Pattern Analysis and Applications manuscript No.
(will be inserted by the editor)

Automatic Classifier Selection for Non-Experts

Matthias Reif · Faisal Shafait · Markus Goldstein · Thomas M. Breuel · Andreas Dengel

Received: / Accepted:

Abstract Choosing a suitable classifier for a given dataset is an important part of developing a pattern recognition system. Since a large variety of classification algorithms are proposed in literature, non-experts do not know which method should be used in order to obtain good classification results on their data. Meta-learning tries to address this problem by recommending promising classifiers based on meta-features computed from a given dataset. In this paper, we empirically evaluate five different categories of state-of-the-art meta-features for their suitability in predicting classification accuracies of several widely used classifiers (including Support Vector Machines, Neural Networks, Random Forests, Decision Trees, and Logistic Regression). Based on the evaluation results, we have developed the first open source meta-learning system that is capable of accurately predicting accuracies of target classifiers. The user provides a dataset as input and gets an automatically created high-performance ready-to-use pattern recognition system in a few simple steps. A user study of the system with non-experts showed that the users were able to develop more accurate pattern recognition systems in significantly less development time when using our system as compared to using a state-of-the-art data mining software.

Keywords meta-learning · meta-features · landmarking · regression · classifier selection · classifier recommendation

1 Introduction

The large variety of classification algorithms challenges developers of a pattern recognition system to select an appropriate algorithm for their problem. In general, it is not clear which specific classifier should be used for a particular

Address(es) of author(s) should be given

dataset in order to achieve good results. Some classifiers such as Support Vector Machines (SVM) deliver high accuracy on a variety of datasets. However, the no-free-lunch theorem [42] tells us that there is no learning scheme that can be uniformly better than all other learning schemes for all problem instances. Hence, no universal recommendation can be made for arbitrary data. The actual performance of a classifier compared to alternatives always depends on the characteristics of the data and how well they satisfy the assumptions made by the classifier.

The idea of meta-learning is to learn about the learning algorithms themselves, i.e. to predict how well a learning algorithm will perform on a given dataset. This prediction is based on extracting meta-features – these are features that describe the dataset itself. These features are used to train a meta-learning model on training data (in this case one dataset corresponds to one training sample). Afterwards, this model is applied on the meta-features of a new dataset. The result is the prediction of the suitability or performance of one or more target classifiers. Especially for non-experts in pattern recognition, meta-learning might significantly reduce the development time of a pattern recognition system by decreasing the required level of expertise for choosing a suitable classifier for a given problem.

Different approaches have been presented in the field of meta-learning in the last two decades. A major focus of research has been on identifying what kind of meta-features are suitable for characterizing a dataset. An overview of different meta-features proposed in the literature is given in Section 1.1. Based on these meta-features, several techniques have been proposed for meta-learning using classification, regression, ranking, and case-based reasoning. We will review them in Section 1.2. In the following Section 1.3, we will illustrate how this work advances the state-of-the-art in meta-learning.

1.1 Meta-Feature Extraction and Selection

Many different meta-features have been proposed in the literature. They are derived from different concepts and therefore can be categorized into five groups: simple, statistical, information-theoretic, model-based, and landmarking meta-features.

Simple meta-features are directly derived from the data, e.g. the number of samples, the number of attributes or the number of classes. Statistical features describe statistical properties of the data, e.g. the "peakedness" or the asymmetry of a probability distribution [13,39]. Information-theoretic features are typically based on entropy measures [38]. The Data Characterization Tool (DCT) of the MetaL project is widely used for calculating these groups of meta-features [25,29,9,40,18]. More recently proposed types of meta-features are landmarking features and model-based features.

The landmarking approach utilizes simple and fast computable classification algorithms [27,30,6]. These classifiers are applied on the dataset and the resulting performance values are used as meta-features of the dataset.

Fürnkranz and Petrak [16] evaluated variants of landmarking for pairwise predicting the more suitable classifier. Five different landmarking representations, such as pairwise landmarker comparisons or ranks of the landmarkers were investigated. Besides, the technique of sub-sampling landmarkers was also evaluated. This approach uses the performance of the target classifier on a subsampled dataset as meta-feature. Using landmarking features for predicting the best out of a pair of classifiers has been evaluated by Pfahringer et al. [30]. The authors used artificial datasets for creating a Decision Tree and real world datasets for testing it. The results with landmarking were compared with information-theoretic features and a combination of both. Bensuan et al. [4] used the same type of meta-features and also artificial datasets for training. However, instead of predicting the best out of a pair of classifiers, the best out of all was predicted directly. Additionally, five different meta-learners were investigated. The results show that landmarking features worked best in both cases.

The model-based approach creates a model from the data and uses its properties as feature values. The used model in this context is typically a decision tree [29, 5]. This group of meta-features has been evaluated by Peng et al. [29] according to a ranking approach. A k-Nearest Neighbor algorithm creates the ranking for different meta-feature groups. Comparisons between model-based features, landmarking, and a set of features from different groups are provided. An additional manual feature selection process does neither cover the model-based features nor the landmarking features.

Besides manual meta-feature selection, work on automatic feature selection for meta-learning has only been reported in two papers to our knowledge. Todorovski et al. [40] showed that reducing the set of meta-features can increase the performance of a meta-learning system. A beam-search was used for selecting the best features, whereas the set of considered features does not contain any landmarking nor model-based measures. Since the feature set is optimized according to the final ranking of classifiers, the features were not selected for each target classifier individually. Kalousis et al. [21] also evaluated feature selection for meta-learning. For each pair of classifiers, a separate model was trained with independently selected features. Many meta-features have been used, but neither landmarking nor model-based features were taken into account.

1.2 Meta-Learning Approaches

Rice [37] first presented an abstract model for meta-learning. For a given problem, a mapping function based on meta-features selects the algorithm that maximizes the performance. Rendell and Cho [36] developed rules based on simple meta-features only determining if a certain algorithm should be used for a problem instance or not. This approach was later extended by using more features and a Decision Tree learner within the well known StatLog

project [23]. As a result, the learned model predicts the most suitable classifier for a new problem.

