Filter Methods for Feature Selection in Supervised Machine Learning Applications—Review and Benchmark

Konstantin Hopf, Sascha Reifenrath

University of Bamberg

Chair of Information Systems and Energy Efficient Systems

Kapuzinerstr. 16, D-96047 Bamberg

contact: konstantin.hopf@uni-bamberg.de

November 25, 2021

Abstract

The amount of data for machine learning (ML) applications is constantly growing. Not only the number of observations, especially the number of measured variables (features) increases with ongoing digitization. Selecting the most appropriate features for predictive modeling is an important lever for the success of ML applications in business and research. Feature selection methods (FSM) that are independent of a certain ML algorithm—so-called filter methods—have been numerously suggested, but little guidance for researchers and quantitative modelers exists to choose appropriate approaches for typical ML problems. This review synthesizes the substantial literature on feature selection benchmarking and evaluates the performance of 58 methods in the widely used R environment. For concrete guidance, we consider four typical dataset scenarios that are challenging for ML models (noisy, redundant, imbalanced data and cases with more features than observations). Drawing on the experience of earlier benchmarks, which have considered much fewer FSMs, we compare the performance of the methods according to four criteria (predictive performance, number of relevant features selected, stability of the feature sets and runtime). We found methods relying on the random forest approach, the double input symmetrical relevance filter (DISR) and the joint impurity filter (JIM) were well-performing candidate methods for the given dataset scenarios.

Keywords: Businss analytics, big data analytics, feature selection, filter methods, machine learning, benchmark

1 Introduction

Machine Learning (ML) is a core technology of artificial intelligence (Russell and Norvig, 2016), which has rendered current outstanding applications of speech processing, image recognition, self-driving cars, and others possible. Next to these remarkable technological developments, ML and related techniques also alter the data processing in business and research. Davenport and Harris (2007) were right in their prediction that firms today build competitive advantages through analytics (Kraus et al., 2020; Müller et al., 2018; Wu et al., 2019). Firms do so by employing effective data analysis and ML to extract relevant information faster and make more informed decisions. Likewise, researchers benefits from ML-related techniques allowing them to explore new patterns in data and thus to discover new insights or theories (Berente et al., 2019; Shmueli and Koppius, 2011).

1.1 Feature selection as a core activity in pursuing ML data analysis

A core task in setting up ML is to identify the relevant features from the wide variety of input data. Feature selection mitigates the "curse of dimensionality" problem (Jain and Zongker, 1997; Keogh and Mueen, 2011), which is known to lower the quality of ML predictions. It allows learning less complex models, which are less likely to be over-fitted to the data, lowers training times, and requires less storage space of data and models (Guyon et al., 2003; Kudo and Sklansky, 2000). Feature selection, thus, reduces computing costs of data analyses (Li et al., 2017).

The necessity of feature selection has increased in recent years: Whereas in the last decades, a number of 50 to 100 features was called a "large" feature set (Kudo and Sklansky, 2000, p. 25), today we are confronted with hundreds or even thousands (Hua et al., 2009) of features that are measured on every entity. Further digitization will aggravate this, as it allows measuring even more variables in business operations through connected devices and novel sensors (Feng and Shanthikumar, 2018). Although some studies argue that modern neural networks (i.e., deep learning) can be applied directly to raw data without requiring feature selection (Kraus et al., 2020; LeCun et al., 2015), it seems that their predictive performance can still be improved in some areas by selecting suitable features before applying ML (Semwal et al., 2017; Borisov et al., 2019). Moreover, not for all application areas of ML are deep learning approaches the most suitable ones (Fernández-Delgado et al., 2014). Thus, feature selection remains an essential task in setting up ML.

When ML is adopted by wider ranges of applications, related data analysis techniques, like Feature Selection Methods (FSMs), similarly have to cope with the challenges of ML use (L'Heureux et al., 2017; Bosu and MacDonell, 2013; Kaur et al., 2019; Branco et al., 2016). Those challenges are, for example, missing values, low data quality (e.g., redundancies and noise in the data), and small datasets (e.g., few observations but many variables, and the infrequent occurence of interesting events, which leads to imbalanced

datasets).

1.2 The multitude of available feature selection methods complicates their choice

Several software libraries offer an extensive number of FSMs to researchers, ML users, and data scientists with ready-to-use interfaces (Robnik-Sikonja and Savicky, 2020; Bischl et al., 2016; Romanski and Kotthoff, 2018; Kursa, 2020; Li et al., 2017). With regard to the feature selection strategy, literature typically classifies the approaches into three categories (Blum and Langley, 1997; Guyon et al., 2003; Li et al., 2017): Wrapper, embedded, and filter methods. Wrappers utilize ML models that are trained and tested to assess the predictive power of single feature sets and to find feature sets with a local maximum of the model performance. Embedded methods are bound to specific ML algorithms. Filter methods, by contrast, obtain a score for each feature based on the dataset, independent of the applied ML algorithm. They either select a set of the best features or they rank features based on their estimated merit based on this scoring.

Due to the large number of available FSMs, it is difficult to choose the most appropriate method for a given ML problem. There is a series of reviews that describe the various available FSMs available (Li et al., 2017; Bommert et al., 2020; Chandrashekar and Sahin, 2014; Guyon et al., 2003; Liu et al., 2010; Saeys et al., 2007). More or less comprehensive FSM benchmark studies also investigated how well existing FSMs behave—in combination or without ML algorithms. This research found that filter methods show a comparable good performance in comparison to other approaches (Haury et al., 2011; Saeys et al., 2008). Filters also usually need less computational resources than wrapper methods and the selection of features does not depend on the choice of the learning algorithm or the evaluation metric. Given that filter methods evaluate features independently of any particular classifier, the extracted features are "generic, having incorporated few assumptions" (Brown et al., 2012).

When focusing on studies that compare filtering methods for feature selection, we notice that, while there are some comparative studies, they provide only a partial overview. One reason is that existing studies consider different sets of FSMs and that many of them use domain-specific datasets (e.g., bioinformatics, text analysis, energy retailing). Another reason is that these works often only focus on the high dimensionality of data, but not other dataset properties, like noise, skewed class distributions or redundant features. Finally, the studies come to different conclusions. This gives rise to the assumption that the no free lunch theorem (Wolpert, 1996) applies for feature selection as well. This theorem states, in basic words, that different optimization problem strategies perform equally well when averaged over all possible problems (Gómez and Rojas, 2016). Yet, when applying ML and related methods, it is of little relevance that there is no all-surpassing method in theory. In the data analysis practice, usually not all theoretical dataset problems occur at the same time, but analysts are often confronted with few but severe dataset problems

(noisy data, skewed data, small datasets, etc.), which are manifested to varying degrees. Therefore, it is rather important to know those FSMs which have turned out to be the most helpful for specific problem classes.

1.3 Objectives of this work

Our work extends previous efforts to obtain actionable knowledge for researchers and ML through benchmark studies of ML algorithms (Baumann et al., 2019; Fitzpatrick and Mues, 2016; Gartner et al., 2015) and ML platforms (Roy et al., 2019). Thus, we seek to address the lack of an overview to effective FSMs in particular problem classes. In doing so, our paper makes two contributions. First, we provide an overview to the current state of benchmark experiments on FSMs and conduct a meta-review of 33 studies. This overview will help ML users as a reference to more specific analyses in already existing benchmark studies. We also derive a synthesized benchmark method, which we apply in the latter part of our study. Second, we report on an extensive benchmark study involving 58 available in the widely used statistical programming environment GNU R, which represents—to the best of our knowledge—the most comprehensive benchmark of FSMs so far. Our experiment builds on learnings from the earlier benchmark studies. Thereby, we evaluated the performance of FSMs in four dataset scenarios that are typical ML problems: We use two problems from the area of *low data quality* (Lee and Shin, 2020; Bosu and MacDonell, 2013; Alonso, 2015), namely (I) noise in the data (class and attribute noise) and (II) redundancy. Besides, we use two problems from the area of *small datasets* (L'Heureux et al., 2017; Bosu and MacDonell, 2013), namely (III) few observations with many features and (IV) few observations of a minority class (i.e., imbalanced classes). Moreover, we examine whether methods that are described as particularly well suited for specific dataset problems (e.g. noisy and imbalanced data) also perform better than other methods in these scenarios.

With the no free lunch theorem in mind, our benchmark study cannot be expected to lead to a selection of methods that always outperform others. Rather than aiming for a general recommendation, we attempt to provide assistance for concrete dataset scenarios and focus on our benchmark analysis on four scenarios.

1.4 Structure of the paper

This paper proceeds with our survey of earlier FSMs benchmark studies. section 3 continues with presenting our benchmark of currently available FSMs in the GNU R environment. Thereby, we compile a holistic benchmark approach for FSMs as a result of the literature survey and apply it to 58 FSMs in the widely-used GNU R environment. We report the performance of each method for four dataset challenges. We conclude this paper with a summary and discussion our results.

2 Background

Several software libraries offer an extensive number of FSMs to researchers, ML users, and data scientists with ready-to-use interfaces (Robnik-Sikonja and Savicky, 2020; Bischl et al., 2016; Romanski and Kotthoff, 2018; Kursa, 2020; Li et al., 2017). This section gives an overview to FSMs and reviews earlier benchmark studies that compare different approaches.

2.1 Overview to FSMs

With regard to the feature selection strategy, literature typically classifies the approaches into three categories (Blum and Langley, 1997; Guyon et al., 2003; Li et al., 2017): Wrapper, embedded, and filter methods. Wrappers utilize ML models that are trained and tested to assess the predictive power of single feature sets and to find feature sets with a local maximum of the model performance. Embedded methods are bound to specific ML algorithms. Filter methods, by contrast, obtain a score for each feature based on the dataset, independent of the applied ML algorithm. They either select a set of the best features or they rank features based on their estimated merit based on this scoring.

Due to the large number of available FSMs, it is difficult to choose the most appropriate method for a given ML problem. There is a series of reviews that describe the various available FSMs available (Li et al., 2017; Bommert et al., 2020; Chandrashekar and Sahin, 2014; Guyon et al., 2003; Liu et al., 2010; Saeys et al., 2007). More or less comprehensive FSM benchmark studies also investigated how well existing FSMs behave—in combination or without ML algorithms. This research found that filter methods show a comparable good performance in comparison to other approaches (Haury et al., 2011; Saeys et al., 2008). Filters also usually need less computational resources than wrapper methods and the selection of features does not depend on the choice of the learning algorithm or the evaluation metric. Given that filter methods evaluate features independently of any particular classifier, the extracted features are "generic, having incorporated few assumptions" (Brown et al., 2012).

When focusing on studies that compare filtering methods for feature selection, we notice that, while there are some comparative studies, they provide only a partial overview. One reason is that existing studies consider different sets of FSMs and that many of them use domain-specific datasets (e.g., bioinformatics, text analysis, energy retailing). Another reason is that these works often only focus on the high dimensionality of data, but not other dataset properties, like noise, skewed class distributions or redundant features. Finally, the studies come to different conclusions. This gives rise to the assumption that the no free lunch theorem (Wolpert, 1996) applies for feature selection as well. This theorem states, in basic words, that different optimization problem strategies perform equally well when averaged over all possible problems (Gómez and Rojas, 2016). Yet, when applying ML and related methods, it is of little relevance that there is no all-surpassing method in theory. In the data analysis practice, usually not all theoretical dataset

problems occur at the same time, but analysts are often confronted with few but severe dataset problems (noisy data, skewed data, small datasets, etc.), which are manifested to varying degrees. Therefore, it is rather important to know those FSMs which have turned out to be the most helpful for specific problem classes. Yet, with the no free lunch theorem in mind, our benchmark study cannot be expected to lead to a selection of methods that always outperform others. Rather than aiming for a general recommendation, we attempt to provide assistance for concrete dataset scenarios and focus on our benchmark analysis on four scenarios.

2.2 Earlier benchmark studies for FSMs

To obtain an overview to existing benchmark studies on FSMs, we queried premier scientific databases (e.g., ACM digital library, Google Scholar, EBSCO Business Source Ultimate). We used a combination of the search terms *feature selection* or *variable selection* with one of the words *benchmark*, *comparison*, *experiment*. Based on the title, abstract, keywords, and if necessary the full-text, we decided whether the respective paper is relevant to our review. Based on the initial set of studies, we also included studies that the most recent articles referenced (backward search). We found several studies that suggest new FSMs and compare their method(s) to existing ones, merely to demonstrate their superiority. We included such studies only when they considered more than five other methods in their comparison (as, for example, Brown et al. (2012) and Meyer et al. (2008) did). In total, our literature search resulted in a collection of 33 studies.

We reviewed all identified studies and pulled out the number and type of considered FSMs, the used data for the benchmark (i.e., number of datasets, if they are real or artificial data, the number of features and observations), and the application subject (e.g., general ML, biomedical, text analysis). We also analyzed which specific dataset challenges the works are concerned with at their core (i.e., more features than observations, imbalanced class distributions, noisy data, redundant features, certain variable types). Finally, we surveyed which evaluation metrics the papers used in evaluating FSMs.

