different attributes or predictors which are used to make the prediction.

**Black Box Model** a black box model is not explainable, because the model is too complex to analyze [1].

**Boostrapping, Bagging , Fold** during training, a proportion of examples are randomly taken for training, and another set of examples are randomly taken for testing. This is repeated, using the same original set of data: the examples are said to be replaced, not removed for each subsequent iteration or fold.

**Cross validation** during training, a proportion of examples are randomly taken and removed for training, and another set of examples are randomly taken and removed for testing. This is repeated, the examples are said to be removed, not replaced for each subsequent iteration or fold.

**Dataset** a dataset is a group of examples that are used to train, test a model.

**Ensemble Method** train several models and decide on the class using the majority vote (for classification) or average prediction (for regression) of the models.

**Explainable** a machine learning is said to be explainable if it is possible to break down how the model arrived at a prediction for a given example [1].

**Generalization** the goal of training a model is that it is capable of performing well on data that is new. When a model is good at predicting unseen data, it manages to generalize what has been learnt from the training examples in order to correctly predict previously unseen examples.

**Grid Search** method used to search through a grid of results and hyperparameter values in order to select the row that has the best performance, and extract the hyperparameter values to use to achieve the best result.

**Hyperparameter** a parameter value that has to be supplied to the training, for example the number of folds.

**Information Gain** is a consideration that is used when selecting which features to leave out: if leaving them out does not affect the accuracy, then there is no information gained from keeping the feature [8]

**Iterative approach** as a project progresses, some of the artifacts of previous work is updated to be useful in the current work.

**Overfit** when training a model the training loss and test loss can start to diverge: the model has become too specific to the training data, and is no longer capable of generalizing.

**Stopping criteria** when to stop training with different hyperparameter values. Stopping can be chosen when overfitting occurs, or based on how long training takes, or other criteria that seem relevant.

**Test Set, validation set** refers to a set of examples that are used evaluate predictions on the model. It can be confusing as to which term is used during training testing and final testing.

**Training set** a training set is a set of examples that are used to create a model for a specific dataset.

**White Box Model** a white box model is explainable, it is possible to analyze how it deduces its predictions [1].

## References

[1] A. Thampi, *Interpretable AI*, MEAP v3., vol. 1, 1 vols. USA: Manning Publications, 2020.

[2] N. L. Pollesch and V. H. Dale, “Normalization in sustainability assessment: Methods and implications,” *Ecol. Econ.*, vol. 130, pp. 195–208, Oct. 2016, doi: 10.1016/j.ecolecon.2016.06.018.

[3] T. M. Khoshgoftaar, M. Golawala, and J. V. Hulse, “An Empirical Study of Learning from Imbalanced Data Using Random Forest,” p. 8.

[4] G. Valentini, “Ensemble methods based on bias–variance analysis,” p. 175.

[5] B. Shmueli, “Multi-Class Metrics Made Simple, Part I: Precision and Recall,” Jul. 02, 2019. https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2.

[6] B. Shmueli, “Multi-Class Metrics Made Simple, Part II: the F1-score,” *Multi-Class Metrics Made Simple, Part II: the F1-score*. .

[7] R. Caruana and A. Niculescu-Mizil, “An empirical comparison of supervised learning algorithms,” in *Proceedings of the 23rd international conference on Machine learning - ICML ’06*, Pittsburgh, Pennsylvania, 2006, pp. 161–168, doi: 10.1145/1143844.1143865.

[8] I. Guyon and A. Elisseeff, “An Introduction to Variable and Feature Selection,” *J. Mach. Learn. Res.*, vol. 3, p. 26, 2003.