Instead of such a classification approach, Gama and Brazdil [17] used regression in order to predict the performance values of different algorithms. They created linear regression models based on 15 meta-features. Instead of directly predicting the accuracy values or error rates, the normalized error rates using the error margins were predicted. Three different methods for normalizing the error rates of the datasets were investigated. The regression approach was compared with a rule learner, a combination of a Decision Tree and Linear Regression as well as a 3-Nearest Neighbor approach. The results show that it is hard to determine which combination of normalization method and learning algorithm is the best one. Unfortunately, the evaluation was only based on about 20 datasets.

Köpf et al. [24] also presented results of using regression for meta-learning. The M6 method [34] was used as a meta-regression learner. In the evaluation, the error rates of three classification algorithms were predicted for artificial datasets. Additionally, the regression method was compared to the classification approach by selecting the classifier with the lowest predicted error as the best classifier.

A more recent paper about using regression for meta-learning is from Bensusan and Kalousis [7]. The authors used Cubist [33] and a kernel method as meta-learners. The kernel method fits a linear regression model around the performance values of the nearest known datasets. The evaluation was performed on 65 datasets from the UCI repository [3]. The performance of the predictions was measured using the Mean Absolute Deviation. Different sets of meta-features and a feature selection have been investigated as well. Unfortunately, landmarking [30] was not included in the feature selection process although it was already known as a good method at that time. In addition, the regression model was used to produce a ranking of the classifiers. The results were compared with Zooming [9].

Case-based reasoning was used for meta-learning by Linder et al. [25]. Using a case base of known and solved problem instances, the most similar problem served as a basis for the recommendation of one or more algorithms. The similarity of problems was determined by meta-features but the recommendation also included application restrictions concerning the class of the algorithm, interpretability of the produced model, as well as training and testing time.

Besides classification and regression, a commonly used technique for recommending algorithms is ranking. The result of the recommendation procedure is a list of algorithms, ordered by their predicted suitability. The ranking approach has the advantage that the user may evaluate the most suitable algorithms for the given data. Brazdil et al. [8] used a k-Nearest Neighbor approach for creating the ranking. For each algorithm that was selected by the Nearest Neighbor method, a score is computed that takes accuracy as well as total execution time of the algorithm into account. The ranking is directly deduced from these values.

Ali et al. [2] used a rule learner based on meta-features to predict the most suitable classifier. For every target classifier, a rule was learned, determining if the classifier should be used. For the training data, the best classifier was determined by a combined measure of accuracy and time. This work is also one of the few papers until today taking an SVM classifier into account.

Besides the already mentioned StatLog project, a second large-scale project in meta-learning was MetaL [26]. In this project, the Data Mining Advisor [18] was developed as a practical implementation of a meta-learning approach. It used the Data Characterization Tool (DCT) and a k-Nearest Neighbor algorithm to deliver a ranking of ten target classifiers. Unfortunately, neither a classification model nor a usable pattern recognition system was provided at the end. The user could not directly use the recommended classifiers. Additionally, the practical usefulness of the proposed system was not evaluated. The e-LICO project investigated a different approach by using ontologies and templates for discovering complex data mining workflows [22, 19].

1.3 Contributions of this Paper

Summarizing the issues of previous work in meta-learning, the following common limitations can be found:

- Not all five groups of meta-features have been evaluated all together.
- No automatic or only a limited feature selection was performed.
- The recently widely used SVM classifier was hardly investigated.
- Parameters of the target classifiers were not optimized although this might change classification results significantly.
- The presented systems have not been released as Open Source.

Table 1 gives an overview of previous papers and their individual strengths and weaknesses according to these aspects.

In this work, we overcome these limitations by performing exhaustive experimental evaluations. Meta-learning is used to predict the accuracy of each single target classifier independently. The major contributions of the presented work are:

Higher variety of target classifiers: In contrast to previous work, we selected nine representative target classifiers with respect to different concepts in the machine learning domain. We also include recently favored algorithms such as Support Vector Machines (SVM) [12] and Random Forests [10] but also traditional methods that have different theoretical foundations.

Parameter optimization of target classifiers: The most important parameters of each target algorithm are optimized in order to get more realistic performance measures and thus a more convincing evaluation.

Evaluation of all groups of meta-features: The meta-features used for the evaluation comprise all five groups. Each set of meta-features is evaluated for each target classifier independently. To the best of our knowledge, model-based meta-features are used for meta-regression for the first time.

	Investigated Meta-Features					Feature Selection **a**utomatic/**m**anual	Classifier Parameter Optimization	SVM investigated	Released as Open Source
	Simple	Statistical	Inf. Theoretic	Model-Based	Landmarking				
Gama and Brazdil [17]	•	•	•						
Köpf et al. [24]	•	•	•						
Bensusan and Kalousis [7]	•	•	•		•	**a**[1]			
Linder et al. [25]	•	•	•						
Brazdil et al. [8]	•	•	•			**m**			
Ali et al. [2]	•	•	•					•	
Fürnkranz and Petrak [16]					•				
Pfahringer et al. [30]			•		•				
Bensuan et al. [4]			•		•				
Peng et al. [29]	•	•	•	•	•	**m**[2]			
Todorovski et al. [40]	•	•	•			**a**			
Kalousis et al. [21]	•	•	•			**a**			
Presented Approach	•	•	•	•	•	**a**	•	•	•

Table 1 Previous work in meta-learning has limitations regarding the set of used meta-features, feature selection, and the target classifiers.

[1] The feature selection was not applied using the landmarking features.

[2] The feature selection was neither applied using the model-based features nor the landmarking features.

Evaluation according to two performance measures: For a more exhaustive evaluation, the root mean squared error (RMSE) as well as the Pearson product-moment correlation coefficient (PMCC) are used as performance measures. RMSE has the advantage of providing a confidence interval along the prediction, whereas PMCC can measure relative deviations well.