Table 1 shows the results of our review with the study characteristics. Except from Bommert et al. (2020), who compare 22 methods with 16 datasets and Hopf (2019), who compares 43 methods with one dataset from electricity retailing, studies include only some, at most eleven, feature filters in their benchmark. Many works use only a limited set of performance criteria and mostly focus on prediction accuracy alone.

Table 1: Earlier FSM benchmark studies (in chronological order)

		Dataset			Dataset challenge							Metrics					
Ref.	Subject	Number of datasets[a]	Max num. of Features	Training samples	High-dimensional	Class imbalance	Noisy data	Redundant features	Variable type (cat., num.)	Classifier[b]	FSMs in benchmark[c]	Predictive performance[d]	Stability	Correct chosen features	Number of selected features	Comparison of F/W/E	Runtime performance
Dash and Liu (1997)	General ML	3 (S)	12	122			X	X		NB, DT	8 (F, W)			X		X	
Kudo and Sklansky (2000)	Diverse domains	8 (R, S)	65	1,000						IB	9 (W)	A			X		X
Liu et al. (2002)	Biomedical	2 (R)	15,154	327	X					IB, DT, NB	6 (F)	A			X		
Forman (2003)	Text classification	1 (R)	16	229	X					NB, DT, LR, SVM	11 (F)	AFR			X		
Reunanen (2003)	Diverse datasets	7 (R)	112	1,000				X		IB	2 (W)	A			X		
Robnik-Šikonja (2003)	General ML	12(S)	25	1,000		X				-	13 (F)			X			
Hall and Holmes (2003)	Diverse domains	18 (R)	1,555	29,598	X					DT, NB	6 (F, W)	A			X	X	X
Inza et al. (2004)	Biomedical data	2 (R)	7,129	72						IB, NB, DT	7 (F, W)	A				X	
Peng et al. (2005)	Medical data and handwritten digits	4 (R, S)	9,703	2,000	X				X	NB, SVM, LDA	3 (F)	A			X		
Lai et al. (2006)	Biomedical data	7 (R)	5,963	102	X			X		SVM, Fisher, IB	7 (F, W, E)	A				X	
Sánchez-Maroño et a (2007)	General ML	2 (S)	200	400	X		X	X		-	4 (F)			X			
Kalousis et al. (2007)	Biomedical data	11 (R)	4,031	7,977	X					-	5 (F)		X				
Saeys et al. (2008)	Biomedical data	6 (R)	15,154	322						RF, IB, SVM	4 (F, E)	A	X			X	
Meyer et al. (2008)	Biomedical data	12(R, S)	15	308			X			SVM, IB	6 (F)	A		X			X
Dougherty et al. (2009)	Biomedical data	2 (S,R)	20	295	X		X			LDA, IB	4 (F, W)	A			X	X	
Hua et al. (2009)	Biomedical data	1 (S)	20	180	X		X			IB, SVM	8 (F,W)	A			X	X	
Haury et al. (2011)	Biomedical data	4 (R)	100	286	X					NC, SVM, NB, LDA	8 (F, W, E)	R	X			X	
Alelyani et al. (2011)	General ML	5 (R)	22,283	718						-	5		X			X	
Brown et al. (2012)	Diverse domains	15 (R)	256	6,435	X					IB	9 (F)	A	X				
Bolón-Canedo et al. (2013)	General ML	11(S, R)	4,060	2,915	X		X	X		DT, SVM, NB, IB	12 (F, W, E)	A		X		X	
Chandrashekar and (2014)	Electrical engineering	6 (R)	34	n.a.						SVM, ANN	4 (F, W)	A				X	
Bolón-Canedo et al. (2014)	Biomedical data	9 (R)	24	254	X					DT, SVM, NB	7 (F)	AS					
Aphinyanaphongs et (2014)	Text	20 (R)	7,549	11,162						SVM, LR, NB, AdaBoost	F, W	R					

Khoshgoftaar et al. (2015)	Software defects	3 (R)	209	661	X	X				SVM, ANN	5 (F, W, E)	R				X		
Xue et al. (2015)	Diverse domains	24 (R)	617	6,435					X	IB, SVM, DT, NB	10 (F, W)	A			X	X	X	
Adams et al. (2017)	General ML	10	64	20						DT, IB, RF	5 (F, W)	A				X		
Liu et al. (2018)	General ML	9 (S)	120	1,650		X				SVM	9 (F)	AFR			X			
Darshan and Jaidha (2018)	Malware data	2 (R)	1,877	600	X	X				LR, RF, DT	4 (F)	AS						
Wah et al. (2018)	Biomedical data	5 (S)	57	1,000	X		X			LR	8 (F, W)	ASR		X		X		
Nogueira et al. (2018)	General ML	1 (S)	100	2,000						LR	2 (E)		X					
Hopf (2019)	Energy retailing	1 (R)	308	451						LR	43 (F)	A	X		X		X	
Bommert et al. (2020)	General ML	16 (R, S)	22,283	7,000	X			X	X	IB, LR, SVM	22 (F)	A	X				X	
Kou et al. (2020)	Text	10 (R)	27,973	1,000	X					SVM, IB, NB	10 (F)	AFR	X					
This study		13 (S)	1,000	2,500	X	X	X	X		SVM, RF, NB	58 (F)	R	X	X	X		X	

a) S: Synthetic, R: Real

b) ANN: Artificial Neural Networks, DT: Decision trees (e.g., C4.5), IB: Instance based learners (e.g., k nearest neighbor, nearest centroid, IB1), LDA: Linear Discriminant Analysis, LR: Logistic Regression, NB: Naïve Bayes, RF: Random Forest, SVM: Support Vector Machines

c) F: Filter, W: Wrapper, E: Embedded

d) A: Accuracy or error rate, F: F-Measure, R: ROC or AUC, S: sensitivity and specificity

In detail, we identified three streams of literature. The first one (Dash and Liu, 1997; Kudo and Sklansky, 2000; Reunanen, 2003, and others) compares wrapper and filter methods on the basis of different datasets in terms of classification accuracy. From these studies, no single method was found to be superior to others. A major drawback of these works is that FSM benchmarks only considered the classification performance. However, some evidence suggests that filter methods show a comparable good performance in comparison to other approaches, or can even outperform more computationally complex wrapper or embedded methods (Haury et al., 2011; Saeys et al., 2008). The second stream compares filter methods with regard to the stability of their selection and whether they identify relevant features (Alelyani et al., 2011; Kalousis et al., 2007; Robnik-Šikonja and Kononenko, 2003; Sánchez-Maroño et al., 2007). While these evaluation results are independent of any classification algorithm, the actual performance gains of the FSMs for classification tasks cannot be drawn. A third and more recent stream suggestst to also include the stability of FSMs (similarity of selected feature sets by slightly varied data caused by randomized subsampling) and the algorithm runtimes in the comparison of FSMs (Haury et al., 2011; Saeys et al., 2008; Bommert et al., 2020; Hopf, 2019).

To sum up our review, studies with comprehensive benchmark experiments (i.e., considering multiple criteria such as predictive performance, stability, and runtime) cover only few FSMs. Those studies with larger numbers of FSMs do not focus on different dataset challenges that we, for example, selected for our study. Furthermore, the fields of applications are dominated by studies in the area of bioinformatics with microarray, mass spectrometry datasets, etc. with domain-specific prediction problems (Bolón-Canedo et al.,

2014; Dougherty et al., 2009; Haury et al., 2011; Hua et al., 2009; Kalousis et al., 2007; Lai et al., 2006; Saeys et al., 2007, 2008).

3 Benchmark of FSMs in GNU R

Given the suggestions and the limitations of earlier benchmark studies, we developed a holistic benchmark approach for FSMs that we present below. This benchmark method draws on approaches of earlier works and considers four evaluation criteria.

For the benchmark of the FSMs we first generated synthetic data covering four scenarios of dataset challenges (noisy and redundant data, few observations with many features and imbalanced classes). Then, we applied the FSMs to the data under these scenarios and computed evaluation metrics based on the outcomes. Finally, we analyzed the results statistically.

3.1 Evaluation scenarios and datasets

According to the four scenarios of datset challenges, we created 13 artificial datasets that form the base of our benchmark experiments (see Table 2). For each dataset, we specified which features are relevant and varied the dataset characteristics like the class distribution, the number of observations, the number of (ir-)relevant, and redundant features. For dataset generation, we used the scikit-learn method "make_classification" (Pedregosa et al., 2011), which is an adaption of Guyon's (2003) algorithm that generates binary classification problems with numerical features of a continuous value range. The method first generates relevant features using clusters with normally distributed points around corners of a multidimensional hypercube whose dimension equals to the specified number of features. For each class, it generates the same number of clusters. Then, the method adds interdependence between features and redundant features through linear combinations of relevant features together with randomly generated noise. Finally, the values of the features are shifted and randomly re-scaled. Datasets that were generated with this method were used in the NIPS 2003 Feature Selection Challange and were used in earlier benchmark studies for FSM (Bommert et al., 2020; Bolón-Canedo et al., 2013; Nogueira et al., 2018).

We used the dataset *Baseline* for all later analyses and combine this dataset without special characteristics with the other datasets. For *ClassNoise_**, we added noise to the dependent variable of the baseline dataset in that we swapped the values of one class with the others for the degree of noise (Zhu and Wu, 2004). For *AttNoise_**, we added noise to the features by adding a random number to the original value of each feature. The datasets with noise were only used during feature selection and classifier training, the baseline dataset was used for evaluation (Sáez et al., 2013). We used the datasets *Redundant_** to investigate the suitability of filter methods in the presence of redundant features. The two datasets each contained

Table 2: Summary of the datasets and their properties

		Features			Noise		
Name	Observations	Total	Relevant	Redundant	Class	Attribute	Minority class
Baseline	2,500	100	10	0	0	0	0.5
ClassNoise_1	2,500	100	10	0	0.1	0	0.5
ClassNoise_2	2,500	100	10	0	0.2	0	0.5
ClassNoise_3	2,500	100	10	0	0.3	0	0.5
AttNoise_1	2,500	100	10	0	0	0.1	0.5
AttNoise_2	2,500	100	10	0	0	0.2	0.5
AttNoise_3	2,500	100	10	0	0	0.3	0.5
Redundant_1	2,500	100	10	10	0	0	0.5
Redundant_2	2,500	100	10	20	0	0	0.5
Imbalanced_1	2,500	100	10	10	0	0	0.2
Imbalanced_1	2,500	100	10	10	0	0	0.1
Dimensionality_1	2,500	500	15	0	0	0	0.5
Dimensionality_2	500	1000	30	0	0	0	0.5

ten relevant features and ten and twenty redundant features, respectively. The datasets *Imbalanced_** had a class distribution where the smaller class has a relative frequency of 0.2 and 0.1 respectively. This allowed to evaluate the suitability of filter methods on imbalanced datasets. *Dimensionality_** were datasets with many features compared to the number of observations (i.e., large p small n problem). In *Dimensionality_1*, we increased the number of features to 500 and in *Dimensionality_2*, we set the number of features higher than the number of observations, which is a typical problem of small datasets. For each dataset, we randomized the order of columns and rows to avoid unintended disturbance of our experiments.

3.2 Measurement of performance metrics

We used four performance metrics to evaluate the FSMs. Each metric is explained below.

3.2.1 Predictive performance

All earlier studies that evaluate FSMs in combination with ML algorithms consider the predictive performance, mostly as the core criterion (see Table 1). Thereby, a wide variety of ML algorithms are used to obtain predictions. Most frequently, Support Vector Machine (SVM) (17 studies), instance-based learners (15 studies), Naïve Bayes (NB) (12 studies) were used.

Our study employs three well-known classification algorithms that belong to different categories of learners[1]: NB, Random Forest (RF), and SVM. NB is a classifier based on the application of Bayes' theorem and assumes, for simplicity, that the features are stochastically independent of each other (Dougherty, 2013). Based on the feature values, the classifier calculates conditional probabilities for each new observation for

[1]Another quite popular class of ML algorithms is neural networks. Following our argumentation in the introduction, we do not consider them in our study. In addition to the already reviewed organizational issues, two reasons led us to focus on other ML methods: First, neural networks depend on large amounts of training data. Since business data analysis often has to work with small data sets, the use of neural networks is often limited. Second, in earlier benchmark experiments with real data sets (Fernández-Delgado et al., 2014), neural networks did not clearly outperform the predictive performance of RF and SVM.

each class and assigns the class with the highest probability to the observation. We use the implementation of Meyer et al. (2019). The RF algorithm composes several decision trees whose results are aggregated to determine the final outcome. This process of training multiple versions of a classifier on bootstrap samples and then aggregating them is called bootstrap aggregation or "bagging" (Breiman, 2001). To reduce the correlation of the resulting trees, RF uses just a random sample of the feature sets at every split in the tree. We use the implementation of Liaw and Wiener (2002) with the standard parameter of 500 trees. SVMs (Vapnik and Vapnik, 1998) are a class of learning algorithms that search for a hyperplane in the feature space as a decision boundary that separates the data points with largest possible distance. The complete separation of data points by a hyperplane is only possible in the case of linearly separable data. In the case of nonlinearly separable data, a kernel function is used to transform the data into a higher-dimensional space. We use the SVM implementation of Meyer et al. (2019) with a radial basis function kernel, with the parameters $cost = 1$ and $\gamma = (number\ of\ features)^{-1}$.

