Does Feature Selection Improve Classification? A Large Scale Experiment in OpenML

Martijn J. Post, Peter van der Putten, and Jan N. van Rijn

Leiden University, The Netherlands
m.j.post@umail.leidenuniv.nl
{p.w.h.van.der.putten,j.n.van.rijn}@liacs.leidenuniv.nl

Abstract. It is often claimed that data pre-processing is an important factor contributing towards the performance of classification algorithms. In this paper we investigate feature selection, a common data pre-processing technique. We conduct a large scale experiment and present results on what algorithms and data sets benefit from this technique. Using meta-learning we can find out for which combinations this is the case. To complement a large set of meta-features, we introduce the Feature Selection Landmarkers, which prove useful for this task. All our experimental results are made publicly available on OpenML.

Keywords: Feature Selection, Meta-Learning, Open Science

1 Introduction

Feature selection can be of value to classification for a variety of reasons. Real world data sets can be rife with irrelevant features, especially if the data was not gathered specifically for the classification task at hand. For instance in many business applications hundreds of customer attributes may have been captured in some central data store, whilst only later is decided what kind of models actually need to be built [14]. Bag of words text classification data will by definition include large numbers of terms that may end up not to be relevant. Micro-array data sets consisting of genetic expression profiles are very wide data sets, whilst the number of instances is typically very small. In general, feature selection may help in terms of making models more interpretable, ensuring that models actually generalize rather than overfit and it will speed up the building of models when costly algorithms are being used. Highly cited surveys exist that provide a more theoretical overview of feature selection [1,6], however classical empirical papers on feature selection are typically based on small numbers of data sets (for example, 3 data sets in [5] and 14 data sets in [10]).

In this paper we investigate the specific question: will feature selection improve binary scoring models for a given data set and algorithm. We base our findings on experiments across a large number of data sets (almost 400) and a range of algorithms, and for repeatability all results have been made available in OpenML, an open science experiment database [20]. This results in a meta-data set that we leverage to learn in what circumstances feature selection may provide

better classifications for a given data set algorithm combination. We introduce a number of new meta-features to characterize data sets and algorithms for this purpose.

Our contributions are the following. We conduct two large scale experiments ranging over almost 400 data sets. The first experiment investigates for which algorithms feature selection generally improves predictive performance. This experiment both confirmed well-established conjectures and raised some interesting new findings. The second experiment exploits meta-learning to understand for which data sets feature selection may improve results. We introduce new meta-features, specific to this problem. All our underlying experimental results as well as the meta-data set are made publicly available, for the purposes of verifiability, reproducibility and generalizability.

The remainder of this paper is structured as follows. We will introduce some background in feature selection and meta-learning (Section 2) as well some additional meta-features that prove useful (Section 3). We will then review the overall experiments and results in terms of when feature selection may add value (Section 4), and a meta-learning experiment where we aim to predict whether to use feature selection for a given data set (Section 5). Section 6 concludes the paper.

2 Background

In this section we discuss relevant background and related work in feature selection, meta-learning and experiment databases.

2.1 Feature selection

As discussed in the introduction feature selection can serve a number of purposes, such as improved interpretation, generalization and learning speed. The merits of and methods for feature selection are discussed extensively in a number of classical survey papers, hence we will keep the overview brief here [1,4,5,6,10]. The goal of feature selection can be to find the optimal set of features that maximizes a given objective, and hence can be seen as a search problem with a given search method, evaluation metric and overall objective, typically some form of predictive power.

Exhaustive search is typically not feasible so different approaches are needed. A simplistic approach would simply select the top features based on predictive power. This is sub optimal, because features may be correlated to features already selected, so not adding much information, or conversely, weak features could jointly actually be predictive, thus subset feature selection rather than rankers are required [6,8]. The evaluation metrics could be so called filter metrics, such as correlation, mutual information or information gain, independent of the classification algorithm used. Alternatively, models could be trained on subsets of features in a so called wrapper approach, which can be valuable if the subsequent learners have very specific biases or limitations [10]. Wrappers do not necessarily perform better than filters [19] so in our work we have focused

on a subset filter approach [8]. Feature construction or dimension reduction can be seen as an extension of feature selection, but this is out of scope for this paper. Note that classification methods can also have some embedded element of feature selection built in, but as we will see this is no guarantee that feature selection is no longer required.

