L42: Machine Learning and Algorithms for Data Mining

Assignment 2 – Topic: classification methodology for arthritis patients

Philippos Papaphilippou (pp417), Wolfson College
Lent Term 2017

1. Introduction

The use of machine learning techniques in medicine has the ability to revolutionise the field by accurately predicting future events of vital importance from existing data. A lot of research work is aiming to improve the automatisation of diagnosis routines using data mining and machine learning approaches such as random forests for classification [6].

Rheumatoid arthritis is a very disabling form of arthritis and an early diagnosis from gene signatures would be ideal for developing new therapeutic procedures [3]. Osteoarthritis is also a common form of arthritis with less pronounced impact on the synovial function, but they require discrimination, such as with radiography, as they are very similar in general and sometimes require different treatment, especially at the later stages [1]. Other advanced methods have been proposed for diagnosis, such as with ultrasound-based scoring systems [1], but machine learning is very promising because the symptoms of rheumatoid arthritis are shown to be closely correlated to different meta-genes and such analysis could lead to a more targeted therapy [3].

Is this project, a random forest application is developed in order to be able to classify potential arthritis patients. The goal is to assign a label of either "rheumatoid arthritis", "osteoarthritis" or "healthy" for each person in the test data. It is important to note that there are many approaches for performing this task and some of them will be described in a following section. Finally, I present some experimental results with which the application's performance is evaluated for different parameters relevant to the random forest and the available dataset.

2. Classification of arthritis patients

The main dataset for this experimentation is described in this work [1]. It combines the data of 79 patients and donors (for the control group) from 3 clinical experiments done in Jena, Berlin, and Leipzig. Out of the 79 people, 20 are labeled as "Control group", 26 as "Osteoarthritis" patients and 33 as "Rheumatoid arthritis" patients. These labels will be used to train our rain forest estimator to be able to classify new persons with this labels. The data for each person has the form of a microarray, containing information about 22283 features. Each feature is described by a floating point number and represent the expression level of a single gene.

The study [1] focused on creating simple but powerful rulesets from the data using a commercially available software specifically developed for disease biology. One of the main contributions was the explicit description of the most important generated rules for classifying the patients. In addition there is analysis of the quality of the dataset as they describe how easy is discrimination for each of the 3 subsets of the dataset, as well as identifying duplicate parameters. These findings are more targeted towards genetic scientists rather than computer scientists, but the dataset itself is very appropriate for further analysis using traditional machine learning methods.

The datasets, as well as the research work is publicly available. The Gene Expression Omnibus is an online repository of such microarray data and also provides an R-based visualisation platform [5]. The interesting part of the selected dataset is that the classification is multiclass which renders the analysis and patient discrimination slightly more challenging than when having to perform binary classification.

2.1 Alternative approaches in literature

As mentioned above, the study that combined the 3 datasets [1] used a software specifically designed for genomics, Pathway studio (Elsevier, Munich, Germany) . Pathway studio is using a comprehensive database of already discovered rules and therefore may not be considered a pure machine learning approach, but a collaborative work of many sciences [10]. The core algorithm might also not be available publicly to reveal such approaches.

This work [2] also conducted a classification study on patient microarrays. The goal was to iprove breast cancer detection. It combined random forests with k-means clustering and other optimisations for filtering the features and observations. The difference with this project is that it was a goal to also eliminate the training time of the platform.

In general, there is a lot of reseach relevant to applying random forests on microarrays, such as the Convergent Random Forest method, for identifying highly predictive biomarkers [11], but there can also be applied and evaluated many other classification tools. Examples of such classification methods also include different clustering, multi-layer perceptron artificial neural networks and support vector machines, all of which can optimal to different applications.

3. The Random Forest algorithm

Random forests [6] is an ensemble machine learning method that is used for regression and classification. A random forest consists of m decision trees that are generated during training.

3.1 Decision Trees (for classification)

A decision tree classifier is a simpler approach for classification that is more prone to overfitting than random forests. However, random forests are considered an ensemble of decision trees because the algorithm uses a voting system to estimate the best answer from many decision tree estimators.