Automatic feature selection: Automatic feature selection is applied twice according to both performance measures and independently for each target classifier. The results are compared to each of the different meta-feature groups as well as to the complete set of meta-features.

Design of a practical system: The complete approach was designed and implemented as a software wizard that recommends classifiers by predicting their accuracies. It uses only freely available algorithms and is fully integrated into RapidMiner [28], the most frequently used data mining software. Moreover, the wizard is able to build a complete pattern recognition system automatically. Therefore, the recommended classifiers can be used right away within RapidMiner.

Practical evaluation of meta-learning: We performed an exhaustive user study to show the usefulness of the presented wizard and the practical applicability of the meta-learning approach. Objective measures such as

achieved accuracy as well as subjective experiences by the users are evaluated.

The rest of this paper is structured as follows. In the next Section, the approach of using regression for meta-learning is presented in more detail. This includes theoretical aspects as well as the proposed design of a practical system. Section 3 contains the evaluation including theoretical comparison of different groups and combinations of meta-features in Section 3.1 as well as the results of a user study of the practical system in Section 3.2. Section 4 concludes the paper.

2 Method

Although especially ranking was often used for classifier recommendation, we think that using regression for meta-learning is more suitable for a practical system used by inexperienced users. Therefore, we first present the theoretical foundations followed by the presentation of a practical system based on these foundations. Furthermore, a ranking can also be obtained directly from our regression approach.

2.1 Meta-Learning using Regression

The goal of using regression in meta-learning [17] is to predict the actual performance outcome for each considered classifier independently instead of predicting the best out of a pair or out of all classifiers (classification). Therefore, a separate regression model is trained for each algorithm. The knowledge of the meta-learner is derived from the training data, which comprises of meta-features of multiple datasets and a target variable. The actual performance values of the classification algorithm serve as this quantitative target variable in this context. Each dataset results in one instance in the training data described by its meta-features and the computed performance of the target classifier. Using these measures, a regression model is learned that describes the relation between the meta-features and the expected performance of datasets.

If the performance of a classifier is to be predicted for a new dataset, the meta-features of this dataset are computed first. Then, the previously learned regression model is applied on these feature values. The result is the predicted performance of the classifier for this new dataset. This procedure is illustrated in Figure 1.

Although some meta-features are originally developed for either nominal or numeric values, both feature types can be converted to each other, e.g. by discretization. Therefore, meta-features can be extracted from datasets with different types of features and the approach is applicable to different types of datasets including datasets with mixed types.

For a recommendation or decision about which algorithm should be used, the results of the different regression models have to be compared. Since the

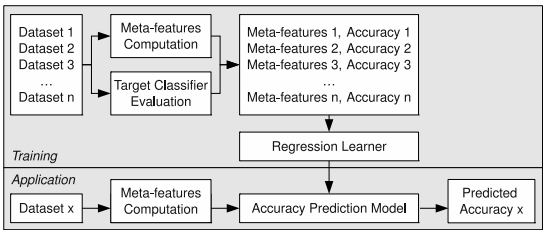


Fig. 1 Training using regression (top) and application of the performance prediction model (bottom) for one target classifier. Here, the accuracy of a classifier is used as the performance measure.

predictions are quantitative values, multiple algorithms may be recommended if two or more predicted accuracies are close. Predicting the accuracy value instead of an arbitrary score has the advantage that the user already gets an estimation of the expected accuracy for his task. Furthermore, the root mean squared error (RMSE) is computed during the training of the meta regression model. This measure can also be used additionally by the user to estimate the confidence of the meta-learner for a given classifier.

A possible drawback of the regression approach is that for each target algorithm, a separate regression model has to be trained. This can be time consuming, especially when an exhaustive parameter optimization for the meta-model is performed. On the other hand, independent models also simplify adding or removing target classifiers. Additionally, the models for different target classifiers can be trained and evaluated independently. In particular, a different regression algorithm, different parameter values or different meta-features can be used.

In addition to the mentioned groups of meta-features, we also apply an automatic feature selection method similar to the one presented by John et al. [20]. Two wrapper approaches are applied to the set of all 43 meta-features whereas the performance of a candidate feature set is determined by a leave-one-out cross-validation of the resulting regression model. First, a forward selection starts with no features and consecutively adds features until no improvement is achieved. In every iteration, it saves the k best feature sets. To avoid being trapped in local maxima, the search stops only after n consecutive iterations without improvement. To remove possibly useless features, a backward elimination is applied afterwards. It works analogically with forward selection except that it consecutively removes features. Unfortunately, this heuristic search does not guarantee that the selected features are the optimal set. For our evaluation, we used values of $k = 5$ and $n = 3$ for the forward selection as well as for the backward elimination step.

The evaluation of a set of meta-features within the feature selection process does not contain any parameter optimization of the meta-learner. Only after the most promising features have been identified, an evaluation with parameter optimization is performed.

2.2 Classifier Recommendation using a Software Wizard

Additionally to the theoretical approach of meta-learning using regression, we present a practical implementation of it. We designed the system as a software wizard that guides the user in just three steps from a dataset to a reasonable well performing classification system. The three steps are as follows:

1. **The user selects a dataset**, meta-features are calculated and supplied to the pre-computed regression models. Both, predicted accuracy and RMSE are presented to the user for each of the target classifiers in a tabular form.
2. **The user selects multiple candidate classifiers,** which are most promising based on the predictions. Each selected classifier is trained on the dataset. Required pre-processing and parameter optimization is performed automatically. The computed classification performance of the selected classifiers are presented to the user.
3. **The user selects a classifier** he wants to get a pattern recognition system for. This is typically the classifier with the highest computed accuracy. A complete pattern recognition system including appropriate pre-processing steps is created automatically. If the user has selected a classification algorithm that already has been evaluated in the previous step, the optimized parameter values will be set automatically. Otherwise, the default parameter values are used.