2.2 Meta-learning

Meta-learning aims to learn which learning techniques work well on what data. A common task, known as the Algorithm Selection Problem [17], is to determine which classifier performs best on a given data set. We can predict this by training a meta-model on data describing the performance of different methods on different data sets, characterized by *meta-features* [2,11,13]. Meta-features are often categorized as either simple (number of examples, number of attributes), statistical (mean standard deviation of attributes, mean skewness of attributes), information theoretic (class entropy, mean mutual information) or landmarkers [12] (performance evaluations of simple classifiers). Alternatively, performance estimates of algorithms on small subsets of the data set can be used [18].

Experiment databases enable the reproduction of earlier results for verification and reusability purposes, and make much larger studies (covering more classifiers and parameter settings) feasible. Above all, experiment databases allow a variety of studies to be executed by a database look-up, rather than setting up new experiments. An example of such an online experiment database is OpenML [20]. All data sets and experimental results used in this work are made publicly available in OpenML. Similar collaborative platforms exist in the commercial domain, such as Kaggle [3], but these typically lack the ability to store and search low level results in a structured manner.

3 Methods

The field of meta-learning addresses the question what machine learning algorithms work well on what data. The algorithm selection problem, formalised by Rice in [17], is a natural problem from the field of meta-learning. According to the definition of Rice, the problem space P consists of all machine learning tasks from a certain domain, the feature space F contains measurable characteristics calculated upon this data (called meta-features), the algorithm space A is the set of all considered algorithms that can execute these tasks and the performance space Y represents the mapping of these algorithms to a set of performance measures. The task is for any given $x \in P$, to select the algorithm $\alpha \in A$ that maximizes a predefined performance measure $y \in Y$, which is a classification problem. Typically, this problem is addressed by creating a meta-data set. Each example represents an experiment where all algorithms in A are run on a data set from P, the meta-features are measurable characteristics of this data set and the target is the best performing algorithm on this data set. A classifier can then learn to predict for new data sets which algorithm will perform best [22].

Table 1. Standard Meta-features.

Category	Meta-features
Simple	# Instances, # Attributes, Dimensionality, Default Accuracy, # Observations with Missing Values, # Missing Values, % Observations With Missing Values, % Missing Values, # Numeric Attributes, # Nominal Attributes, # Binary Attributes, Majority Class Size, % Majority Class
Statistical	Mean of Means of Numeric Attributes, Mean Standard Deviation of Numeric Attributes, Mean Kurtosis of Numeric Attributes, Mean Skewness of Numeric Attributes
Information Theoretic	Class Entropy, Mean Attribute Entropy, Mean Mutual Information, Equivalent Number Of Attributes, Noise to Signal Ratio
Landmarkers [12]	Accuracy of Decision Stump, Kappa of Decision Stump, Area under the ROC Curve of Decision Stump, Accuracy of Naive Bayes, Kappa of of Naive Bayes, Area under the ROC Curve of Naive Bayes, Accuracy of k-NN, Kappa of k-NN, Area under the ROC Curve of k-NN

In this work we address the following problem. Given a data set and an algorithm, should we use feature selection or not? We aim to solve this in a similar manner. We construct a meta-data set, where each example represents the combination of data set d and algorithm α. The features are measurable characteristics of data set d, and the target is whether the performance of algorithm α is (significantly) better after performing feature selection than without it.

The performance of meta-learning solution typically depends on the quality of the meta-features. Typical meta-features are often categorized as either simple, statistical, information theoretic or landmarkers. The simple meta-features can all be calculated by one single pass over all instances and describe the data set in an aggregated manner. The statistical meta-features are calculated by considering a statistical concept (e.g., standard deviation, skewness or kurtosis), calculate this for all numeric attributes and taking the mean of this. This leads to, e.g., the mean standard deviation of numeric attributes. Likewise, the information theoretic meta-features are calculated by considering a information theoretic concept (e.g., mutual information or attribute entropy), calculate this for all nominal attributes and taking the mean of this. This leads to, e.g., mean mutual information. Landmarkers are performance evaluations of fast classifiers on a data set, characterising the complexity landscape and bias of various learners. Table 1 shows all traditional meta-features used in the experiments.

Landmarkers are generally considered the most expensive meta-features (in terms of resources), as well as the most useful (in terms of predictive power). Although this might be true for the algorithm selection problem, there are reasons to suspect that this might be different for the task of determining whether or not to perform feature selection. First, many feature selection methods operate on statistical and information theoretical concepts. Second, information about the learning bias of various classifiers seems less relevant, as we try to obtain information about one algorithm at a time.