After a classification tree is properly trained, the algorithm used for classification is simple and when the depth of the tree is low enough, it can be easily followed by a human. Starting from the root node we follow the child node that corresponds to the condition for a specific feature in our data record. When we reach a leaf node we assign to our record the label of the class that is denoted to have the highest probability.

The algorithm for the generation of a decision tree is recursive. For each node there are N data records that are to be split into M subsets, corresponding to M child nodes, based on a condition. Each data point has the form (x_1, x_2, …, x_n, C), where x_i is the value of the feature i and C is the class which the record belongs to. A common practice is to use binary splits. To create the condition the algorithm searches for the feature and split value that will maximize the information gain.

3.2 Information Gain

The information gain is a metric that is used to determine the best split of the data set. In order to calculate the information gain we need the impurity metric, which denotes how impure is a dataset. The information gain for our case is calculated as follows

$$Information\,Gain = Impurity_{Before} - Impurity_{After} = Impurity_{Before} - \sum_{i}^{N} \frac{|S_i|}{|S_{Before}|} Impurity_{S_i} \quad ,$$

where N is the number of subsets that occur after the split and Impurity is a function that can be defined by one of the following examples.

Common measures of Impurity are [7]: $Entropy = \sum_j -p_j \log p_j$, $Gini\ Index = 1 - \sum_j p_j^2$ and $Classification\ Error = 1 - max\{p_j\}$, where $p_j = \frac{1}{|S|} \sum \{x \in S | C_x = j\}$ is the proportion of class k observations in the respective dataset S.

3.3 The algorithm [4]

For each of the m iterations, create a bootstrap T′ of the dataset T and use it to train a decision tree of maximum k random features. Then, the tree is added to the random forest structure. In regards to testing a single record, we get the predicted class from each of the decision tree estimators and the most frequently voted class returned by the trees.

4. Practical Evaluation

For the purposes of evaluating random forests on the provided dataset I have developed a python application that parses the data of the microarray form and evaluates a random forest by using either the Out-Of-Bag error metric or k-folds Cross-Validation. The software supports all 3 impurity measures that are mentioned here – Gini index, Entropy and Misclassification – and a variable bootsrap size, as well as the option to remove the replacement option when sampling for bootstrap (it is not called bootstrap when sampling without replacement).

It consists of 3 python scripts:

- *DatasetLoader.py* – The initial script to produce the binary file of the dataset. By default, it reads the microarray files "GDS5401_full.soft", "GDS5402_full.soft" and "GDS5403_full.soft", but it can be easily modifies to read other datasets.
- *Classifier.py* – The main file that implements all the random forest variations. After the binary file of the dataset "dataset_dump.dat" is produced by the above script, the random fores is called by the command `$python Classifier.py <m> <k> <Impurity measure>`, where m is the number of trees in the ensemble, k the maximum depth of each tree (or size of subset of features for each tree) and impurity measure an integer from 0 to 2, 0 for Gini, 1 for Entropy, 2 for Misclassification.
- *Threaded.py* – A python threadpool to run many experiments in parallel for a multicore systems using N processes.
- (Attached are also included the gnuplot scripts that have been used for plotting the figures in this report)

4.1 Evaluation Metrics

4.1.1 Out-Of-Bag Error

The default metric suggested by Breiman [6] is the Out-Of-Bag Error, which is the average of the missclassication records of all patients among the trees who did not include each of them in their respective bootsrap. The bootstrap part of the random forest algorithm is providing an unbiased method to calculate error, eliminating the need to perform an additional time-expensive cross-validation.

In Figure 1, I present the OOB error rates for different number of trees and features. The most important observation is that the number of both trees and features is crucial for the performance of the random forest but after around k=100 there is little gain from increasing the parameters, at least for our data. In the 3D plot there are also noted the standard deviation of each point with green lines across the series of each m value. It is interesting that the variance is greater when m is small and this is because for each observation we get an error representing around m/3 trees [8]. The OOB rate can give significant insight for selecting a practical and effective value of k and m.