To assess the predictive performance, we compare the true classes of the test set with the predicted classes and use the Receiver Operating Characteristic (ROC) curve. The curve graphically represents the trade-off between the true positive and the false postive rate for different classification thresholds in a unit square. Based on the curve, the Area Under ROC Curve (AUC) is a reliable classification performance metric that ranges between 0 and 1, where 0.5 represents random classification and values above indicate a better predictive performance. The majority of earlier FSM benchmark studies (see Table 1) use weaker metrics that simply count correct or wrong classified examples (e.g., accuracy, precision, recall, F-measure). We decided to use AUC, because it is not biased by the class distribution of the dependent variable (Fawcett, 2006). Therefore, AUC values can be compared across different classification problems.

We used stratified 10-fold cross-validation to estimate performance metrics (Kohavi, 1995). With this apporach, the data is divided into ten equally sized subsets, taking into account the class distribution. In ten iterations, each of these subsets were used as the test dataset and the remaining nine subsets as the training dataset for feature selection and model training. We estimate the predictive performance by computing the mean of the ten individual values. Given that the observations are allocated into the ten folds using random sampling, the model performance is affected by this sampling. To lower this effect, we repeated the stratified tenfold cross-validation five times. This means that each filter method was applied a total of 50 times per dataset and the 50 values of the performance metric were averaged.

3.2.2 Number of relevant features selected

As a second criterion, we propose to consider the number of relevant features that each FSM selected. This metric is common to all studies that do not consider classification algorithms in their benchmark of FSMs (see Table 1). Given that we employ synthetic data for the comparison of filter methods, we know in

advance what features are relevant, redundant or irrelevant.

3.2.3 Stability of selected feature sets

Due to random components in the FSMs, and given that only subsets of data are used to evaluate them, several runs of FSMs might produce different feature sets that cause variations in the ML models. To measure the stability, we suggest to use the distance metric recently suggested by Nogueira et al. (2018). They suggest a metric that overcomes several undesirable properties of other pairwise similarity or frequency based metrics that were used in earlier studies that compare FSMs.

Given $X_1, \ldots, X_p$ available features (p is the number of available features) in m datasets, a FSM obtains $V_1, \ldots, V_m$ sets of chosen features based on the datasets. When h_j denotes the number of sets that contain feature X_j so that h_j is the absolute frequency with which feature X_j is chosen and $q = \sum_{j=1}^{p} h_j = \sum_{i=1}^{m} |V_i|$ is the average number of features chosen for the m datasets, then

$$\hat{\mathcal{N}}(V) = 1 - \frac{\frac{1}{p}\sum_{j=1}^{p} \frac{m}{m-1}\frac{h_j}{m}\left(1 - \frac{h_j}{m}\right)}{\frac{q}{mp}(1 - \frac{q}{mp})}$$

is a stability metric for a FSM that is fully defined, is strict monotonic, has upper and lower bounds, has its maximum value when all feature sets are equal, and is corrected for chance (Nogueira et al., 2018).

3.2.4 Runtime

Finally, the time (e.g., in seconds) that the execution of each FSM takes in an analysis is a practically relevant criterion. Especially in the case of large datasets this matters as a selection criterion for practical use.

Our study focuses on selected dataset characteristics, but not on the size of the datasets, as this was already subject of earlier studies (Bommert et al., 2020; Saeys et al., 2008; Aphinyanaphongs et al., 2014). Thus, we include the criterion to complete our picture on the investigated methods.

For the benchmark experiments, we used an instance of Amazon Elastic Compute Cloud (Amazon EC2) and kept the hardware configuration similar throughout all experiments. The instances were of type *m5.2xlarge*, had eight virtual CPUs and 32 gibibibytes of RAM. The software environment was Ubuntu 18.04 LTS and we used R version 3.6.0.

4 Filter methods in GNU R

The focus of our benchmark study is the statistical programming environment GNU R, which is open source software and has gained high popularity in academia and practice[2]. We searched the Comprehensive

[2]See Robert A. Muenchen's website for latest popularity figures: `http://r4stats.com/articles/popularity/` (last access on August 18, 2020)

R Archive Network (CRAN), a repository for open source R software libraries, for implemented FSMs that could be used by analysts and ML users. We found the packages *CORElearn*, *FSelector*, *mlr*, and *praznik* that in total contain 58 unique methods. We considered the FSMs only once, even if they were implemented in multiple methods. One exception is the method *gain ratio*, because *FSelector* and *CORElearn* use different data discretization methods. We excluded the *m:variance* filter from our benchmark, which identifies relevant features based on their variance. As our experiments are based on synthetically generated data, this method almost perfectly separated relevant from irrelevant ones which we consider is not the true behavior of the method with real data. Table 3 lists all considered FSMs with a brief description and, if we found, a literature reference. We denote the package that contains a method with the leading letters *C, F, m,* and *p* respectively. If there is no literature reference given, we point readers to the package documentation in the respective library.

Table 3: Feature filter methods considered in the benchmark study

Name	Description	Ref.	Noise		Cost	Multi-
			Class	Attribute	sensitive	variate
C:Accuracy	Filter based on Accuracy of resulting split					
C:DistAngle	Cosine of angular distance between splits					
C:DistAUC	AUC distance between splits					
C:DistEuclid	Euclidean distance between splits					
C:DistHellinger	Hellinger distance between class distributions in branches					
C:DKM	A measure that is suitable for two class problems	(Dietterich et al., 1996)				
C:DKMcost	Cost-sensitive variant of DKM	(Robnik-Šikonja, 2003)			×	
C:EqualDKM	DKM with equal weights for splits					
C:EqualGini	Gini index with equal weights for splits					
C:EqualHellinger	Two equally weighted splits based Hellinger distance					
C:EqualInf	Information gain with equal weights for splits	(Hunt et al., 1966)				
C:GainRatio	Gain ratio, which is normalized information gain to prevent bias to multi-valued attributes	(Quinlan, 1986)				
C:GainRatioCost	Cost-sensitive variant of GainRatio	(Robnik-Šikonja, 2003)			×	
C:Gini	Gini-index of the attributes	(Breiman, 2001)				
C:ImpurityEuclid	Euclidean distance as impurity function on within node class distributions					
C:ImpurityHellinge	Hellinger distance as impurity function on within node class distributions					
C:InfGain	Information Gain as used in the original decision tree	(Quinlan, 1986)				
C:MDL	Minimum Description Length, a method with favorable bias for multi-valued and multi-class problems	(Kononenko, 1994)				
C:MDLsmp	Cost-sensitive variant of MDL where costs are introduced through sampling	(Robnik-Šikonja, 2003)			×	
C:MyopicReliefF	Myopic version of the 'ReliefF' algorithm resulting from assumption of no local dependencies and attribute dependencies upon class	(Kononenko, 1994)	×	×		
C:Relief	Calculates scores for each feature, based on the Euclidean distance to nearest neighbor training instance pairs	(Kira and Rendell, 1992)	×	×		×
C:ReliefFavgC	Cost-sensitive 'ReliefF' version with average costs	(Robnik-Šikonja, 2003)	×	×	×	×
C:ReliefFbestK	'ReliefF' variant testing all possible k nearest instances for each feature and returns the highest score	(Robnik-Šikonja, 2003)	×	×		×
C:ReliefFdistance	'ReliefF' variant where k nearest instances are weighed directly with its inverse distance from the selected instance	(Robnik-Šikonja, 2003)	×	×		×
C:ReliefFequalK	'ReliefF' algorithm where k nearest instances have equal weight	(Robnik-Šikonja, 2003)	×	×		×
C:ReliefFexpC	Cost-sensitive 'ReliefF' algorithm with expected costs	(Robnik-Šikonja, 2003)	×	×	×	×

Name	Description	Ref.	Noise		Cost	Multi-
			Class	Attribute	sensitive	variate
C:ReliefFexpRank	'ReliefF' algorithm where k nearest instances have weight exponentially de-creasing with increasing rank; ranks are determined by the increasing (Manhattan) distance from the selected instance; conditional dependencies among attributes are taken into account					×
C:ReliefFmerit	'ReliefF' algorithm where for each random instance the merit of each feature is normalized by the sum of differences in all attributes	(Robnik-Šikonja, 2003)	×	×		×
C:ReliefFpa	Cost-sensitive 'ReliefF' algorithm with average probability	(Robnik-Šikonja, 2003)	×	×	×	×
C:ReliefFpe	Cost-sensitive 'ReliefF' algorithm with expected probability	(Robnik-Šikonja, 2003)	×	×	×	×
C:ReliefFsmp	Cost-sensitive 'ReliefF' algorithm with cost sensitive sampling	(Robnik-Šikonja, 2003)	×	×	×	×
C:ReliefFsqr-Distance	'ReliefF' variant where k nearest instances are weighed with its inverse square distance from the selected instance	(Robnik-Šikonja, 2003)	×	×		×
C:ReliefKukar	Cost-sensitive 'Relief' variant	(Kukar et al., 1999)	×	×	×	×
C:UniformAccuracy	Accuracy with uniform priors					
C:UniformDKM	DKM measure with uniform priors					
C:UniformGini	Gini index with uniform priors					
C:UniformInf	Information gain with uniform priors					
F:cfs	The algorithm finds attribute subset using correlation and entropy measures for continuous and discrete data	(Hall, 1999)	×	×		×
F:chi.squared	The algorithm finds weights of discrete attributes basing on a chi-squared test	(Liu and Setiono, 1995)				
F:consistency	The algorithm finds attribute subset using consistency measure for continuous and discrete data	(Dash et al., 2000)	×			×
F:gain.ratio	The algorithms find weights of discrete attributes basing on their correlation with continuous class attribute	(Quinlan, 1986)				
F:oneR	Find weights of discrete attributes basing on very simple association rules involving only one attribute in condition part	(Holte, 1993)				
F:random.forest.-importance	The algorithm creates a weighting of the features using the Random Forest algorithm.	(Breiman, 2001)				×
F:relief	The algorithm calculates scores for each feature, based on the Euclidean distance to nearest neighbor training instance pairs	(Kira and Rendell, 1992)	×	×		×
F:symmetrical.-uncertainty	Weighting of discrete features based on the correlation to the target variable; unlike information gain, no preference for features with many values	(Yu and Liu, 2003)				
m:anova	The algorithm finds weights of features based on an analysis of variance.	(Bommert et al., 2020)				
m:auc	AUC filter for binary classification problems. Weight of a feature is based on achieved AUC when used directly and separately from other features for prediction.	(Bommert et al., 2020)				
m:kruskal.test	The algorithm finds weights of features based on a Kruskal test.	(Bommert et al., 2020)				
m:ranger_impurity	Variable importance based on ranger impurity importance					×
m:ranger_permutation	Variable importance based on ranger permutation importance					×
p:CMIM	First selects the feature with the greatest maximum mutual information with the target variable, then featires are iteratively selected, which supply most information about the target variable, whereby the information of the features already selected is considered	(Fleuret, 2004)				×
p:DISR	Normalized version of JMI	(Meyer and Bontempi, 2006)				×
p:JIM	Joint impurity filter					×
p:JMI	First selects feature with the greatest maximum mutual information with the target variable, then features are selected iteratively, which maximize the cumulative summation of the common mutual information with the already selected feature subset	(Yang and Moody, 2000)				×
p:JMIM	Modification of JMI using minimal common information about already selected features instead of a sum	(Bennasar et al., 2015)				×

Name	Description	Ref.	Noise		Cost	Multi-
			Class	Attribute	sensitive	variate
p:MIM	Calculates the mutual information between all features and the target variable	(Battiti, 1994)				×
p:MRMR	The method first selects the feature with the greatest maximum mutual information with the target variable. Afterwards features are iteratively selected on the basis of measures of relevance and redundancy	(Peng et al., 2005)				×
p:NJMIM	Normalized version of JMIM	(Bennasar et al., 2015)				×

With regard to the investigated scenarios of dataset challenges of low data quality and small datasets, we determined whether the methods are suitable for applying it to noisy data (i.e., class and attribute noise) and imbalanced class distributions, based on the method documentation. For this, we reviewed the method documentations in order to find respective descriptions. In addition, we determined whether each methods considers features individually or in combination (multivariate). Multivariate methods can be assumed to be more suitable to exclude redundant features. In total, we found 16 methods with special support for class noise, 15 methods for attribute noise, 9 methods that are cost-sensitive which means they have special support for imbalanced class problems, and 27 methods that are multivariate, whereas 31 were not.