For this reason, we introduce specific *feature selection landmarkers*. We run a simple (fast) classifier with and without feature selection. By subtracting one from the other, we can see what the effect of feature selection was when using a fast algorithm. Similar to regular landmarkers, we assume that this effect translates to the results of more expensive algorithms as well.

Table 2. Algorithms used in the experiments. All algorithms are as implemented in Weka 3.7.13 [7] run with default parameter settings, unless stated different.

Algorithm	Model type	Parameter settings
Naive Bayes	Bayesian	
IBk	k-NN	$k = 1$
Stochastic Gradient Descent (SGD)	SVM	
Sequential Minimal Optimization (SMO)	SVM	Polynomial kernel
Logistic	Logistic	ridge = 0.00000001
Multilayer Perceptron	Neural Network	1 hidden layer
JRip	Rules	
J48	Decision Tree	
Hoeffding Tree	Decision Tree	
REP Tree	Decision Tree	
RandomForest	Bagging	100 trees
AdaBoost	Boosting	100 iterations

4 Effect of Feature Selection

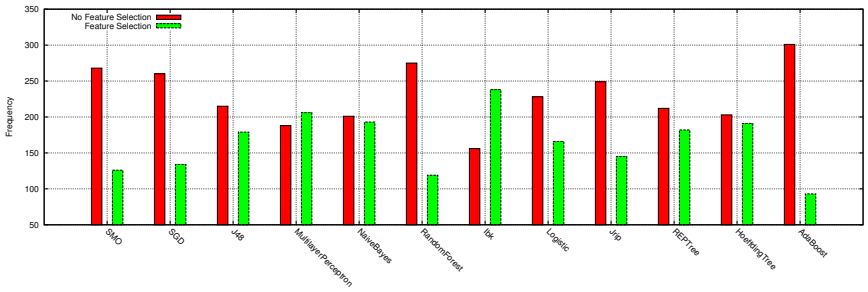
In this section we will present some explorative results, surveying per algorithm how often feature selection is beneficial and how large the effects are. All data sets, algorithm and experimental results can be obtained from OpenML[1] [20]. Figure 3 also gives some basic insight in the number of features and the dimensionality of the data sets.

4.1 Experiment

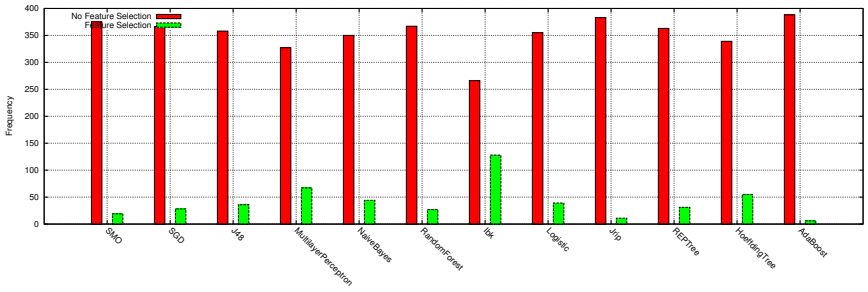
All algorithms are evaluated over the data sets using 10-fold cross-validation, with and without feature selection. We measure the difference in Area under the ROC Curve (AUC) for each algorithm with and without feature selection. We prefer AUC over zero one loss accuracy as an evaluation criterion for a variety of reasons. First, if the outcome class distribution is very skewed, a simple majority vote may achieve very high accuracy, whereas in practice this may not be very useful model. Second, false positives and false negative classifications may come at a different cost, but these costs are not known, hence it makes sense to evaluate model performance across the entire model score range.

For feature selection, the Correlation-based Feature Subset Selection (CfsSubsetEval) algorithm is used [8]. We experimented with other feature selection methods as well (i.e., GainRatio and InfoGain) but as the differences in performance were too marginal and subset feature selection is generally considered to be a better approach we stick to CfsSubsetEval.

The data sets that are used in the experiments are all data sets containing between 10 and 200,000 instances. As we are focusing on Area under the ROC Curve, we selected data sets with a binary target. In total 394 data sets from OpenML matched these criteria. Table 2 shows the algorithms that were used and their parameter settings.