We integrated the recommendation tool into RapidMiner [28], that is the most frequently used data mining tool according to the KDNuggets poll from 2010 [31]. The wizard is able to handle datasets with numeric, nominal, binary, and mixed features. The pattern recognition system created by the wizard is directly loaded into RapidMiner and can be used immediately.

3 Evaluation

According to the previous section, the evaluation is split up into two parts. The theoretical evaluation of the meta-learning method is presented first. Afterwards, the evaluation of the practical system is presented.

3.1 Comparison of Meta-features

For the evaluation of the regression approach with different meta-features, we used 54 datasets from the UCI machine learning repository [3] and from StatLib [41], although artificially created datasets might be used as well [14]. We randomly selected datasets with different number of samples as well as with different number of nominal and numeric attributes. Additionally, we only used datasets that had an acceptable training time for all target classifiers. For every dataset, the following 43 meta-features are calculated using R and the landmarking extension for RapidMiner [1]:

Simple meta-features: number of samples, number of classes, number of attributes, number of nominal attributes, number of numerical attributes, ratio of nominal attributes, ratio of numerical attributes, dimensionality (number of attributes divided by number of samples)

Statistical meta-features: kurtosis, skewness, canonical discriminant correlation (cancor1), first normalized eigenvalues of canonical discriminant matrix (fract1), absolute correlation

Information-theoretic meta-features: normalized class entropy, normalized attribute entropy, joint entropy, mutual information, noise-signal-ratio, equivalent number of attributes

Model-based meta-features: For these features, a Decision Tree is trained without pruning. Different properties of this tree are used as feature values: number of leaves, number of nodes, nodes per attribute, nodes per sample, leaf corroboration. Additionally, the minimum, maximum, mean value and the standard deviation of the following measures are used: length of a branch (min-, max-, mean-, devBranch), number of nodes in a level (min-, max-, mean-, devLevel), number of occurrences of attributes in a split (min-, max-, mean-, devAtt)

Landmarking meta-features: The accuracy values of the following simple learners are used: Naive Bayes, Linear Discriminant Analysis, One-Nearest Neighbor, Decision Node, Random Node, Worst Node, Average Node.

We use the classification accuracy as the target value for regression. The best classification accuracy for each target classifiers was determined by parameter optimization using a grid search and a ten-fold cross-validation. We selected nine target classifiers based on two principles:

Recently favored used: In previous work, more recent algorithms were not investigated. We include SVM and Random Forest in the set of target classifiers since both are successfully used classifiers on diverse machine learning tasks. Especially SVMs are nowadays usually the first choice of pattern recognition engineers.

Different learning approaches: As already mentioned, the variety of categories of the target classifiers is rather limited in previous work. Usually, the set of target classifiers consists of many tree and rule based methods as well as variations of the C5.0 algorithm [35]. We selected classification algorithms that use different learning foundations such as tree or rule-based learners but also statistical and instance-based learners as well as Neural Networks.

The selected classification algorithms as well as their optimized parameters are listed in Table 2. The features of all datasets have been normalized and nominal features have been converted to numeric features for the SVM, the Multilayer Perceptron (MLP), and the Fisher's Linear Discriminant (FLD) since these algorithms do not support nominal values. In order to avoid a singular within-class covariance matrix S_W within the FLD, this matrix have been regularized using the parameter λ and the identity matrix I [15,32]:

$$S_W(\lambda) = (1 - \lambda)S_W + \lambda I. \tag{1}$$

Classifier	Parameter	Range	Steps	Scale
Decision Tree	confidence	[1.0e-7, 0.5]	100	linear
SVM	γ	[0.001, 10.0]	100	logarithmic
	C	[0.0, 50]	10	logarithmic
Nearest Neighbor	k	[1,500]	100	logarithmic
MLP	learning rate	[1.0e-5, 1.0]	100	logarithmic
Random Forest	k	[1, 21]	10	linear
	depth	{1,2,3,4,5,7,10,20}	-	-
OneR	-	-	-	-
Naive Bayes	-	-	-	-
Logistic Regression	-	-	-	-
FLD	λ	[1.0e-5, 0.1]	8	logarithmic

Table 2 The most important parameters of the nine investigated target classifiers have been optimized in order to get a well-founded ground truth estimation of their performance values.

For each of the nine target classifiers, separate regression models are trained on the normalized meta-dataset. Therefore, the regression variant of a Support Vector Machine, the ε-SVR, is used as learning algorithm based on the LibSVM [11] implementation. We used a radial basis function as kernel whereas the parameters γ and C were optimized independently for every target classifier.

The performance of a regression model is evaluated by a leave-one-out cross-validation. For measuring the performance of the meta-learner we used the root mean squared error (RMSE) and the Pearson product-moment correlation coefficient (PMCC). Feature selection as well as parameter optimization were done independently for both measures.

3.1.1 Root Mean Squared Error

We selected RMSE as one measure because it is an often used measure of precision and can also serve as a confidence indicator of the predictions. Table 3 shows the RMSE values of the trained regression models for the different sets of meta-features. The first column "average accuracy" provides a baseline for our experiments. It contains the RMSE for the case that the average accuracy of the target classifier is used as prediction. This baseline indicates the overall usefulness of meta-learning in general. The last column is the result of the feature selection approach, whereas the subset of meta-features differs for each target classifier. The most frequently selected meta-features are the Naive Bayes landmarker (9 times), the One-Nearest Neighbor landmarker (8 times), and the Decision Node landmarker (8 times). All selected meta-features for each single target classifier can be found in Table 5.