The nine cost-sensitive methods require a cost matrix as an additional parameter to be considered in the feature selection. We followed the recommendation of Robnik-Šikonja (2003) and weighted the misclassification of the minority class 20 times more than the misclassification of the majority class. When data sets are imbalanced, there is usually a greater interest in correctly classifying the minority class, so higher costs are assigned to misclassification of the minority class. The nine cost-sensitive methods were used exclusively for the two datasets with an imbalanced class distribution, since the application of these methods with a cost matrix is not useful when the class distribution is balanced Robnik-Šikonja (2003).

5 Results

We organize the presentation of the results in three steps. First, we present the results of each evaluation with respect to the four dataset scenarios that are (I) class and attribute noise, (II) redundant features (both are data quality problems in the data), (III) imbalanced data, and (IV) more features than observations (both are problems of small datasets). Second, we test whether methods that expose special support in their documentation for one of the dataset scenario outperform other methods with this respect. Finally, we make a multi-criteria comparison of the FSMs considering all evaluation criteria to obtain an overall comparison of the benchmarked methods. Before we start describing the results, we briefly introduce the method used to analyze the results.

5.1 Analysis of benchmark results

We analyze the benchmark results using a multiple linear regression with ordinary least squares estimation and robust standard errors (White, 1980; Zeileis, 2004). This approach is common when investigating several control variables on an outcome variable (Rawlings et al., 1998). For the analyses, we use the following model specification:

$$Y^{\gamma} = \beta_0^{\gamma} + \beta_1^{\gamma} * FSM + \beta_2^{\gamma} * C_1 + \ldots + \beta_m^{\gamma} * C_{m-1} + \epsilon$$

The dependent variable is one of the evaluation criteria γ, namely $\gamma = \{$ *predictive performance (AUC), number of relevant features, stability, runtime* $\}$. We dedicate separate subsections to each of the dependent variables in the following.

FSM is a categorical variable with a dummy-encoding that contains the name of the FSM (see Table 3 for the full list). In total, our benchmark comprises 58 methods. Nine of them are variants of other methods that add cost-sensitivity to better handle class imbalance (Robnik-Šikonja, 2003). Given that in a uniform class distribution, the results of these are similar to the original methods, we only computed results for the imbalanced dataset scenario for these cost-sensitive variants. For the analysis of the results, we consider no feature selection as a reference case. This means that the coefficients of the estimates labeled with "method*" are deviations from a prediction model without feature selection.

$C_1, \ldots, C_{m-1}$ are control variables that help us to quantify the impact of model characteristics that we vary during the simulation experiments. Table 4 lists all used control variables together with their values. These values result from the generated datasets (see subsection 3.1), bold values are the respective baseline value in the models.

Table 4: Control variables in the statistical models for benchmark analyses

Control variable		Explanation	Values
C_1	classnoise	Percent of noise added to the dependent variable	**0**, 0.1, 0.2, 0.3
C_2	attributenoise	Percent of noise added to the features	**0**, 0.1, 0.2, 0.3
C_3	classifier	Supervised ML algorithm used	**NB**, SVM, RF
C_4	num_redundant_features	Number of redundant features	**0**, 10, 20
C_5	minClassDev	Deviation of the equal distribution in percentage points, multiplied with factor 10.	**0**, 3, 4
C_6	relFeatObs	Relation of the number of features p to the number of observations n	$\frac{\mathbf{100}}{\mathbf{2500}}$, $\frac{500}{2500}$, $\frac{1000}{500}$

The unit of analysis for the regression models are the performances of each method under a certain combination of control variables and one fold of the 10-fold cross validation. For all analyses, we kept the same random allocation of data examples in the cross-folds to ensure comparability. For regression analysis this leads to relatively large samples (745-5250 measurements), which quickly lead to statistically significant results. For completeness, we report the levels of significance in this paper, but focus our evaluation on

effect sizes and ranks.

The analysis method allows to compare the performance metrics with the reference case (no feature selection), but cannot express the performance differences between single FSMs. This is, however, without detriment given that such a comparison would allow little conclusion to be drawn due to the relatively high variance and closely divergent results of several methods. In addition, such a comparison is also not recommended as it misleads conclusions (Demšar, 2006).

5.2 Predictive performance

We computed linear models based on the predictive performance for the four dataset scenarios and show the results in Table 5. The reference case—a model trained with the NB classifier, without feature selection, using the Baseline data for training—shows classification performance that is higher than $AUC = 0.5$ in all four scenarios, which indicates a classification performance that is better than random.

The estimated coefficients in Table 5 show the performance deviation from the base case, which means that all methods with a positive estimate that is significantly different from zero (as indicated by the star notation of the significance levels) showed a better performance than no feature selection. The overall best performance in this analysis showed *F:random.forest.importance, p:DISR* in all scenarios. *p:ranger_impurity, p:ranger_permutation*, and *p:JMI* perform very well in some scenarios.

The models explain between 65 and 77 percent of variance (Adjusted R^2) and the control variables show meaningful estimates: The use of RF and SVM improves the classification by several percentage points compared to the base case of NB. This effect is particularly strong in scenario II (Redundant). In Scenario I (Noise), the performance decreases strongly with noise (if the dependent variable is noisy much more than if the features are noisy), with very strong noise the estimated AUC decreases as expected below 0.5. In Scenario III (Imbalanced), the performance decreases with increasing skew of the dependent variable distribution. In scenario IV (Dimensionality), the performance decreases with an increasing ratio of the number of features to the number of observations. All in all, the estimates of the control variables are as we expected and the model quality is sufficient. This allows us to analyze the effects of the FSMs on the model performance.

5.3 Percentage of relevant features

For the percentage of relevant features that each FSM selects, we computed linear models for each dataset scenario. We list the complete model in Table 10 in the appendix (on page 27) and illustrate the estimated loss of performance (here: percentage of relevant features selected) compared with no feature selection in Figure 1. In order to compare the number of relevant features selected in the four dataset scenarios, we computed the rank of each FSM for each scenario separately and we ordered the methods in the figure

Table 5: Models for the criterion 'predictive performance' (AUC) in the four dataset scenarios

	Noise	Redundant	Imbalanced	Dimensionality
(Intercept)	0.60 (0.00)***	0.56 (0.01)***	0.62 (0.01)***	0.58 (0.00)***
FSMs				
C:Accuracy	0.02 (0.00)***	0.03 (0.01)***	−0.01 (0.01)	0.06 (0.01)***
C:DistAngle	−0.01 (0.00)***	−0.01 (0.01)	−0.02 (0.01)**	0.03 (0.01)***
C:DistAUC	0.02 (0.00)***	0.03 (0.01)***	0.00 (0.01)	0.06 (0.01)***
C:DistEuclid	0.02 (0.00)***	0.03 (0.01)***	0.00 (0.01)	0.06 (0.01)***
C:DistHellinger	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.08 (0.01)***
C:DKM	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.08 (0.01)***
C:DKMcost			0.01 (0.01)	
C:EqualDKM	−0.05 (0.00)***	−0.04 (0.01)***	−0.04 (0.01)***	−0.01 (0.01)
C:EqualGini	−0.05 (0.00)***	−0.04 (0.01)***	−0.04 (0.01)***	−0.01 (0.01)
C:EqualHellinger	−0.05 (0.00)***	−0.04 (0.01)***	−0.03 (0.01)**	−0.01 (0.01)
C:EqualInf	−0.05 (0.00)***	−0.04 (0.01)***	−0.04 (0.01)***	−0.01 (0.01)
C:GainRatio	−0.04 (0.00)***	−0.04 (0.01)***	−0.03 (0.01)**	−0.01 (0.01)
C:GainRatioCost			−0.03 (0.01)***	
C:Gini	0.05 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.07 (0.01)***
C:ImpurityEuclid	−0.05 (0.00)***	−0.04 (0.01)***	−0.04 (0.01)***	−0.01 (0.01)
C:ImpurityHellinger	−0.05 (0.00)***	−0.04 (0.01)***	−0.04 (0.01)***	−0.01 (0.01)
C:InfGain	0.05 (0.00)***	0.06 (0.01)***	0.05 (0.01)***	0.08 (0.01)***
C:MDL	0.05 (0.00)***	0.06 (0.01)***	0.05 (0.01)***	0.08 (0.01)***
C:MDLsmp			−0.00 (0.01)	
C:MyopicReliefF	0.02 (0.00)***	0.03 (0.01)***	0.01 (0.01)	0.06 (0.01)***
C:Relief	−0.05 (0.01)***	−0.00 (0.02)	−0.04 (0.01)***	−0.05 (0.01)***
C:ReliefFavgC			−0.01 (0.01)	
C:ReliefFbestK	−0.04 (0.01)***	−0.00 (0.02)	−0.01 (0.01)	−0.05 (0.01)***
C:ReliefFdistance	0.04 (0.01)***	0.06 (0.01)***	0.05 (0.01)***	0.05 (0.01)***
C:ReliefFequalK	−0.01 (0.00)***	0.03 (0.01)*	0.01 (0.01)	−0.00 (0.01)
C:ReliefFexpC			−0.01 (0.01)	
C:ReliefFexpRank	0.00 (0.00)	0.04 (0.01)***	0.02 (0.01)*	0.02 (0.01)**
C:ReliefFmerit	0.00 (0.00)	0.04 (0.01)***	0.02 (0.01)**	0.02 (0.01)***
C:ReliefFpa			0.04 (0.01)***	
C:ReliefFpe			0.04 (0.01)***	
C:ReliefFsmp			−0.02 (0.01)	
C:ReliefFsqrDistance	0.04 (0.01)***	0.06 (0.01)***	0.05 (0.01)***	0.05 (0.01)***
C:ReliefKukar			−0.01 (0.01)	
C:UniformAccuracy	0.02 (0.00)***	0.03 (0.01)**	0.00 (0.01)	0.06 (0.01)***
C:UniformDKM	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.08 (0.01)***
C:UniformGini	0.05 (0.00)***	0.06 (0.01)***	0.05 (0.01)***	0.07 (0.01)***
C:UniformInf	0.05 (0.00)***	0.06 (0.01)***	0.05 (0.01)***	0.08 (0.01)***
F:cfs	0.03 (0.00)***	0.02 (0.01)*	0.03 (0.01)***	0.04 (0.01)***
F:chi.squared	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.05 (0.01)***
F:consistency	−0.00 (0.00)	−0.04 (0.01)***	−0.00 (0.01)	0.02 (0.01)***
F:gain.ratio	0.04 (0.00)***	0.06 (0.01)***	0.05 (0.01)***	0.05 (0.01)***
F:oneR	0.02 (0.00)***	0.04 (0.01)***	−0.01 (0.01)	0.03 (0.01)***
F:random.forest.importance	**0.11** (0.01)***	**0.10** (0.01)***	**0.13** (0.01)***	**0.12** (0.01)***
F:relief	−0.06 (0.01)***	−0.13 (0.01)***	−0.06 (0.01)***	−0.06 (0.01)***
F:symmetrical.uncertainty	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.05 (0.01)***
m:anova	−0.02 (0.00)***	−0.03 (0.01)*	−0.04 (0.01)***	0.03 (0.01)***
m:auc	−0.03 (0.00)***	−0.04 (0.01)**	−0.04 (0.01)***	0.02 (0.01)*
m:kruskal.test	−0.03 (0.00)***	−0.04 (0.01)**	−0.04 (0.01)***	0.02 (0.01)**
m:ranger_impurity	**0.10** (0.01)***	**0.09** (0.01)***	0.06 (0.01)***	**0.12** (0.01)***
m:ranger_permutation	**0.10** (0.01)***	**0.09** (0.01)***	**0.12** (0.01)***	**0.11** (0.01)***
p:CMIM	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.05 (0.01)***
p:DISR	**0.10** (0.01)***	**0.09** (0.01)***	**0.08** (0.01)***	**0.13** (0.02)***
p:JIM	**0.07** (0.01)***	0.08 (0.01)***	0.06 (0.01)***	**0.11** (0.01)***
p:JMI	**0.07** (0.01)***	**0.09** (0.01)***	**0.07** (0.01)***	**0.12** (0.02)***
p:JMIM	0.04 (0.00)***	0.07 (0.01)***	0.03 (0.01)**	0.07 (0.01)***
p:MIM	0.04 (0.00)***	0.06 (0.01)***	0.04 (0.01)***	0.05 (0.01)***
p:MRMR	−0.02 (0.00)***	−0.08 (0.01)***	−0.04 (0.01)***	0.01 (0.01)
p:NJMIM	0.06 (0.01)***	0.08 (0.01)***	0.05 (0.01)***	0.08 (0.01)***
Controls				
classnoise	−0.44 (0.01)***			
attributenoise	−0.23 (0.01)***			
classifierRandom Forest	0.04 (0.00)***	0.11 (0.00)***	0.03 (0.00)***	0.05 (0.00)***
classifierSupport Vector Machine	0.04 (0.00)***	0.12 (0.00)***	0.03 (0.00)***	0.05 (0.00)***
num_redundant_features		0.00 (0.00)***		
minClassDev			−0.03 (0.00)***	
relFeatObs				−0.06 (0.00)***
R^2	0.75	0.78	0.65	0.66
Adj. R^2	0.74	0.77	0.65	0.65
Num. obs.	5250	2250	2520	2235
F statistic	288.71	148.30	76.42	80.89

Asterisks indicate statistical significance ($^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$), standard errors are in parentheses

according to their average rank—from left (best) to right (worst).