(a) Number of data sets where 'no feature selection' obtained better results (red) and 'feature selection' obtained better results (green)



(b) Number of data sets on which 'feature selection' was statistically significant better (green) and not statistically significant better (red)

Fig. 1. Number of data sets on which feature selection improves performance.

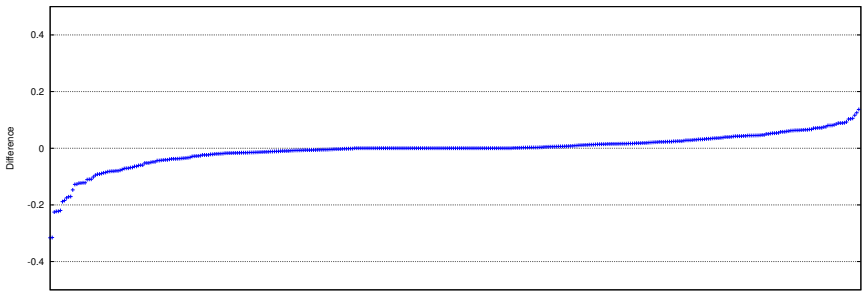
4.2 Results

Figure 1(a) shows for each algorithm in how many cases feature selection yields better results. Figure 1(b) shown for each algorithm in how many cases this difference was also statistically significant (using a double tailed T-test of 0.05).

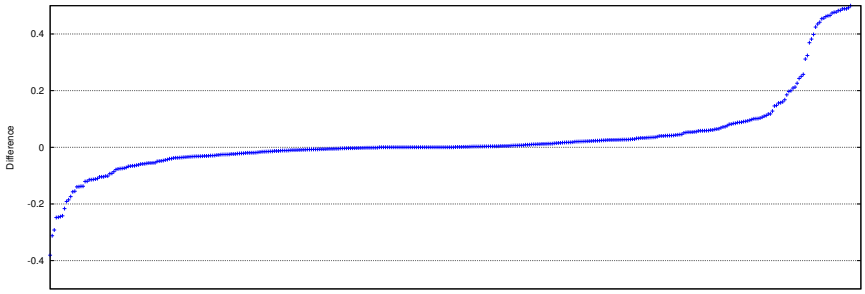
We can also focus on how big the effect of feature selection per data set is. In Figure 2 we plotted for some algorithms the difference in performance with and without feature selection. The x-axis represents the various data sets, the y-axes the difference in performance (AUC). The x-axis is sorted on this effect, so we can see the big trends. For every dot above 0, using feature selection yields better results than not using feature selection.

In Figure 1(a) it is observed that no feature selection is slightly better for every algorithm, except for IBk and Multilayer Perceptron. J48 is noteworthy because it is expected that a tree partitioning algorithm has feature selection embedded. Controversially, the figure shows that feature selection can still add value for many data sets (see also Figure 2(a)). For the Multilayer Perceptron,

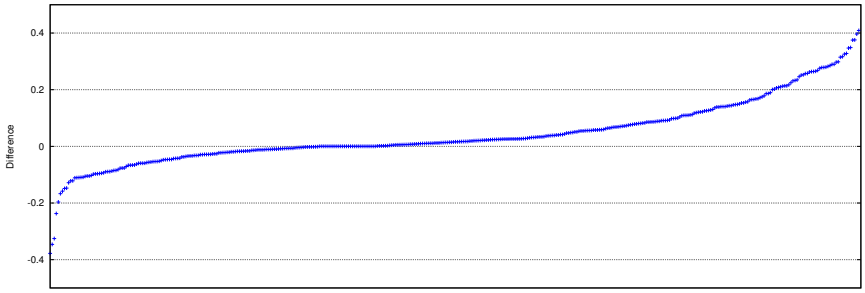
[1] Full details: `http://www.openml.org/s/15`



(a) J48



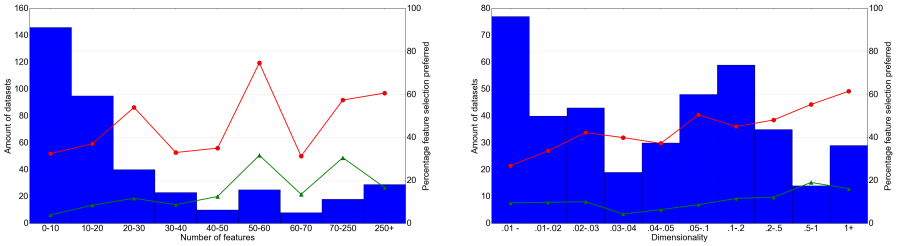
(b) Multilayer Perceptron



(c) IBk

Fig. 2. The effect of feature selection per data set on the x-axis for a given algorithm, sorted by difference in Area under the ROC Curve as the y-axis. When the difference is positive, the algorithm performed better after feature selection.