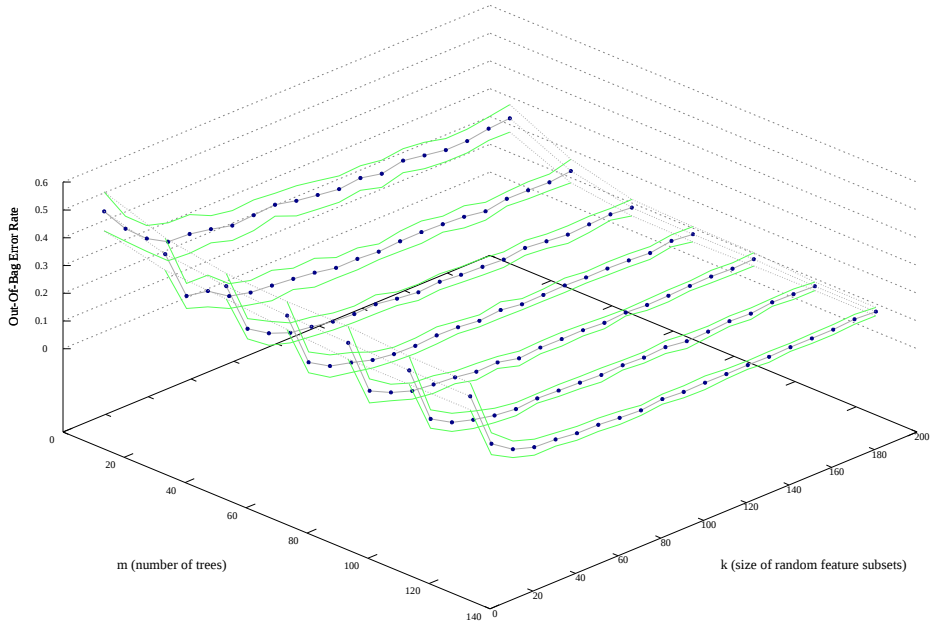

Figure 1: Out-Of-Bag error for different parameter combinations of the random forest.
Each point represents the average value of 30 runs and the green lines the standard deviations of each datapoint.

In our case the dataset size is only 79 patients and when selecting the 66% (with replacement falls down to 63%[9]) the decision trees might be trained using little information. I also wanted to use a more common metric for comparing with other techniques in the future. For those reasons I have selected k-Folds Cross-Validation with k=10 for the evaluation in the following sections, although the difference might be negligible.

4.1.2 K-Folds Cross-Validation

K-Folds Cross-Validation is another measure to estimate the accuracy of a predictive model. It is used for validating a wide range of models, such as with neural networks. In order to perform cross-validation, we select a k, typically ranging from 3 to 10, and split the dataset into k equally sized subsets. For each of the k regions, we calculate the error of the region as a test set after we train the model using the records from all other regions. In the end, the true error is obtained by averaging all k errors.

In Figure 2, we see the confusion matrices of a 5-fold cross-validation on our model. The confusion matrices are used for identifying weaknesses in a model, such as confusing osteoarthritis patients for rheumatoid arthritis patients. One observation could be that in many of the folds some osteoarthritis patients are confused for healthy persons, although Figure 2 is only provided as an example for visualization, as in my evaluation I use k=10 and the true error is averaged again from multiple runs.