Among the top five methods according to this evaluation criterion, there is no clear picture which one is best overall, as it depends on the dataset scenario. However, the filters based on RF feature importance measures (*F:random.forest-importance* and *m:ranger_impurity*) together with *F:cfs* seem to perform particularly well in selecting relevant features with noisy and imbalanced data. *p:DISR* is also among the top five methods and performs well across all dataset scenarios.

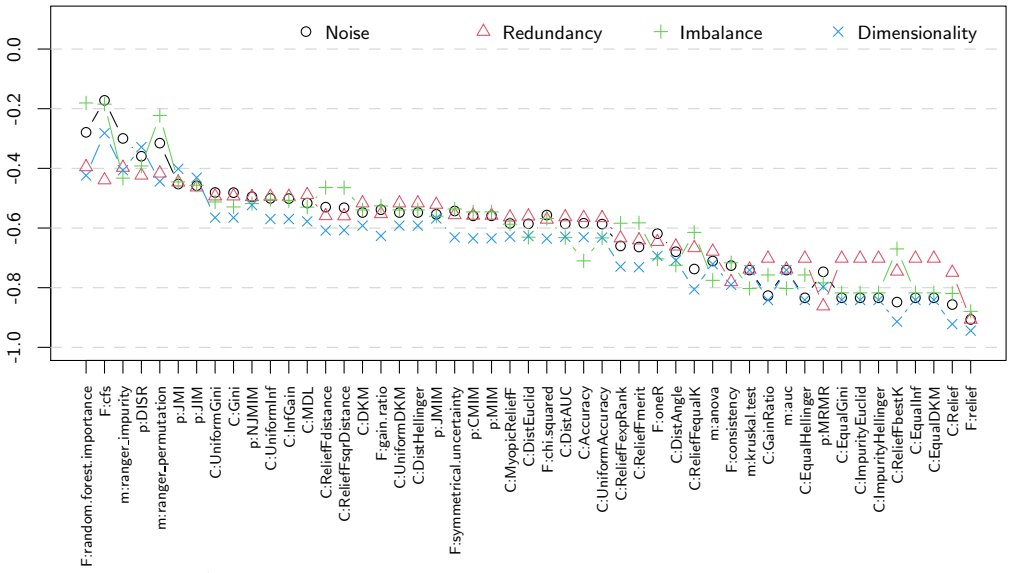

Figure 1: Loss in percentage of relevant features selected per method and dataset scenario

5.4 Stabiliy of selected feature sets

We proceeded similarly for the stability of selected feature sets, which we measured in the distance $\hat{\mathcal{N}}$ of Nogueira et al. (2018), and computed linear models for each dataset scenario (see Table 11 in the appendix on page 28). We illustrate the estimated stability loss per method, compared to no feature selection, ordered by average rank, in Figure 2.

The four dataset scenarios have a strong influence on the stability of selected features sets, especially in the noise scenario, selected feature sets were less stable than in others. Similarly to the earlier evaluation metrics, there is no method that clearly outperforms all others, but apparently, the RF-based methods (*m:ranger_impurity, F:random.forest.importance*, and *m:ranger_permutation*) rank again among the top performing methods. *p:DISR* is also among the top five methods.

5.5 Runtime

For the method runtime, we estimated linear models that we show in Table 6. Given that we ran all tests on the same R instance, we considered no control variable in the models. The methods in the first three

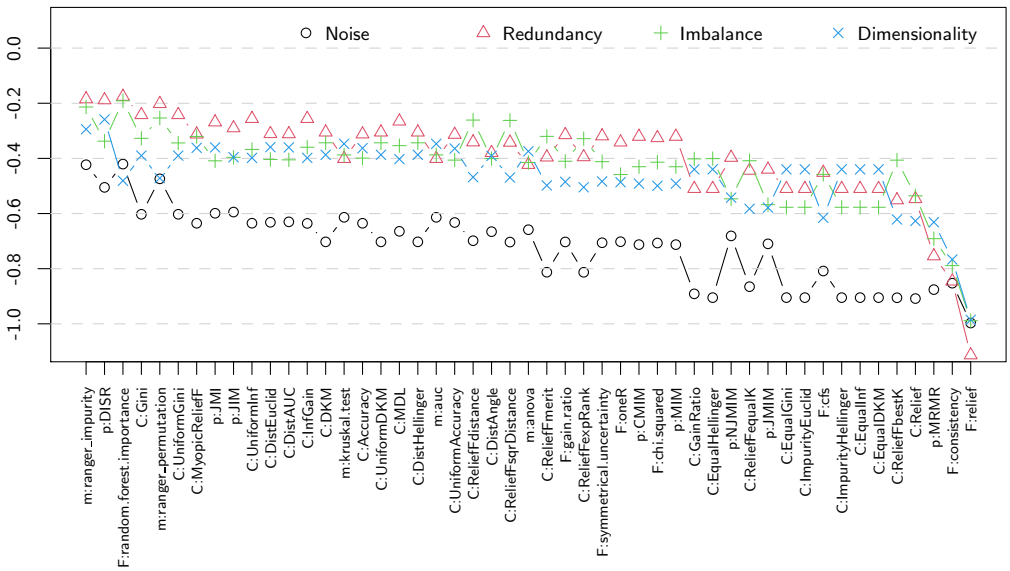

Figure 2: Stability deviation from no selection per method and dataset scenario

scenarios (noise, redundant, imbalanced) have a very low runtime. Thus, only some methods have higher than one second runtime (they need significantly more computational resources). In the dataset scenario with more features than observations, especially the methods *F:cfs, F:consistency*, and *F:relief* showed higher execution times. Methods with the lowest runtime (on average below 5 seconds in all four dataset scenarios) are *m:auc, p:MRMR, p:MIM*, and *p: JIM*.

In addition, we compare the runtime of all methods with the predictive performance, as we would intuitively assume that computationally more expensive methods lead to higher predictive performance. Our experiments, however, could not support this hypothesis, as we found no correlation between method runtime and predictive performance (Pearson's $r(56) = -.08$, $p = .543$). This result is also illustrated in Figure 3.

5.6 FSM characteristics

In our review of the FSMs, we found three properties that make methods appear particularly suitable for certain dataset scenarios. These are robustness to noise, cost sensitivity to counteract class imbalance and the consideration of feature combinations (multivariate methods) for the detection of redundant features. In the following we investigate whether the method properties contribute with statistical evidence to a better handling of the respective dataset challenges. Table 7 gives an overview to these results.

Performance of noise robust methods with noisy data: We tested whether the methods that specified that they support attribute or class noise perform better than other methods. We found neither support for the 11 methods for attribute noise ($t(47) \leq -1.44$, $p \geq .922$, $d \leq -0.51$), nor for the 10 class noise resistant methods ($t(47) \leq -1.65$, $p \geq .947$, $d \leq -0.57$). Given that our experiments with noise do not include

Table 6: Models for the criterion 'method runtime' in the four dataset scenarios

	Noise	Redundant	Imbalanced	Dimensionality
(Intercept)	0.00 (0.00)***	0.00 (0.00)***	0.00 (0.00)***	0.00 (0.00)***
FSMs				
C:Accuracy	0.09 (0.00)***	0.08 (0.00)***	0.08 (0.00)***	0.66 (0.11)***
C:DistAngle	0.10 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.72 (0.12)***
C:DistAUC	0.09 (0.00)***	0.08 (0.00)***	0.08 (0.00)***	0.70 (0.12)***
C:DistEuclid	0.09 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.69 (0.11)***
C:DistHellinger	0.09 (0.00)***	0.08 (0.00)***	0.09 (0.00)***	0.69 (0.11)***
C:DKM	0.09 (0.00)***	0.08 (0.00)***	0.08 (0.00)***	0.74 (0.12)***
C:DKMcost			0.10 (0.01)***	
C:EqualDKM	0.09 (0.00)***	0.09 (0.00)***	0.08 (0.00)***	0.86 (0.21)***
C:EqualGini	0.09 (0.00)***	0.09 (0.00)***	0.08 (0.00)***	0.69 (0.11)***
C:EqualHellinger	0.09 (0.00)***	0.08 (0.00)***	0.08 (0.00)***	0.70 (0.12)***
C:EqualInf	0.09 (0.00)***	0.09 (0.00)***	0.08 (0.00)***	0.69 (0.12)***
C:GainRatio	0.10 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.71 (0.12)***
C:GainRatioCost			0.10 (0.00)***	
C:Gini	0.09 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.70 (0.12)***
C:ImpurityEuclid	0.09 (0.00)***	0.09 (0.00)***	0.08 (0.00)***	0.69 (0.11)***
C:ImpurityHellinger	0.09 (0.00)***	0.08 (0.00)***	0.08 (0.00)***	0.70 (0.11)***
C:InfGain	0.09 (0.00)***	0.09 (0.00)***	0.10 (0.00)***	0.71 (0.12)***
C:MDL	0.12 (0.00)***	0.11 (0.00)***	0.11 (0.00)***	0.78 (0.13)***
C:MDLsmp			0.10 (0.00)***	
C:MyopicReliefF	0.10 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.71 (0.11)***
C:Relief	9.61 (0.08)***	9.25 (0.01)***	9.15 (0.01)***	27.44 (7.20)***
C:ReliefFavgC			9.24 (0.01)***	
C:ReliefFbestK	11.76 (0.08)***	11.43 (0.03)***	11.61 (0.04)***	38.53 (10.28)***
C:ReliefFdistance	1.93 (0.02)***	1.87 (0.02)***	1.85 (0.02)***	7.67 (2.06)***
C:ReliefFequalK	1.90 (0.01)***	1.85 (0.01)***	1.83 (0.01)***	7.54 (2.04)***
C:ReliefFexpC			9.24 (0.01)***	
C:ReliefFexpRank	2.00 (0.02)***	1.88 (0.01)***	1.88 (0.01)***	7.55 (1.99)***
C:ReliefFmerit	9.76 (0.08)***	9.40 (0.01)***	9.33 (0.01)***	27.98 (7.33)***
C:ReliefFpa			9.25 (0.01)***	
C:ReliefFpe			9.24 (0.01)***	
C:ReliefFsmp			9.24 (0.01)***	
C:ReliefFsqrDistance	1.92 (0.02)***	1.86 (0.01)***	1.84 (0.01)***	7.73 (2.10)***
C:ReliefKukar			9.10 (0.01)***	
C:UniformAccuracy	0.09 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.69 (0.11)***
C:UniformDKM	0.10 (0.00)***	0.09 (0.00)***	0.09 (0.00)***	0.72 (0.12)***
C:UniformGini	0.09 (0.00)***	0.08 (0.00)***	0.08 (0.00)***	0.70 (0.12)***
C:UniformInf	0.10 (0.00)***	0.10 (0.00)***	0.09 (0.00)***	0.71 (0.12)***
F:cfs	15.04 (0.37)***	6.68 (1.04)***	14.80 (0.88)***	1500.61 (479.84)**
F:chi.squared	0.61 (0.00)***	0.60 (0.00)***	0.60 (0.00)***	3.13 (0.49)***
F:consistency	157.66 (1.14)***	153.38 (1.06)***	143.97 (2.76)***	403.67 (61.52)***
F:gain.ratio	0.68 (0.00)***	0.68 (0.00)***	0.68 (0.00)***	3.43 (0.52)***
F:oneR	0.67 (0.00)***	0.66 (0.00)***	0.66 (0.00)***	3.53 (0.56)***
F:random.forest.importance	37.76 (0.06)***	36.97 (0.15)***	37.37 (0.08)***	66.49 (13.66)***
F:relief	82.42 (0.22)***	83.11 (0.28)***	82.78 (0.29)***	472.34 (72.48)***
F:symmetrical.uncertainty	0.68 (0.00)***	0.68 (0.00)***	0.69 (0.01)***	3.48 (0.54)***
m:anova	0.20 (0.00)***	0.20 (0.01)***	0.20 (0.01)***	1.00 (0.18)***
m:auc	0.05 (0.00)***	0.05 (0.00)***	0.05 (0.00)***	0.13 (0.02)***
m:kruskal.test	1.34 (0.01)***	1.33 (0.01)***	1.35 (0.01)***	3.97 (0.57)***
m:ranger_impurity	7.85 (0.01)***	7.55 (0.04)***	7.34 (0.09)***	8.96 (1.60)***
m:ranger_permutation	14.10 (0.02)***	13.66 (0.07)***	13.72 (0.06)***	23.40 (4.53)***
p:CMIM	0.09 (0.00)***	0.09 (0.00)***	0.10 (0.00)***	2.30 (0.52)***
p:DISR	0.05 (0.00)***	0.05 (0.00)***	0.05 (0.00)***	0.50 (0.09)***
p:JIM	**0.01** (0.00)***	**0.01** (0.00)***	**0.01** (0.00)***	**0.02** (0.00)***
p:JMI	0.04 (0.00)***	0.04 (0.00)***	0.04 (0.00)***	0.38 (0.07)***
p:JMIM	0.06 (0.00)***	0.06 (0.00)***	0.06 (0.00)***	0.87 (0.16)***
p:MIM	**0.02** (0.00)***	**0.02** (0.00)***	**0.02** (0.00)***	**0.03** (0.00)***
p:MRMR	0.04 (0.00)***	0.03 (0.00)***	0.03 (0.00)***	0.25 (0.04)***
p:NJMIM	0.07 (0.00)***	0.06 (0.00)***	0.07 (0.00)***	1.16 (0.23)***
R^2	1.00	1.00	1.00	0.52
Adj. R^2	1.00	1.00	1.00	0.48
Num. obs.	1750	750	840	745
F statistic	20426.65	12355.62	2913.56	15.25

Asterisks indicate statistical significance (***$p < 0.001$; **$p < 0.01$; *$p < 0.05$), standard errors are in parentheses

class imbalance, we did not include the cost-sensitive variants of methods in this comparison. The statistics of attribute and noise robust methods are almost similar, because there is only one method difference in the set of methods.