Naive Bayes and Hoeffding Tree, in about half of the cases feature selection improves the performance. Figure 1(b) shows that applying feature selection seldom results in a performance gain that is statistically significant. The IBk and Multilayer Perceptron algorithms have the highest amount of data sets where



(a) Number of Features (b) Dimensionality

Fig. 3. The amount of data sets (blue bar) in some ranges of two meta-features, with the red line meaning the percentage that feature selection was better in that range and the green line where the improvement was also statistical significant.

the benefits of feature selection are statistically significant, and behind these is Hoeffding Tree with just over 50 data sets.

Figure 3 shows a univariate analysis on how the amount of features and the dimensionality affect the probability that feature selection improves classification. Although a higher number of features results in a slightly higher percentage of data sets that benefit from feature selection, no clear distinction can be made with just one feature. A similar observation can be made for the dimensionality. Later, we will see that meta-models leveraging multiple meta-features, also highly depend on the number of features in a data set.

4.3 Discussion

The previous experiment shows both some expected behaviour as well as some interesting patterns. First of all, from Figure 1(a) we can see that feature selection is most beneficial for methods as IBk and Naive Bayes (reflected by Figure 1 and 2(c)). This is exactly what we would expect: due to the curse of dimensionality, nearest neighbour methods can suffer from too many attributes [16] and Naive Bayes is vulnerable to correlated features [9]. We also see unexpected behaviour. For example, it has been noted that tree-based algorithms such J48 have built-in protection against irrelevant features [15], however it can be observed from Figure 2(a) that still in many cases it appears to benefit from feature selection. Multilayer Perceptrons are also supposed to learn themselves which features are relevant [21], however Figure 2(b) shows that in many cases feature selection makes a substantial difference for the better. This delta to the right of the curve is higher than the delta on the left.

In general, feature selection seems to pay of for certain data sets, but the effect is not often statistically significant. A possible explanation could be that the data sets from OpenML are all Machine Learning data sets, where most features have been already carefully selected by domain experts. Feature selection would possibly yield more effect on raw data from production environments.

5 Learning when to use Feature Selection

In this section we investigate whether we can learn when to use feature selection, which is a novel form of meta-learning.

5.1 Experiment

We want to use meta-learning to predict for a given data set and algorithm whether feature selection will improve the Area under the ROC Curve score. Every instance in the meta-data set are two 10-fold cross validation runs on a algorithm, one run with and one run without feature selection, and the target is whether the run with feature selection had a better performance. The attributes are all the meta-features as mentioned in Section 3, for example the number of features and the percentage of numeric features, together with attributes about the algorithm. As meta-algorithm, we use Weka's Random Forest (100 trees).

In order to assess whether our proposed meta-features add any predictive value, we run the experiment with various sets of meta-features. The *simple* set contains just the simple meta-features (see Table 1) totalling to 13 features. The *no landmarkers* set contains all simple, statistical and information theoretic meta-features, (i.e., all meta-features from Table 1 except the landmarkers) which are in total 22 features. The *default landmarkers* set contains all meta-features from the no landmarkers set, and the traditionally described landmarkers (i.e., all meta-features from Table 1) which give a total amount of 31 features. The *Feature Selection Landmarkers* set contains all meta-features from the no landmarkers set, and the newly created Feature Selection Landmarkers as described in Section 3 thus also 31 features in total. The *All Landmarkers* set is the union of all previous sets totalling up to 40 features.

The data set can also be split in various subsets containing the results of only one algorithm. For example, we can investigate whether we can learn for a given algorithm whether to use feature selection or not.

The main motivation for using meta-learning here is primarily to obtain a further understanding of when feature selection may or may not add value, across multiple dimensions, to complement the analysis in the previous section that mainly focused on the algorithms used. A meta-model could be used in practice to assess beforehand whether performance may be improved in general or for specific algorithms, for example algorithms which are very costly to run. If exhaustive search is possible and reliable (i.e., run all algorithms for all parameters) it may still be preferred over using meta-learning.

5.2 Results

The results are shown in Table 3. Each row represents a partition of the data set, i.e., how well we could predict for each classifier whether we should use feature selection.