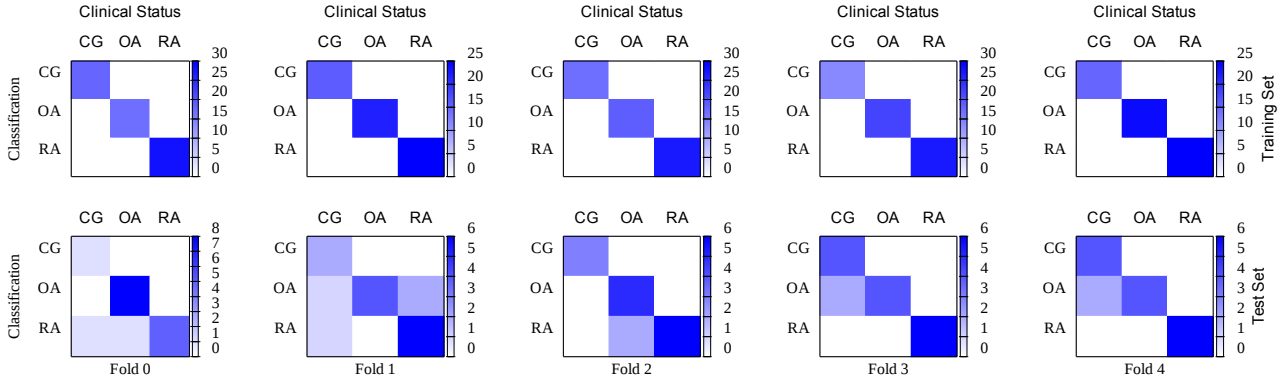

Figure 2: Confusion matrices for each of the folds of a single 5-fold Cross-Validation. Forest parameters: 65 trees of 35 features.
CG, OA and RA represent the “Control Group”, “Osteoarthritis” and “Rheumatoid arthritis” flags that are assigned by the classifier.

4.2 Optimal parameter value selection

In this section I use my preferred evaluation metric for selecting the optimal parameters for the random forest classifier. Each point in the following graphs is the average of 50 iterations of the Cross-Validation for more representative values.

In Figure 3 we can see the cross-validation version of the Figure 1. We compare the number of trees against the number of features in respect to performance. We can clearly observe that increasing the number of trees has a significant effect on performance. The optimal values for m and k seem to lay in the darker area of the graph of a true error close to 0.1. An example of a good combination of values laying in the darker area could be m = 100 and k=50, since k is generally more expensive than m in terms of computation complexity.

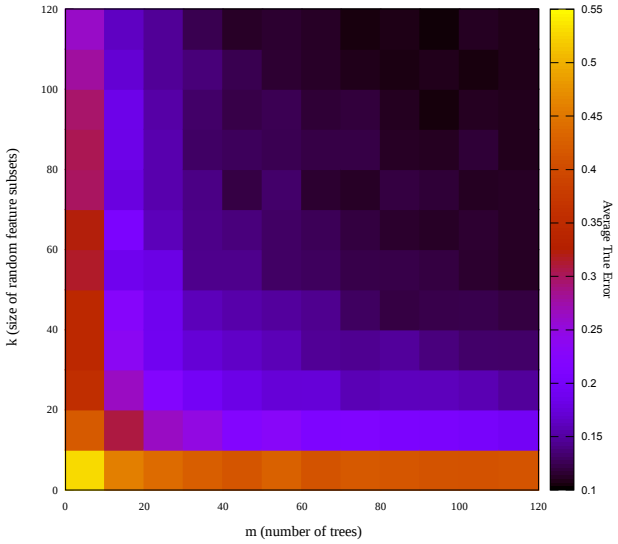

Figure 3: Heatmap graph for comparing how the number of trees and features affect the random forest performance. The graph is not interpolated, each square represents a separate experiment (average of 40 runs).

An interesting experimentation is the comparison of the 3 impurity measures used in the decision tree generation phase as well as the bootstrap size and to evaluate the replacement aspect of the bootstrap technique. In Figure 4 I explore many combinations of the above parameters in a single plot.