As a first result it can be seen that meta-learning performs clearly better than the baseline. Especially models using feature selection or landmarking features are able to predict the accuracy precisely. It is also noticeable that landmarking alone works better than feature selection for Naive Bayes and

Classifier	average accuracy	simple	statistical	information theoretic	model based	landmarking	all	feature selection
Decision Tree	0.183	0.127	0.126	0.100	0.117	0.076	0.079	**0.069**
SVM	0.193	0.125	0.112	0.113	0.117	0.075	0.083	**0.070**
Nearest Neighbor	0.179	0.130	0.108	0.102	0.110	**0.056**	0.071	0.063
MLP	0.187	0.124	0.115	0.115	0.119	0.081	0.086	**0.076**
Random Forest	0.164	0.111	0.097	0.088	0.100	0.059	0.067	**0.056**
OneR	0.221	0.155	0.145	0.115	0.162	0.071	0.079	**0.065**
Naive Bayes	0.193	0.145	0.125	0.131	0.129	**0.045**	0.061	0.050
Logistic Regression	0.195	0.138	0.126	0.124	0.120	0.062	0.075	**0.059**
FLD	0.190	0.138	0.115	0.116	0.115	0.059	0.074	**0.058**

Table 3 The RMSE values of predicted and computed accuracies of classifiers for different meta-feature groups show that landmarking and feature selection contain the most useful meta-features.

Nearest Neighbor. The reason why the feature selection yields to worse results in these cases is that it does not guarantee to deliver the optimal feature set. Additionally, during the feature selection, no parameter optimization is performed. For example, using only the Naive Bayes landmarker for predicting the Naive Bayes target classifier is of course the best choice. But the feature selection picked more features since this set delivers a higher performance on default parameters of the ϵ-SVR than just the Naive Bayes landmarker.

However, the difference of the results for landmarking and feature selection does not seem to be significant and therefore we applied a two-sided t-test. The differences are not statistically significant with a confidence level of 95%. This means that the computationally intensive procedure of feature selection may be omitted and only landmarking features could be used instead. As already mentioned, the most frequently selected measures by the automatic feature selection are landmarking features.

We also evaluated different set-ups of the landmarking meta-features. We applied them with and without cross-validation and reduced the set of landmarking algorithms according to Pfahringer et al. [30]. The results show that there is no uniformly better set-up and the achieved error rates are quite close. Therefore, no clear recommendation can be made. Nevertheless, we prefer the Pfahringer set-up since the reduced set of classifiers requires less computation time.

3.1.2 Correlation

The RMSE only considers the magnitude of the deviation between predicted and actual performance, but not the sign of it. Nevertheless this aspect can be important for algorithm recommendation. To get a more complete and reliable evaluation of the presented method, the performance was additionally evaluated with a second performance measure. The Pearson product-moment correlation coefficient (PMCC) of the predicted accuracy and the actual, computed accuracy was calculated. In this context, the correlation between two variables X and Y is defined as

$$\rho_{X,Y} = \frac{E\left[(X - \mu_X)(Y - \mu_Y)\right]}{\sigma_X \sigma_Y}. \tag{2}$$

Classifier	simple	statistical	information theoretic	model based	landmarking	all	feature selection
Decision Tree	0.427	0.375	0.729	0.220	0.791	0.819	**0.838**
SVM	0.503	0.558	0.684	0.253	0.865	0.826	**0.883**
Nearest Neighbor	0.478	0.635	0.645	0.340	0.909	0.815	**0.916**
MLP	0.576	0.578	0.592	0.256	0.838	0.810	**0.882**
Random Forest	0.594	0.629	0.733	0.263	0.899	0.825	**0.903**
OneR	0.520	0.544	0.780	0.186	0.900	0.869	**0.919**
Naive Bayes	0.451	0.545	0.585	0.415	0.965	0.914	**0.998**
Logistic Regression	0.560	0.540	0.660	0.391	0.899	0.849	**0.925**
FLD	0.497	0.507	0.696	0.393	0.902	0.866	**0.925**

Table 4 The Pearson product-moment correlation coefficients of different groups of meta-features for the set of target classifiers also show that feature selection performs best.

This measure returns values in the interval $[-1; 1]$. A value of 1 (-1) indicates a perfect positive (negative) relationship. If the correlation is 0, the two input variables are independent.

Table 4 shows the correlation values for the same meta-feature groups as previously evaluated with the RMSE. The feature selection yields the highest correlation values for all nine target classifiers. The most frequently selected features are the One-Nearest Neighbor landmarker (8 times), the Naive Bayes landmarker (6 times), and the number of classes (6 times). All selected features for each single classifier are shown in Table 5. It can be seen that, compared to RMSE, less meta-features have been selected. The reason might be that predicting the accuracy relations as measured by correlation is easier and required therefore less features than predicting precise accuracy values as measured by the RMSE.

The landmarking group is again almost as precise as using the computationally intensive feature selection algorithm. The correlation values were again calculated for different landmarking set-ups. The results are similar to the results using the RMSE: there is no set-up that is uniformly better and the Pfahringer set-up still seems to be a good choice due to its low complexity.

However, it is still not clear, which landmarking algorithm correlates best with a certain target classifier. For this reason, the correlation values between each single accuracy of the landmarkers and each target classifier were computed. Figure 2 shows the results for the four most sophisticated classification algorithms. It can be seen that the One-Nearest Neighbor and the Naive Bayes landmarkers deliver the highest correlation values for these target classifiers. This is not surprising since both classifiers are the only landmarking algorithms that take the complete dataset into account and are non-linear. However, as visible from the table of selected meta-features for RMSE, the remaining landmarkers are suitable for predicting accuracies precisely using an additional meta-regression model.

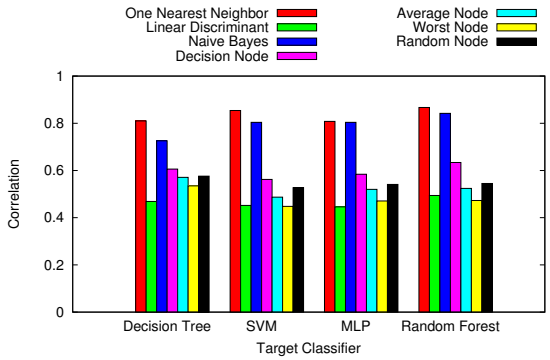


Fig. 2 The correlation values for each single landmarker and the four most sophisticated target classifiers reveal that in general One-Nearest Neighbor and Naive Bayes should be the first choice for landmarkers.