First, from this we conclude that meta-learning can answer the question whether to use feature selection or not. Compared to just predicting majority

Table 3. Area under the ROC Curve scores for various sets of meta-features on different partitions of the meta-data set.

Partition	Simple	No LM	Default LM	FS LM	All LM
J48	0.705	0.703	0.737	0.733	0.731
IBk	0.680	0.700	0.750	0.768	0.783
Multilayer Perceptron	0.734	0.704	0.708	0.711	0.710
Logistic	0.623	0.625	0.711	0.676	0.695
SMO	0.642	0.632	0.695	0.713	0.704
SGD	0.705	0.698	0.736	0.746	0.733
Hoeffding Tree	0.612	0.617	0.679	0.647	0.670
REP Tree	0.593	0.573	0.614	0.591	0.621
Naive Bayes	0.620	0.660	0.714	0.708	0.721
JRip	0.590	0.581	0.595	0.616	0.639
AdaBoost	0.623	0.634	0.638	0.649	0.668
RandomForest	0.712	0.722	0.764	0.774	0.784
Total data set	0.704	0.728	0.765	0.768	0.773

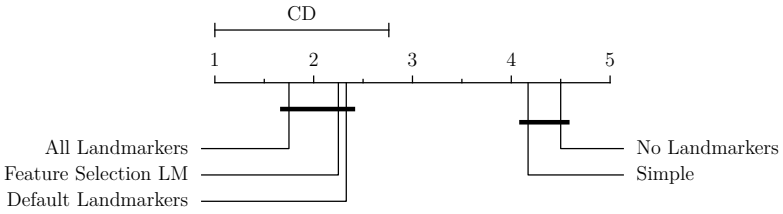


Fig. 4. Results of Nemenyi test. Sets of meta-features are sorted by their average rank (lower is better). Classifiers that are connected by a horizontal line are statistically equivalent.

class (which always has an Area under the ROC Curve of 0.5), we score better on all defined tasks, even with just a set of simple meta-features. Second, we observe that using just the two sets without landmarkers are clearly worse than the sets that use landmarkers. Finally, it appears that the set of default landmarkers and the newly created feature selection landmarkers perform similar. However, putting them together is beneficial. Figure 4 shows the result of a statistical test. This adds to the empirical evidence that the meta-classifier benefits from the landmarkers. However, there is no statistical evidence that one set is better than another. One interesting observation is that the set of meta-features without landmarkers performs worse than the set of simple meta-features. However, the difference is not statically significant.

As an example of deeper inspection of meta-models, Figure 5 shows a decision tree that determines when to use feature selection in combination with a Multilayer Perceptron. It splits on meta-features the Number of Attributes, Class Entropy and twice on a Feature Selection Landmarker, suggesting that these are important features. The interplay of these features is interesting. For example if the number of features exceeds 48, feature selection will be useful, if the number of features is smaller than 9 than not, and otherwise it depends on the interplay between the feature selection landmarkers, class entropy and number of features.

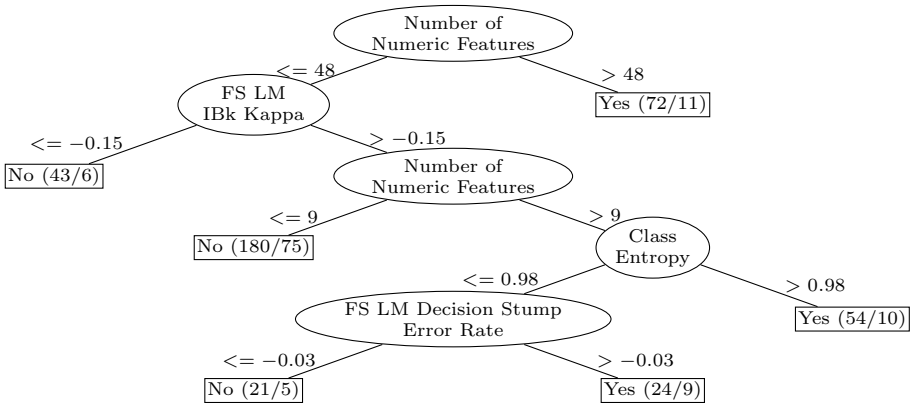


Fig. 5. Decision tree determining whether to use feature selection with a Multilayer Perceptron. Each leaf node contains the amount of correctly classified instances and the amount of misclassified instances.