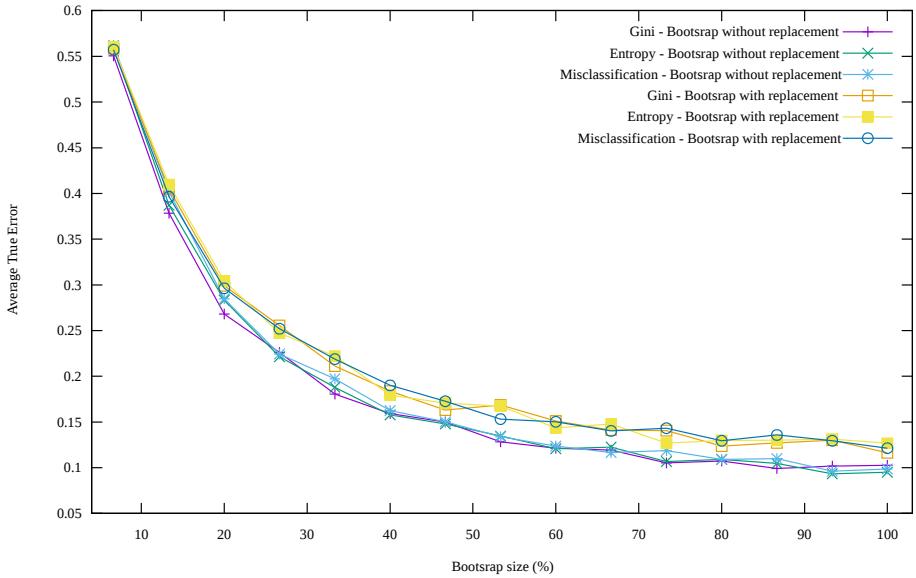

Figure 4: Comparison of the 3 impurity measures, different bootstrap sized and evaluation of the replacement aspect of the bootstrap. Steady random forest parameters: 65 trees of 35 features.

One observation is that the 2/3 default bootstrap size [6] is a good choice as it performs near optimally for all the series in the plot, if we also take into account that the bootstrap size also affects performance. All impurity measures seem to perform similarly but the replacement seems to worsens performance for all cases. This observations, however, are for our relatively small dataset and a different in could yield different conclusions. I would say that a bootstrap of 80% the original dataset size and the misclassification impurity measure would be a good selection of parameters.

5. Conclusion

It is possible to construct an experimentation such as the above to improve the selection of the parameters. I provide a set of values for a good selection of number of trees and number of features for each tree. In addition I provide a comparison of some secondary parameter values the 3 impurity measures, the bootstrap size and the bootsrap replacement. The three impurity measures are shown to perform similarly and the replacement in bootstrap was not favourable in our dataset. In addition, the 2/3 selection of bootsrap size is validated to be reasonable. Cross-validation true error is evaluated to be very close to OOB error rate. Finally, a general conclusion is that overfitting was indeed not observed with random forests here.

~2400 Words

References

[1] Woetzel, Dirk, Rene Huber, Peter Kupfer, Dirk Pohlers, Michael Pfaff, Dominik Driesch, Thomas Häupl et al. I*dentification of rheumatoid arthritis and osteoarthritis patients by transcriptome-based rule set generation.* Arthritis research & therapy 16, no. 2 (2014): R84.

[2] Griffith, Obi L., François Pepin, Oana M. Enache, Laura M. Heiser, Eric A. Collisson, Paul T. Spellman, and Joe W. Gray. *A robust prognostic signature for hormone-positive node-negative breast cancer.* Genome medicine 5, no. 10 (2013): 92.

[3] Afroz, Sumbul, Jeevan Giddaluru, Sandeep Vishwakarma, Saima Naz, Aleem Ahmed Khan, and Nooruddin Khan. *A comprehensive gene expression meta-analysis identifies novel immune signatures in Rheumatoid Arthritis patients.* Frontiers in Immunology 8 (2017).

[4] Sazonau, Viachaslau. *Implementation and Evaluation of a Random Forest Machine Learning Algorithm* University of Manchenster (2012): 9.

[5] Barrett T, Wilhite SE, Ledoux P, et al. *NCBI GEO: archive for functional genomics data sets*—update. Nucleic Acids Research. 2013;41(Database issue):D991-D995. doi:10.1093/nar/gks1193.

[6] Breiman, Leo. *Random forests.* Machine learning 45, no. 1 (2001): 5-32.

[7] Teknomo, Kardi. (2009) *Tutorial on Decision Tree.* people.revoledu.com/kardi/tutorial/DecisionTree/

[8] James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning.* Vol. 6. New York: springer, 2013.

[9] Breiman, Leo. "Out-of-bag estimation." (1996).

[10] Nikitin, Alexander, Sergei Egorov, Nikolai Daraselia, and Ilya Mazo. P*athway studio—the analysis and navigation of molecular networks.* Bioinformatics 19, no. 16 (2003): 2155-2157.

[11] Bienkowska, Jadwiga R., Gul S. Dalgin, Franak Batliwalla, Normand Allaire, Ronenn Roubenoff, Peter K. Gregersen, and John P. Carulli. *Convergent Random Forest predictor: methodology for predicting drug response from genome-scale data applied to anti-TNF response*. Genomics 94, no. 6 (2009): 423-432.