Appropriateness of cost-senstive methods on imbalanced data: According to the method descriptions, cost-sensitive filter methods are particularly designed for imbalanced datasets (Robnik-Šikonja, 2003). In our experiments with the imbalanced class dataset scenario, however, the methods had a lower average predictive performance than other methods. The differences are statistically significant with $t(56) \geq 1.80$, $p \leq .038$, $d \geq 0.65$.

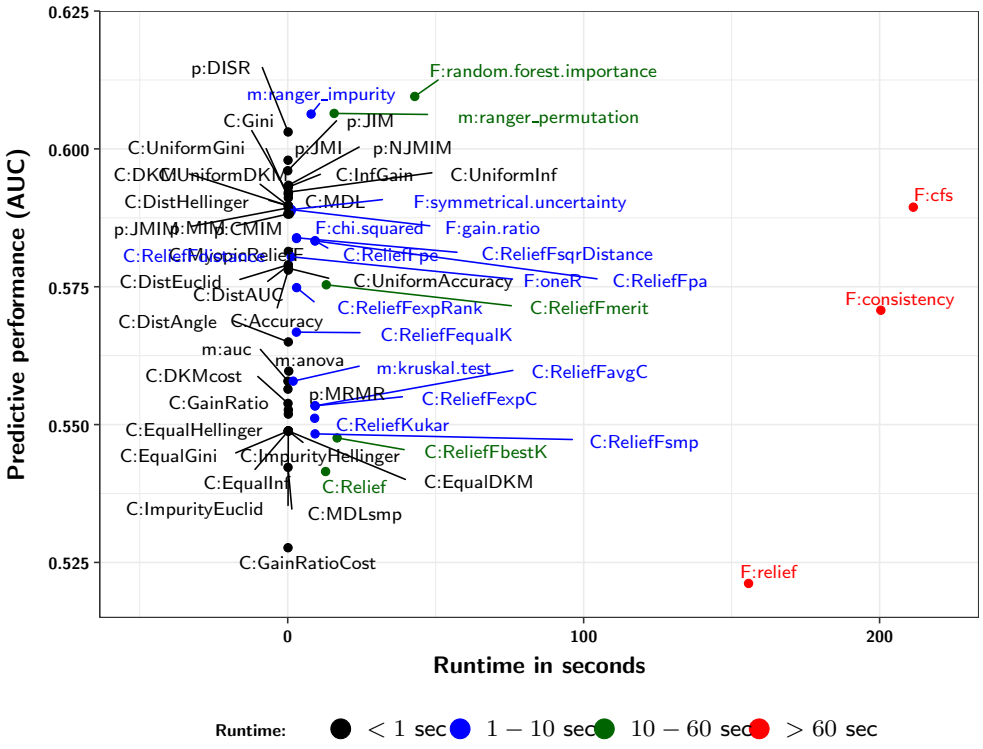

Figure 3: Runtime and predictive performance for each method over all experiments

Performance of multi-dimensional methods on redundant features: In the case of datasets with correlations and redundant features, filter methods that take into account dependencies between features (multivariate methods) should perform significantly better than univariate methods. We find support for this hypothesis based on our experiments, as the multivariate methods performed significantly better than others in our redundant feature dataset scenario with the RF and SVM classifier ($t(47) \geq 1.81$, $p \leq .036$, $d \geq 0.53$).

Table 7: Mean and standard deviation (in brackets) of predictive performance for three method characteristics

Subset of methods	Methods	AUC_{NB}	AUC_{RF}	AUC_{SVM}
Robust for attribute noise	10	0.552 (0.02)	0.577 (0.04)	0.577 (0.04)
Other	39	0.564 (0.02)	0.605 (0.06)	0.612 (0.06)
Robust for class noise	11	0.552 (0.02)	0.577 (0.04)	0.577 (0.04)
Other	38	0.564 (0.02)	0.606 (0.06)	0.613 (0.06)
Cost-sensitive	9	0.555 (0.02)	0.548 (0.03)	0.539 (0.03)
Other	49	0.568 (0.02)	0.606 (0.05)	0.607 (0.06)
Multivariate	20	0.626 (0.03)	0.749 (0.07)	0.766 (0.08)
Univariate	29	0.618 (0.02)	0.714 (0.06)	0.727 (0.07)

5.7 Multi-criteria benchmark of methods

Finally, we compare the benchmark results for the methods with all evaluation criteria in Table 8. This table denotes the predictive performance with a positive / negative sign when the AUC is significantly higher / lower than no feature selection. The top / bottom five methods are denoted with ++ / −−. We denote percentage of relevant features and the stability with the symbols based on the 10, 25, 50, 75, an 90 percent quantile. The runtime performance is denoted with a ++ when the runtime was lower than 0.1 seconds, with + when the runtime was less than 1 seconds, with a 0 when the runtime was between 1 and 10 seconds, with − when it is between 10 and 60 seconds, and with −− when it exceeded 60 seconds.

6 Discussion and Implications

This study provided an overview to existing benchmark studies on FSMs and compared the peformance of 58 FSMs in four typical dataset scenarios that are challenging for ML models, namely: Noisy data, redundant data, imbalanced datasets, and datasets with more features than observations. To examine the performance of FSMs in these scenarios, we created 13 synthetic datasets with characteristics that represent these dataset challenges. We measured the performance of the methods using four criteria that earlier studies have found to be reasonable. The criteria were (1) predictive performance (we used AUC as a reliable metric and tested three ML algorithms), (2) the number of selected relevant features, (3) the stability of the feature sets (for this purpose we used a recently developed distance metric for feature sets (Nogueira et al., 2018) which overcomes limitations of previous metrics), and (4) the method runtime.

Over all dataset scenarios and criteria, those methods that are based on random forest rankings (*F:random.forest.importance*, *m:ranger_impurity*, *m:ranger_permutation*) show a good performance, however their runtime was higher than average in our experiments. Similarly well performed the methods *p:DISR* (a normalized version of *p:JMI*, which also showed a good performance) and *p:JIM*, considering all criteria and dataset scenarios. We compiled a summary of the results for each dataset scenario and criterion in Table 8.

6.1 Implications of the results

Based on our empirical comparison, we can recommend researchers and quantitative modelers to start their search for an appropriate FSM with these filter methods.

We also tested three characteristics that the respective documentation of the methods exhibited as special features. The results of these statistical analyses were suprising: Methods that were described as *noise robust* did not perform better with noisy data than other methods (neither for class nor for attribute noise). Also, the use of cost-senstive methods, which implement approaches to encounter *class imbalance*

Table 8: Benchmark results overview

Method	Pred. performance				Relevant features				Stability				Run-time
	Noi	Red	Imb	Dim	Noi	Red	Imb	Dim	Noi	Red	Imb	Dim	
C:Accuracy	+	+	0	+	−	−	−	0	0	0	0	+	+
C:DistAngle	−	0	−	+	−	−	−	−	0	−	0	0	+
C:DistAUC	+	+	0	+	−	−	−	0	+	0	−	+	+
C:DistEuclid	+	+	0	+	−	0	−	0	+	0	−	++	+
C:DistHellinger	+	+	+	+	0	0	0	0	−	0	+	+	+
C:DKM	+	+	+	+	0	0	0	0	0	+	+	0	+
C:EqualDKM	−	− −	− −	0	− −	− −	− −	− −	− −	− −	− −	−	+
C:EqualGini	−	− −	− −	0	− −	− −	− −	− −	− −	− −	− −	0	+
C:EqualHellinger	−	−	−	0	− −	−	− −	− −	− −	− −	0	−	+
C:EqualInf	−	− −	− −	0	− −	− −	− −	− −	− −	− −	− −	0	+
C:GainRatio	−	−	−	0	− −	− −	− −	− −	− −	− −	0	−	+
C:Gini	+	+	+	+	+	+	0	+	+	+	+	0	+
C:ImpurityEuclid	−	−	− −	0	− −	− −	− −	− −	− −	− −	− −	0	+
C:ImpurityHellinger	− −	−	−	0	− −	− −	− −	− −	− −	− −	− −	−	+
C:InfGain	+	+	+	+	+	+	+	0	0	+	0	0	+
C:MDL	+	+	+	+	0	+	0	0	0	+	0	0	+
C:MyopicReliefF	+	+	0	+	−	−	−	0	0	0	+	+	+
C:Relief	− −	0	−	− −	− −	− −	− −	− −	− −	− −	− −	− −	−
C:ReliefFbestK	−	0	0	− −	− −	− −	−	− −	− −	− −	−	− −	−
C:ReliefFdistance	+	+	+	+	0	0	+	0	0	−	++	−	0
C:ReliefFequalK	−	+	0	0	−	−	−	− −	− −	−	−	− −	0
C:ReliefFexpRank	0	+	0	0	−	−	−	−	−	−	+	− −	0
C:ReliefFmerit	0	+	+	0	−	−	−	−	−	−	+	− −	−
C:ReliefFsqrDistance	+	+	+	+	0	0	+	0	−	−	++	−	0
C:UniformAccuracy	+	+	0	+	−	−	−	−	0	0	−	+	+
C:UniformDKM	+	+	+	+	0	0	0	0	−	0	0	0	+
C:UniformGini	+	+	+	+	+	+	+	+	+	++	0	0	+
C:UniformInf	+	+	+	+	+	+	+	+	0	+	0	0	+
F:cfs	+	+	+	+	++	++	++	++	−	− −	− −	− −	− −
F:chi.squared	+	+	+	+	0	−	0	−	−	0	−	− −	0
F:consistency	0	−	0	0	−	− −	−	− −	− −	− −	− −	− −	− −
F:gain.ratio	+	+	+	+	0	0	0	0	−	0	−	−	0
F:oneR	+	+	0	+	−	−	−	−	0	−	− −	−	0
F:random.forest.importance	++	++	++	++	++	++	++	++	++	++	++	−	−
F:relief	− −	− −	− −	− −	− −	− −	− −	− −	− −	− −	− −	− −	− −
F:symmetrical.uncertainty	+	+	+	+	0	0	0	−	−	0	−	−	0
m:anova	−	−	−	+	−	−	− −	−	0	−	−	+	+
m:auc	−	−	−	+	− −	− −	− −	−	+	−	0	++	++
m:kruskal.test	−	−	−	+	− −	− −	− −	−	+	−	0	++	0
m:ranger_impurity	++	+	+	++	++	++	++	++	++	++	++	++	0
m:ranger_permutation	++	+	++	+	++	++	++	+	++	++	++	−	−
p:CMIM	+	+	+	+	0	0	0	−	−	0	−	−	+
p:DISR	++	++	++	++	++	++	++	++	++	++	+	++	+
p:JIM	+	+	+	+	+	+	+	+	++	+	0	0	++
p:JMI	+	+	++	++	+	+	+	++	+	+	−	+	+
p:JMIM	+	+	+	+	0	0	0	+	−	−	− −	− −	+
p:MIM	+	+	+	+	0	0	0	−	−	0	−	− −	++
p:MRMR	−	− −	−	0	− −	− −	− −	− −	− −	− −	− −	− −	++
p:NJMIM	+	+	+	+	+	0	0	+	0	−	− −	− −	+

(Robnik-Šikonja, 2003), did not achieve better results on imbalanced data than other methods. One reason for this result may be that methods that claim to be particularly suitable for noisy data or imbalanced datasets were developed at a time when the methods that performed well in our benchmark did not exist. Based on this finding, future research should further develop well-performing approaches in our benchmark for these specific dataset scenarios. Conversely, developers and maintainers of methods should be careful in labeling their methods as particularly suitable for a certain dataset scenario.

Only multi-dimensional methods showed a better performance with *redundant features* than other methods. As a consequence of this result, users do not need to apply special methods for noisy data and imbalanced datasets and can use those that have performed well in our benchmark in these scenarios or in general. This saves effort and resources during implementation and when training experts. Only for datasets with redundant features, we recommend to use multi-dimensional methods.