3.2 Practical Evaluation of Classifier Accuracy Prediction

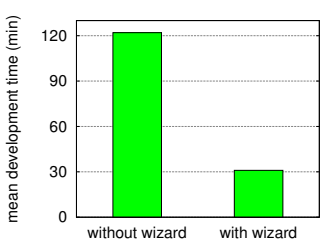
A system which accurately predicts the accuracy of multiple classifiers and is able to construct a classification system automatically can improve and speed-up the development process of pattern recognition systems. Especially non-experts may benefit from an automated approach. The time of getting familiar with pattern recognition theory in general but also with the used software should be decreased significantly. Furthermore, the wizard can also provide additional help to more experienced users. In order to verify the usefulness and the applicability of the proposed wizard, a user study was performed. Since the theoretical evaluation showed that the landmarking features perform well, we also focused on this set of features.

3.2.1 Experiment Set-up

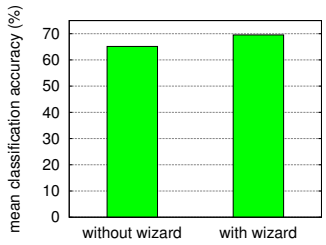
We asked computer science students at the University of Kaiserslautern to build a pattern recognition system for the *profb* dataset from StatLib [41]. The system should deliver a classification model with the highest classification accuracy as possible. To get a comparison between the presented wizard and the traditional approach without meta-learning, the students were asked to solve the problem two times:

Task A: A plain RapidMiner installation has to be used.
Task B: The wizard should be used additionally.

We decided to let the students do task A before task B because the order of performing the tasks should not have a large influence on the results. If task B (using the wizard) would be performed before task A (without wizard), the user would already know a good classifier and its suitable parameters for the given task. But doing task B after task A does not influence the accuracy achieved by the wizard since the classifiers are recommended automatically.



(a) The development time is much lower when using the wizard.



(b) If the wizard was used, a higher accuracy was reached.

Fig. 3 Mean development time and classification accuracy for both tasks.

We also decided to give the students a very short introduction to pattern recognition and RapidMiner. We want to avoid that the students spend too much time with basic understanding and hence may deliver no result at all. Furthermore, it was also allowed to use any additional help as needed, such as the RapidMiner manual or any other resource on the Internet. The importance of additional help was investigated in the evaluation.

The complete process was done two times. The first, preliminary round consisted of nine students, which gave important feedback about the usability of the wizard. Based on this feedback, we revised the user interface of the wizard for the second round with 24 students. All results presented in this paper are based on the second round only.

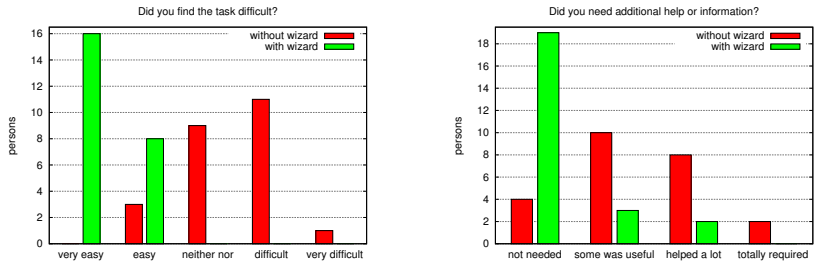
3.2.2 Accuracy and Time

The achieved classification accuracy as well as the time needed were recorded for each single student. The mean time that the students needed for developing the systems for task A and task B is plotted in Figure 3(a). The accuracy values were determined by using a separate test set. The achieved mean classification accuracies are shown in Figure 3(b). The plots show that using the wizard, the development time could be decreased significantly. Moreover, the achieved performance could be increased as well.

3.2.3 Questionnaire

In addition to the actual tasks, we asked each student to answer a questionnaire. It contained multiple-choice questions about previous knowledge, comparison between RapidMiner and the wizard as well as general questions about the wizard.

Previous Knowledge: These questions give us an impression about the experience of the students in pattern recognition. The goal was to determine if the wizard is equally useful for differently experienced users. The students were asked what their main focus of study is, if they have attended a pattern



(a) The wizard is able to decrease the difficulty of developing a pattern recognition system.

(b) The required level of experience is much lower when the wizard is used.

Fig. 4 Results of the questionnaire comparing the presented wizard for classifier recommendation and the standard approach.

recognition course, and if they are already familiar with RapidMiner. Approximately half of the students had attended the pattern recognition course taught at the University of Kaiserslautern but none of them had prior experience with RapidMiner.

Comparison of Task A and Task B: This part of the questionnaire is the most important part since it directly compares the standard method with the presented approach. In addition to the more objective measures of time and accuracy, these questions show the advantages of using meta-learning and, particularly, the presented wizard approach.

An important aspect of the user study was to compare the perceived level of difficulty of both tasks. It is interesting to see that the mean opinion score (MOS) about the difficulty level of the classification task, on a scale from 1 (very easy) to 5 (very difficult), was reduced from 3.4 when using RapidMiner only to 1.3 when using the wizard. The frequencies of the answers are shown in Figure 4(a).

Another question examines the level of expertise or the amount of experience that is required for solving the tasks. It was asked, how much additional help or information the user needed for solving the tasks. The MOS on a scale from 1 (not needed) to 4 (totally required) was reduced from 2.3 to 1.3 by using the wizard. Figure 4(b) shows a summary of user responses to this question.

Rating of the Wizard: The last part of the questionnaire evaluates the wizard and its meta-learning approach in general. All students considered the wizard to be simple and intuitive and most of them have a good overall impression about it. Additionally, most of the users would use the wizard as an additional help instead of using either RapidMiner or the wizard only for future pattern recognition tasks. Also, most of the users would recommend the wizard to somebody else, who is new to pattern recognition.