Real world data sets often have more than 50 features, so this is an indication that even though the OpenML collection of data sets is large, it may be still be skewed towards 'cleaned-up' data sets collected for machine learning and data mining research. By inspecting these meta-models observations like these may surface, and in this case the meta-model will still recommend to apply feature selection for these broader data sets.

6 Conclusion

In this paper we present the results of a large scale experiment on the benefits of using feature selection for classification. We ran 12 algorithms across almost 400 data sets, and created a meta-model to understand when feature selection improves classification accuracy for a given model. Surprisingly, for 41 per cent of algorithm data set combinations feature selection improved the results, but only in 10 per cent of cases this improvement was statistically significant. A possible explanation for this low percentage could be that the data sets from OpenML consist mostly of features that have already been carefully selected by domain experts. The experimental setting would possibly yield other results on raw data from production environments, which would be an interesting direction for future work. Major deciding factors are the number of attributes in the data set, the relative difficulty of the task as measured by landmarkers and the algorithm type. Across algorithms, nearest neighbor benefits most often, but also algorithms that have feature selection built in (such as decision trees) may still benefit.

Future work will focus on extending the set of Feature Selection landmarkers, aiming to perform even better on the meta-leaning task. Having a publicly available meta-data set enables the community to actively participate in this process.

References

1. Blum, A.L., Langley, P.: Selection of relevant features and examples in machine learning. Artificial Intelligence 97(12), 245 – 271 (1997)
2. Brazdil, P., Gama, J., Henery, B.: Characterizing the Applicability of Classification Algorithms Using Meta-Level Learning. In: Machine Learning: ECML-94, Lecture Notes in Computer Science, vol. 784, pp. 83–102. Springer (1994)
3. Carpenter, J.: May the best analyst win. Science 331(6018), 698–699 (2011)
4. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Computers & Electrical Engineering 40(1), 16–28 (2014)
5. Dash, M., Liu, H.: Feature selection for classification. Intelligent data analysis 1(3), 131–156 (1997)
6. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. 3, 1157–1182 (Mar 2003)
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. ACM SIGKDD explorations newsletter 11(1), 10–18 (2009)
8. Hall, M.A.: Correlation-based Feature Subset Selection for Machine Learning. Ph.D. thesis, University of Waikato, Hamilton, New Zealand (1998)
9. John, G.H., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh conference on Uncertainty in artificial intelligence. pp. 338–345. Morgan Kaufmann Publishers Inc. (1995)
10. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial intelligence 97(1), 273–324 (1997)
11. Peng, Y., Flach, P.A., Soares, C., Brazdil, P.: Improved dataset characterisation for meta-learning. In: Discovery Science. pp. 141–152. Springer (2002)
12. Pfahringer, B., Bensusan, H., Giraud-Carrier, C.: Tell me who can learn you and i can tell you who you are: Landmarking various learning algorithms. In: Proceedings of the 17th international conference on machine learning. pp. 743–750 (2000)
13. Pinto, F., Soares, C., Mendes-Moreira, J.: Towards automatic generation of metafeatures. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 215–226. Springer (2016)
14. van der Putten, P., van Someren, M.: A bias-variance analysis of a real world learning problem: The coil challenge 2000. Machine Learning 57(1), 177–195 (2004)
15. Quinlan, J.R.: Induction of decision trees. Machine learning 1(1), 81–106 (1986)
16. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. JMLR 11, 2487–2531 (2010)
17. Rice, J.R.: The Algorithm Selection Problem. Advances in Computers 15, 65118 (1976)
18. van Rijn, J.N., Abdulrahman, S.M., Brazdil, P., Vanschoren, J.: Fast algorithm selection using learning curves. In: International Symposium on Intelligent Data Analysis. pp. 298–309. Springer (2015)
19. Tsamardinos, I., Aliferis, C.: Towards principled feature selection: Relevancy, filters and wrappers. In: Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (2003)
20. Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: networked science in machine learning. ACM SIGKDD Explorations Newsletter 15(2), 49–60 (2014)
21. Verikas, A., Bacauskiene, M.: Feature selection with neural networks. Pattern Recognition Letters 23(11), 1323–1335 (2002)
22. Vilalta, R., Giraud-Carrier, C.G., Brazdil, P., Soares, C.: Using meta-learning to support data mining. IJCSA 1(1), 31–45 (2004)