6.2 Limitations and future research

We consider four limitations of our study, which we believe are areas for future research. First, our benchmark is based on synthetic data. This allowed us to deliberately evaluate FSMs regarding the dataset characteristics in the focus of our study, under laboratory conditions and independent of any application field. Although we have used established data generation methods from the ML-field, the data generation is subject to assumptions from which reality will differ. Earlier studies on the comparison of FSMs used synthetic as well as real datasets. Therefore, we motivate future research that validates our results with data from various application domains. Second, we measured the runtime for all our experiments with the same hardware and the same datasets to be valid internally. This measurement is likely to be different in magnitude when the employed hardware and software environment is modified. Given that the focus of our study was the comparison of FSMs with respect to different dataset scenarios, we refer readers who are interested in more detailed analysis of the algorithms runtime with very large datasets to previous studies, e.g., Bommert et al. (2020), Aphinyanaphongs et al. (2014), and Xue et al. (2015). Third, we considered three well-known ML algorithms to obtain the predictive performance. A promising approach for business data analytics is deep learning which is capable to use raw data directly without the need of feature selection (Kraus et al., 2020). However, such neural networks are "data hungry" (Marcus, 2018, p. 6) and therefore only usable in some applications. We motivate future research to examine how classical ML algorithms together with FSMs perform in comparison with deep learning approaches. Fourth, the dataset challenges considered in this study are only a selection of challenges and further ones should be addressed. For example, the problem of missing values usually makes data analysis difficult. Even though there are numerous basic approaches how to deal with missing values, we could not find FSMs that explicitly address this problem. We motivate future research to further develop the methods in this respect.

7 Conclusion

Feature selection is an important task in setting up ML models in business analytics. Identifying most appropriate features for predictive modeling enables the creation of less complex models that are more interpretable in decision situations, can improve model performance, and saves computational effort in later analysis steps. These are particular requirements in big data analytics tasks, not only today, given that digitization will allow measuring a wide variety of new variables in the future.

As feature selection is important, many methods have been suggested so far: In the widely used statistical software environment GNU R, we found over fifty different methods—ready to use by analysts and researchers. A comprehensive comparison of these methods, however, is currently missing, which makes the decision on which method to use difficult.

Our study narrowed this gap and first provided a comprehensive literature review to benchmarks of FSMs. We outlined limitations of current studies and suggested—based on the learnings of earlier benchmark studies—a multi-criteria benchmark method for FSMs, which we applied in the second part of our paper. There, we provided a benchmark of 58 currently available feature filter methods in the GNU R environment. For the comparison of methods, we used four dataset challenges that are typical in ML applications. The method overview and comparison guides the automatic feature selection task of researchers and quantitative modelers. Our benchmark method provides a base for further studies that employ it to case-specific datasets.

Acknowledgements

We appreciate Andreas Weigert's support in implementing early stages of the benchmark analysis and his insightful comments to earlier versions of this manuscript. We thank Thorsten Staake for his feedback on the initial study design.

Appendices

Additional statistical models are listed in Table 10 and Table 11.

Table 10: Models for the criterion 'number of relevant features selected' in the four dataset scenarios

	Noise	Redundant	Imbalanced	Dimensionality
(Intercept)	1.00 (0.00)***	1.00	1.00 (0.00)***	1.00 (0.00)***
FSMs				
C:Accuracy	−0.58 (0.02)***	−0.57 (0.03)***	−0.71 (0.06)***	−0.63 (0.05)***
C:DistAngle	−0.68 (0.02)***	−0.66 (0.02)***	−0.72 (0.03)***	−0.71 (0.03)***
C:DistAUC	−0.59 (0.02)***	−0.56 (0.03)***	−0.63 (0.05)***	−0.63 (0.05)***
C:DistEuclid	−0.59 (0.02)***	−0.56 (0.03)***	−0.63 (0.05)***	−0.63 (0.05)***
C:DistHellinger	−0.55 (0.03)***	−0.52 (0.05)***	−0.54 (0.06)***	−0.59 (0.07)***
C:DKM	−0.55 (0.03)***	−0.52 (0.05)***	−0.54 (0.06)***	−0.59 (0.07)***
C:DKMcost			−0.62 (0.04)***	
C:EqualDKM	−0.83 (0.01)***	−0.70 (0.00)***	−0.82 (0.03)***	−0.84 (0.03)***
C:EqualGini	−0.83 (0.01)***	−0.70 (0.00)***	−0.82 (0.03)***	−0.84 (0.03)***
C:EqualHellinger	−0.83 (0.01)***	−0.70 (0.00)***	−0.76 (0.02)***	−0.84 (0.03)***
C:EqualInf	−0.83 (0.01)***	−0.70 (0.00)***	−0.82 (0.03)***	−0.84 (0.03)***
C:GainRatio	−0.83 (0.01)***	−0.70 (0.00)***	−0.76 (0.02)***	−0.84 (0.03)***
C:GainRatioCost			−0.82 (0.01)***	
C:Gini	−0.48 (0.02)***	−0.49 (0.04)***	−0.53 (0.05)***	−0.57 (0.06)***
C:ImpurityEuclid	−0.83 (0.01)***	−0.70 (0.00)***	−0.82 (0.03)***	−0.84 (0.03)***
C:ImpurityHellinger	−0.83 (0.01)***	−0.70 (0.00)***	−0.82 (0.03)***	−0.84 (0.03)***
C:InfGain	−0.50 (0.03)***	−0.49 (0.04)***	−0.51 (0.05)***	−0.57 (0.06)***
C:MDL	−0.52 (0.03)***	−0.49 (0.04)***	−0.53 (0.06)***	−0.58 (0.07)***
C:MDLsmp			−0.74 (0.02)***	
C:MyopicReliefF	−0.58 (0.02)***	−0.56 (0.03)***	−0.59 (0.03)***	−0.63 (0.05)***
C:Relief	−0.86 (0.01)***	−0.75 (0.03)***	−0.82 (0.01)***	−0.92 (0.01)***
C:ReliefFavgC			−0.70 (0.03)***	
C:ReliefFbestK	−0.85 (0.01)***	−0.75 (0.02)***	−0.67 (0.04)***	−0.91 (0.01)***
C:ReliefFdistance	−0.53 (0.04)***	−0.56 (0.06)***	−0.46 (0.04)***	−0.61 (0.07)***
C:ReliefFequalK	−0.74 (0.02)***	−0.67 (0.03)***	−0.61 (0.01)***	−0.81 (0.03)***
C:ReliefFexpC			−0.70 (0.03)***	
C:ReliefFexpRank	−0.66 (0.02)***	−0.63 (0.04)***	−0.58 (0.01)***	−0.73 (0.04)***
C:ReliefFmerit	−0.66 (0.02)***	−0.64 (0.04)***	−0.58 (0.01)***	−0.73 (0.04)***
C:ReliefFpa			−0.60 (0.00)***	
C:ReliefFpe			−0.60 (0.00)***	
C:ReliefFsmp			−0.71 (0.03)***	
C:ReliefFsqrDistance	−0.53 (0.04)***	−0.56 (0.06)***	−0.46 (0.04)***	−0.61 (0.07)***
C:ReliefKukar			−0.82 (0.01)***	
C:UniformAccuracy	−0.59 (0.02)***	−0.57 (0.03)***	−0.63 (0.05)***	−0.63 (0.05)***
C:UniformDKM	−0.55 (0.03)***	−0.52 (0.05)***	−0.54 (0.06)***	−0.59 (0.07)***
C:UniformGini	−0.48 (0.02)***	−0.49 (0.04)***	−0.51 (0.05)***	−0.57 (0.06)***
C:UniformInf	−0.50 (0.03)***	−0.49 (0.04)***	−0.50 (0.05)***	−0.57 (0.06)***
F:cfs	−0.17 (0.01)***	−0.44 (0.09)***	−0.18 (0.03)***	−0.28 (0.08)***
F:chi.squared	−0.56 (0.02)***	−0.57 (0.03)***	−0.57 (0.04)***	−0.64 (0.06)***
F:consistency	−0.73 (0.02)***	−0.78 (0.03)***	−0.72 (0.03)***	−0.79 (0.04)***
F:gain.ratio	−0.54 (0.02)***	−0.55 (0.04)***	−0.53 (0.05)***	−0.63 (0.07)***
F:oneR	−0.62 (0.02)***	−0.65 (0.03)***	−0.70 (0.04)***	−0.69 (0.04)***
F:random.forest.importance	−0.28 (0.04)***	−0.40 (0.08)***	−0.18 (0.04)***	−0.42 (0.10)***
F:relief	−0.91 (0.01)***	−0.91 (0.01)***	−0.88 (0.01)***	−0.94 (0.01)***
F:symmetrical.uncertainty	−0.54 (0.02)***	−0.56 (0.04)***	−0.54 (0.04)***	−0.63 (0.06)***
m:anova	−0.71 (0.01)***	−0.68 (0.01)***	−0.78 (0.02)***	−0.72 (0.02)***
m:auc	−0.74 (0.01)***	−0.74 (0.01)***	−0.80 (0.02)***	−0.74 (0.02)***
m:kruskal.test	−0.74 (0.01)***	−0.74 (0.01)***	−0.80 (0.02)***	−0.74 (0.02)***
m:ranger_impurity	−0.30 (0.04)***	−0.40 (0.07)***	−0.43 (0.08)***	−0.41 (0.09)***
m:ranger_permutation	−0.32 (0.04)***	−0.42 (0.08)***	−0.22 (0.04)***	−0.44 (0.10)***
p:CMIM	−0.56 (0.02)***	−0.56 (0.04)***	−0.55 (0.05)***	−0.63 (0.06)***
p:DISR	−0.36 (0.04)***	−0.42 (0.08)***	−0.39 (0.09)***	−0.33 (0.12)**
p:JIM	−0.46 (0.04)***	−0.46 (0.06)***	−0.46 (0.07)***	−0.43 (0.10)***
p:JMI	−0.45 (0.04)***	−0.45 (0.07)***	−0.45 (0.08)***	−0.40 (0.10)***
p:JMIM	−0.55 (0.03)***	−0.52 (0.04)***	−0.56 (0.06)***	−0.57 (0.07)***
p:MIM	−0.56 (0.02)***	−0.56 (0.04)***	−0.55 (0.05)***	−0.63 (0.06)***
p:MRMR	−0.75 (0.01)***	−0.86 (0.04)***	−0.78 (0.03)***	−0.80 (0.03)***
p:NJMIM	−0.50 (0.03)***	−0.50 (0.05)***	−0.52 (0.07)***	−0.52 (0.08)***
R^2	0.65	0.48	0.58	0.41
Adj. R^2	0.64	0.44	0.55	0.37
Num. obs.	1750	750	840	745
F statistic	63.71	13.13	18.52	9.76

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 11: Models for the criterion 'stability' in the four dataset scenarios