4 Conclusion

An exhaustive evaluation of meta-features for meta-learning using regression was performed based on 54 datasets. We showed that meta-learning does make sense in general by comparing the results with a baseline strategy. Five different categories of meta-features, namely simple, statistical, information-theoretic, model-based and landmarking were used and comparatively evaluated. Furthermore, a feature selection approach according to RMSE and correlation has been utilized in order to select the most promising features out of all categories. As a result, we found that feature selection yields the best results in total, whereas landmarking achieved nearly the same performance, but with times less computational effort. Especially the landmarking set-up by Pfahringer et al. [30] with only four landmarkers can decrease the computational effort further without significantly reducing the performance.

We also presented a technical implementation of meta-learning to evaluate the applicability and practical usefulness of the theoretical approach. A software wizard for classifier recommendation using regression and landmarking features was developed and integrated into RapidMiner. With this open-source tool it is now for the first time possible for researchers to actually reproduce results of meta-learning processes precisely. From a system engineer point of view, the wizard was evaluated according to its usefulness by a user study with 24 students. The meta-learning tool shows improvements in development time and achieved a classification accuracy compared to a standard pattern recognition approach. Additionally, the level of expertise required for building a pattern recognition system as well as the overall level of difficulty could be reduced. The wizard was released in collaboration with Rapid-I GmbH as the first third-party extension for RapidMiner and can be directly installed via the RapidMiner extension manager.

References

1. Abdelmessih, S.D., Shafait, F., Reif, M., Goldstein, M.: Landmarking for meta-learning using RapidMiner. In: RapidMiner Community Meeting and Conference (2010)
2. Ali, S., Smith, K.A.: On learning algorithm selection for classification. Applied Soft Computing **6**, 119–138 (2006)
3. Asuncion, A., Newman, D.: UCI machine learning repository (2007). `http://www.ics.uci.edu/~mlearn/MLRepository.html` University of California, Irvine, School of Information and Computer Sciences
4. Bensusan, H., Giraud-Carrier, C.: Casa batló is in passeig de gràcia or how landmark performances can describe tasks. In: Proceedings of the ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, pp. 29–46 (2000)
5. Bensusan, H., Giraud-Carrier, C., Kennedy, C.: A higher-order approach to meta-learning. In: Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, pp. 109–117 (2000)
6. Bensusan, H., Giraud-Carrier, C.G.: Discovering task neighbourhoods through landmark learning performances. In: PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 325–330. Springer-Verlag, London, UK (2000)

7. Bensusan, H., Kalousis, A.: Estimating the predictive accuracy of a classifier. In: L. De Raedt, P. Flach (eds.) Machine Learning: ECML 2001, *Lecture Notes in Computer Science*, vol. 2167, pp. 25–36. Springer Berlin / Heidelberg (2001)
8. Brazdil, P., Soares, C., da Costa, J.P.: Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. Machine Learning **50**(3), 251–277 (2003)
9. Brazdil, P.B., Soares, C.: Zoomed ranking: Selection of classification algorithms based on relevant performance information. In: In Proceedings of Principles of Data Mining and Knowledge Discovery, 4th European Conference (PKDD-2000, pp. 126–135. Springer (2000)
10. Breiman, L.: Random forests. Machine Learning **45**, 5–32 (2001)
11. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
12. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3), 273–297 (1995)
13. Engels, R., Theusinger, C.: Using a data metric for preprocessing advice for data mining applications. In: Proceedings of the European Conference on Artificial Intelligence (ECAI-98, pp. 430–434. John Wiley & Sons (1998)
14. Frasch, J.V., Lodwich, A., Shafait, F., Breuel, T.M.: A bayes-true data generator for evaluation of supervised and unsupervised learning methods. Pattern Recogn. Lett. **32**(11), 1523–1531 (2011)
15. Friedman, J.H.: Regularized discriminant analysis. Journal of the American Statistical Association **84**(405), pp. 165–175 (1989)
16. Fürnkranz, J., Petrak, J.: An evaluation of landmarking variants. In: C. Giraud-Carrier, N. Lavrač, S. Moyle, B. Kavšek (eds.) Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning (IDDM-2001), pp. 57–68. Freiburg, Germany (2001)
17. Gama, J., Brazdil, P.: Characterization of classification algorithms. In: C. Pinto-Ferreira, N. Mamede (eds.) Progress in Artificial Intelligence, *Lecture Notes in Computer Science*, vol. 990, pp. 189–200. Springer Berlin / Heidelberg (1995)
18. Giraud-Carrier, C.: The data mining advisor: meta-learning at the service of practitioners. In: Machine Learning and Applications, 2005. Proceedings. Fourth International Conference on, p. 7 pp. (2005)
19. Hilario, M., Nguyen, P., Do, H., Woznica, A., Kalousis, A.: Ontology-based meta-mining of knowledge discovery workflows. In: N. Jankowski, W. Duch, K. Gra¸bczewski (eds.) Meta-Learning in Computational Intelligence, *Studies in Computational Intelligence*, vol. 358, pp. 273–315. Springer Berlin / Heidelberg (2011)
20. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: International Conference on Machine Learning, pp. 121–129. Morgan Kaufmann (1994)
21. Kalousis, A., Hilario, M.: Feature selection for meta-learning. In: D. Cheung, G. Williams, Q. Li (eds.) Advances in Knowledge Discovery and Data Mining, *Lecture Notes in Computer Science*, vol. 2035, pp. 222–233. Springer Berlin / Heidelberg (2001)
22. Kietz, J.U., Serban, F., Bernstein, A., Fischer, S.: Data mining workflow templates for intelligent discovery assistance and auto-experimentation. In: Proceedings of the ECML/PKDD-10 Workshop on Third Generation Data Mining: Towards Service-Oriented Knowledge Discovery, pp. 1–12 (2010)
23. King, R.D., Feng, C., Sutherland, A.: Statlog: Comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence **9**(3), 289–333 (1995)
24. Köpf, C., Taylor, C., Keller, J.: Meta-analysis: From data characterisation for meta-learning to meta-regression. In: Proceedings of the PKDD-00 Workshop on Data Mining, Decision Support,Meta-Learning and ILP (2000)
25. Lindner, G., Studer, R.: Ast: Support for algorithm selection with a cbr approach. In: Recent Advances in Meta-Learning and Future Work, pp. 418–423 (1999)
26. Esprit project METAL (#26.357): A meta-learning assistant for providing user support in data mining and machine learning (1999-2002). Http://www.ofai.at/research/impml/metal/
27. Michie, D., Spiegelhalter, D., Taylor, C.: "Machine Learning, Neural & Statistical Classification". Ellis Horwood, Chichester (1994)

28. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: L. Ungar, M. Craven, D. Gunopulos, T. Eliassi-Rad (eds.) KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 935–940. ACM, New York, NY, USA (2006)
29. Peng, Y., Flach, P., Soares, C., Brazdil, P.: Improved dataset characterisation for meta-learning. In: S. Lange, K. Satoh, C. Smith (eds.) Discovery Science, *Lecture Notes in Computer Science*, vol. 2534, pp. 193–208. Springer Berlin / Heidelberg (2002)
30. Pfahringer, B., Bensusan, H., Giraud-Carrier, C.: Meta-learning by landmarking various learning algorithms. In: In Proceedings of the Seventeenth International Conference on Machine Learning, pp. 743–750. Morgan Kaufmann (2000)
31. Piatetsky-Shapiro, G.: Data mining / analytic tools used poll (2010). Http://www.kdnuggets.com/polls/2010/data-mining-analytics-tools.html
32. Qiao, Z., Zhou, L., Huang, J.Z.: Sparse linear discriminant analysis with applications to high dimensional low sample size data. IAENG International Journal of Applied Mathematics **39**(1), 48–60 (2009)
33. Quinlan, J.R.: Cubist. Http://www.rulequest.com/cubist-info.html
34. Quinlan, J.R.: Learning with continuous classes. In Proceedings AI'92 pp. 343–348 (1992)
35. Quinlan, R.: Data mining tools see5 and c5.0 (2002). Http://www.rulequest.com/see5-info.html
36. Rendell, L., Cho, H.: Empirical learning as a function of concept character. Machine Learning **5**, 267–298 (1990)
37. Rice, J.R.: The algorithm selection problem. Advances in Computers **15**, 65–118 (1976)
38. Segrera, S., Pinho, J., Moreno, M.: Information-theoretic measures for meta-learning. In: E. Corchado, A. Abraham, W. Pedrycz (eds.) Hybrid Artificial Intelligence Systems, *Lecture Notes in Computer Science*, vol. 5271, pp. 458–465. Springer Berlin / Heidelberg (2008)
39. Sohn, S.Y.: Meta analysis of classification algorithms for pattern recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on **21**(11), 1137 –1144 (1999)
40. Todorovski, L., Brazdil, P., Soares, C.: Report on the experiments with feature selection in meta-level learning. In: Proceedings of the PKDD-00 Workshop on Data Mining, Decision Support, Meta-Learning and ILP, pp. 27–39 (2000)
41. Vlachos, P.: StatLib datasets archive (1998). `http://lib.stat.cmu.edu` Department of Statistics, Carnegie Mellon University
42. Wolpert, D.H.: The lack of a priori distinctions between learning algorithms. Neural Comput. **8**(7), 1341–1390 (1996)

A Selected Meta-Features

	RMSE										Correlation									
Meta-Features	Count	Decision Tree	SVM	Nearest Neighbor	MLP	Random Forest	OneR	Naive Bayes	Logistic Regression	FLD	Count	Decision Tree	SVM	Nearest Neighbor	MLP	Random Forest	OneR	Naive Bayes	Logistic Regression	FLD
Simple																				
Number of classes	3	•		•		•					6	•	•	•		•			•	•
Number of attributes	1							•			1		•							
Number of numeric attributes	0										0									
Number of nominal attributes	2	•			•						0									
Number of samples	2		•		•						1								•	
Dimensionality	2				•				•		3			•	•				•	
Rate of numeric attributes	0										0									
Rate of nominal attributes	0										1						•			
Statistical																				
skewness	4	•	•		•			•			2		•						•	
kurtosis	1			•							2	•	•							
absolute correlation	2	•						•			0									
cancor1	7		•	•	•	•		•	•	•	0									
fract1	0										0									
Information-theoretic																				
Normalized class entropy	0										1					•				
Normalized attribute entropy	1	•									3		•			•				•
Joint entropy	6	•	•	•	•	•	•				1	•								
Mutual information	2					•		•			2	•	•							
Equivalent number of attributes	4	•		•	•			•			2			•	•					
Noise-signal ratio	2				•					•	2			•	•					
Model-based																				
Number of nodes	1								•		0									
Number of leaves	1								•		0									
Nodes per attribute	2	•				•					0									
Nodes per sample	5		•		•			•	•	•	0									
Leaf corrobation	2					•		•			1	•								
minLevel	1				•						5	•		•				•	•	•
maxLevel	1					•					0									
meanLevel	2				•			•			1				•					
devLevel	2	•			•						1				•					
minBranch	2					•	•				3			•	•		•			
maxBranch	4		•	•	•				•		0									
meanBranch	3			•	•				•		0									
devBranch	2			•	•						0									
minAtt	2	•					•				0									
maxAtt	4	•		•	•			•			0									
meanAtt	0										0									
devAtt	2			•	•						1									•
Landmarking																				
Linear Discriminant	5		•		•	•			•	•	0									
One-Nearest Neighbor	8	•	•	•	•	•		•	•	•	8	•	•	•	•	•	•		•	•
Decision Node	8	•		•	•	•	•	•	•	•	1						•			
Worst Node	1						•				0									
Average Node	5			•		•	•	•	•		0									
Random Node	7		•	•		•	•	•	•	•	1						•			
Naive Bayes	9	•	•	•	•	•	•	•	•	•	6			•	•	•		•	•	•
Count		14	10	15	21	14	8	15	13	8		7	7	8	8	5	5	2	7	6

Table 5 The selected meta-features of each target classifier. The regression model was optimized according to either RMSE or correlation criterion.