	Noise	Redundant	Imbalanced	Dimensionality
(Intercept)	1.26 (0.03)***	1.14 (0.02)***	1.06 (0.01)***	1.06 (0.02)***
FSMs				
C:Accuracy	−0.64 (0.03)***	−0.31 (0.02)***	−0.40 (0.02)***	−0.36 (0.02)***
C:DistAngle	−0.67 (0.03)***	−0.38 (0.03)***	−0.40 (0.02)***	−0.39 (0.03)***
C:DistAUC	−0.63 (0.03)***	−0.31 (0.02)***	−0.41 (0.02)***	−0.36 (0.02)***
C:DistEuclid	−0.63 (0.03)***	−0.31 (0.02)***	−0.40 (0.02)***	−0.36 (0.02)***
C:DistHellinger	−0.70 (0.03)***	−0.31 (0.02)***	−0.34 (0.02)***	−0.39 (0.02)***
C:DKM	−0.70 (0.03)***	−0.31 (0.02)***	−0.34 (0.02)***	−0.39 (0.02)***
C:DKMcost			−0.50 (0.04)***	
C:EqualDKM	−0.90 (0.04)***	−0.51 (0.02)***	−0.58 (0.02)***	−0.44 (0.03)***
C:EqualGini	−0.90 (0.04)***	−0.51 (0.02)***	−0.58 (0.02)***	−0.44 (0.03)***
C:EqualHellinger	−0.90 (0.04)***	−0.51 (0.02)***	−0.40 (0.03)***	−0.44 (0.03)***
C:EqualInf	−0.90 (0.04)***	−0.51 (0.02)***	−0.58 (0.02)***	−0.44 (0.03)***
C:GainRatio	−0.89 (0.04)***	−0.51 (0.02)***	−0.40 (0.03)***	−0.44 (0.03)***
C:GainRatioCost			−0.36 (0.02)***	
C:Gini	−0.60 (0.03)***	−0.24 (0.02)***	−0.33 (0.02)***	−0.39 (0.03)***
C:ImpurityEuclid	−0.90 (0.04)***	−0.51 (0.02)***	−0.58 (0.02)***	−0.44 (0.03)***
C:ImpurityHellinger	−0.90 (0.04)***	−0.51 (0.02)***	−0.58 (0.02)***	−0.44 (0.03)***
C:InfGain	−0.64 (0.03)***	−0.26 (0.02)***	−0.36 (0.03)***	−0.40 (0.02)***
C:MDL	−0.66 (0.03)***	−0.27 (0.02)***	−0.35 (0.03)***	−0.40 (0.02)***
C:MDLsmp			−0.70 (0.03)***	
C:MyopicReliefF	−0.64 (0.03)***	−0.31 (0.02)***	−0.32 (0.02)***	−0.36 (0.02)***
C:Relief	−0.91 (0.04)***	−0.55 (0.04)***	−0.54 (0.03)***	−0.63 (0.02)***
C:ReliefFavgC			−0.40 (0.02)***	
C:ReliefFbestK	−0.91 (0.04)***	−0.55 (0.04)***	−0.41 (0.05)***	−0.62 (0.02)***
C:ReliefFdistance	−0.70 (0.03)***	−0.34 (0.02)***	−0.26 (0.03)***	−0.47 (0.03)***
C:ReliefFequalK	−0.87 (0.03)***	−0.44 (0.04)***	−0.41 (0.04)***	−0.58 (0.03)***
C:ReliefFexpC			−0.40 (0.02)***	
C:ReliefFexpRank	−0.81 (0.03)***	−0.39 (0.04)***	−0.33 (0.04)***	−0.50 (0.03)***
C:ReliefFmerit	−0.81 (0.03)***	−0.40 (0.03)***	−0.32 (0.04)***	−0.50 (0.02)***
C:ReliefFpa			−0.22 (0.02)***	
C:ReliefFpe			−0.22 (0.02)***	
C:ReliefFsmp			−0.43 (0.02)***	
C:ReliefFsqrDistance	−0.70 (0.03)***	−0.34 (0.01)***	−0.26 (0.03)***	−0.47 (0.03)***
C:ReliefKukar			−0.54 (0.02)***	
C:UniformAccuracy	−0.63 (0.03)***	−0.31 (0.02)***	−0.41 (0.02)***	−0.36 (0.02)***
C:UniformDKM	−0.70 (0.03)***	−0.31 (0.02)***	−0.34 (0.02)***	−0.39 (0.02)***
C:UniformGini	−0.60 (0.03)***	−0.24 (0.02)***	−0.34 (0.03)***	−0.39 (0.03)***
C:UniformInf	−0.64 (0.03)***	−0.26 (0.02)***	−0.37 (0.03)***	−0.40 (0.02)***
F:cfs	−0.81 (0.03)***	−0.45 (0.01)***	−0.46 (0.01)***	−0.62 (0.05)***
F:chi.squared	−0.71 (0.03)***	−0.33 (0.03)***	−0.41 (0.02)***	−0.50 (0.02)***
F:consistency	−0.85 (0.04)***	−0.85 (0.02)***	−0.79 (0.02)***	−0.77 (0.03)***
F:gain.ratio	−0.70 (0.03)***	−0.31 (0.02)***	−0.41 (0.02)***	−0.49 (0.03)***
F:oneR	−0.70 (0.03)***	−0.34 (0.02)***	−0.46 (0.04)***	−0.49 (0.03)***
F:random.forest.importance	−0.42 (0.04)***	−0.18 (0.04)***	−0.19 (0.03)***	−0.48 (0.09)***
F:relief	−1.00 (0.04)***	−1.11 (0.02)***	−0.99 (0.02)***	−0.99 (0.03)***
F:symmetrical.uncertainty	−0.71 (0.03)***	−0.32 (0.03)***	−0.41 (0.02)***	−0.48 (0.02)***
m:anova	−0.66 (0.03)***	−0.42 (0.02)***	−0.41 (0.02)***	−0.37 (0.03)***
m:auc	−0.61 (0.03)***	−0.40 (0.02)***	−0.39 (0.02)***	−0.35 (0.02)***
m:kruskal.test	−0.61 (0.03)***	−0.40 (0.02)***	−0.39 (0.02)***	−0.35 (0.02)***
m:ranger_impurity	−0.42 (0.04)***	−0.18 (0.03)***	−0.21 (0.02)***	−0.29 (0.05)***
m:ranger_permutation	−0.47 (0.04)***	−0.20 (0.03)***	−0.25 (0.03)***	−0.47 (0.07)***
p:CMIM	−0.71 (0.03)***	−0.32 (0.03)***	−0.43 (0.02)***	−0.49 (0.02)***
p:DISR	−0.51 (0.04)***	−0.19 (0.04)***	−0.34 (0.06)***	−0.26 (0.07)***
p:JIM	−0.59 (0.03)***	−0.29 (0.03)***	−0.40 (0.02)***	−0.40 (0.04)***
p:JMI	−0.60 (0.03)***	−0.27 (0.03)***	−0.41 (0.04)***	−0.36 (0.05)***
p:JMIM	−0.71 (0.03)***	−0.44 (0.02)***	−0.57 (0.03)***	−0.58 (0.03)***
p:MIM	−0.71 (0.03)***	−0.32 (0.03)***	−0.43 (0.02)***	−0.49 (0.02)***
p:MRMR	−0.88 (0.03)***	−0.75 (0.02)***	−0.69 (0.02)***	−0.63 (0.04)***
p:NJMIM	−0.68 (0.03)***	−0.40 (0.02)***	−0.55 (0.03)***	−0.54 (0.04)***
Controls				
classnoise	−1.83 (0.03)***			
attributenoise	−1.17 (0.03)***			
mean_informativeFeatures		−0.01 (0.00)***		
minClassDev			−0.03 (0.00)***	
relFeatObs				−0.09 (0.00)***
R^2	0.85	0.83	0.76	0.72
Adj. R^2	0.84	0.82	0.74	0.70
Num. obs.	1750	750	840	745
F statistic	181.85	68.70	41.50	36.50

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

References

Stephen Adams, Ryan Meekins, and Peter A. Beling. An Empirical Evaluation of Techniques for Feature Selection with Cost. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 834–841, New Orleans, LA, November 2017. IEEE. ISBN 978-1-5386-3800-2. doi: 10.1109/ICDMW.2017.153. URL http://ieeexplore.ieee.org/document/8215748/.

Salem Alelyani, Zheng Zhao, and Huan Liu. A Dilemma in Assessing Stability of Feature Selection Algorithms. In *2011 IEEE International Conference on High Performance Computing and Communications*, pages 701–707, Banff, AB, Canada, September 2011. IEEE. ISBN 978-1-4577-1564-8. doi: 10.1109/HPCC.2011.99. URL http://ieeexplore.ieee.org/document/6063062/.

Omar Alonso. Challenges with label quality for supervised learning. *Journal of Data and Information Quality*, 6(1):2:1–2:3, 2015. doi: 10.1145/2724721. URL https://doi.org/10.1145/2724721.

Yindalon Aphinyanaphongs, Lawrence D. Fu, Zhiguo Li, Eric R. Peskin, Efstratios Efstathiadis, Constantin F. Aliferis, and Alexander Statnikov. A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization: A Comprehensive Empirical Comparison of Modern Supervised Classification and Feature-Selection Methods for Text Categorization. *Journal of the Association for Information Science and Technology*, 65(10):1964–1987, October 2014. doi: 10.1002/asi.23110. URL http://doi.wiley.com/10.1002/asi.23110.

R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, July 1994. doi: 10.1109/72.298224.

P. Baumann, D.S. Hochbaum, and Y.T. Yang. A comparative study of the leading machine learning techniques and two new optimization algorithms. *European Journal of Operational Research*, 272(3):1041–1057, 2019. doi: 10.1016/j.ejor.2018.07.009. URL https://linkinghub.elsevier.com/retrieve/pii/S0377221718306143.

Mohamed Bennasar, Yulia Hicks, and Rossitza Setchi. Feature selection using Joint Mutual Information Maximisation. *Expert Systems with Applications*, 42(22):8520–8532, December 2015. doi: 10.1016/j.eswa.2015.07.007. URL https://linkinghub.elsevier.com/retrieve/pii/S0957417415004674.

Nicholas Berente, Stefan Seidel, and Hani Safadi. Research commentary—data-driven computationally intensive theory development. *Information Systems Research*, 30(1):50–64, 2019. doi: 10.1287/isre.2018.0774.

Bernd Bischl, Michel Lang, Lars Kotthoff, Julia Schiffner, Jakob Richter, Erich Studerus, Giuseppe Casal-

icchio, and Zachary M. Jones. mlr: Machine learning in r. *Journal of Machine Learning Research*, 17 (170):1–5, 2016. URL http://jmlr.org/papers/v17/15-066.html.

Avrim L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2):245–271, December 1997. doi: 10.1016/S0004-3702(97)00063-5. URL //www.sciencedirect.com/science/article/pii/S0004370297000635.

Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and Information Systems*, 34(3):483–519, March 2013. doi: 10.1007/s10115-012-0487-8. URL http://link.springer.com/10.1007/s10115-012-0487-8.

Verónica Bolón-Canedo, Noelia Sánchez-Maroño, Amparo Alonso-Betanzos, J.M. Benítez, and F. Herrera. A review of microarray datasets and applied feature selection methods. *Information Sciences*, 282:111–135, October 2014. doi: 10.1016/j.ins.2014.05.042. URL https://linkinghub.elsevier.com/retrieve/pii/S0020025514006021.

Andrea Bommert, Xudong Sun, Bernd Bischl, Jörg Rahnenführer, and Michel Lang. Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis*, 143:106839, March 2020. doi: 10.1016/j.csda.2019.106839. URL https://linkinghub.elsevier.com/retrieve/pii/S016794731930194X.

Vadim Borisov, Johannes Haug, and Gjergji Kasneci. CancelOut: A Layer for Feature Selection in Deep Neural Networks. In Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning*, Lecture Notes in Computer Science, pages 72–83, Cham, 2019. Springer International Publishing. ISBN 978-3-030-30484-3. doi: 10.1007/978-3-030-30484-3_6.

Michael Franklin Bosu and Stephen G. MacDonell. A taxonomy of data quality challenges in empirical software engineering. In *2013 22nd Australian Software Engineering Conference*, pages 97–106. IEEE, 2013. ISBN 978-0-7695-4995-8. doi: 10.1109/ASWEC.2013.21. URL http://ieeexplore.ieee.org/document/6601297/.

Paula Branco, Luís Torgo, and Rita P. Ribeiro. A Survey of Predictive Modeling on Imbalanced Domains. *ACM Computing Surveys*, 49(2):31:1–31:50, August 2016. ISSN 0360-0300. doi: 10.1145/2907070.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Lujan. Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection. *Journal of Machine Learning Research*, 13:27–66, 2012.

Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, January 2014. doi: 10.1016/j.compeleceng.2013.11.024. URL http://www.sciencedirect.com/science/article/pii/S0045790613003066.

Shiva Darshan and C. D. Jaidhar. Performance Evaluation of Filter-based Feature Selection Techniques in Classifying Portable Executable Files. *Procedia Computer Science*, 125:346–356, January 2018. doi: 10.1016/j.procs.2017.12.046. URL http://www.sciencedirect.com/science/article/pii/S1877050917328107.

M Dash and H Liu. Feature Selection for Classification. *Intelligent Data Analysis*, 1:131–167, 1997.

Manoranjan Dash, Huan Liu, and Hiroshi Motoda. Consistency based feature selection. In *Knowledge Discovery and Data Mining. Current Issues and New Applications*, pages 98–109. Springer, 2000. ISBN 3-540-67382-2.

Thomas H. Davenport and Jeanne G. Harris. *Competing on Analytics: The New Science of Winning*. Harvard Business Press, 2007. ISBN 978-1-4221-0332-6.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006. 05008.

Tom Dietterich, Michael Kearns, and Yishay Mansour. Applying the weak learning framework to understand and improve C4.5. In *Proceedings of the 13. International Conference on Machine Learning*, pages 96–104. Morgan Kaufmann, 1996.

Edward R Dougherty, Jianping Hua, and Chao Sima. Performance of Feature Selection Methods. *Current Genomics*, 10(6):365–374, September 2009. doi: 10.2174/138920209789177629. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2766788/.

Geoff Dougherty. Estimating and comparing classifiers. In *Pattern Recognition and Classification*, pages 157–176. Springer, 2013.

T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

Qi Feng and J. George Shanthikumar. How research in production and operations management may evolve in the era of big data. *Production and Operations Management*, 27(9):1670–1684, 2018. doi: 10.1111/poms.12836. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/poms.12836.

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

Trevor Fitzpatrick and Christophe Mues. An empirical comparison of classification algorithms for mortgage default prediction: evidence from a distressed mortgage market. *European Journal of Operational Research*, 249(2):427–439, 2016. doi: 10.1016/j.ejor.2015.09.014. URL https://linkinghub.elsevier.com/retrieve/pii/S0377221715008383.

François Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

George Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3(Mar):1289–1305, 2003.

Daniel Gartner, Rainer Kolisch, Daniel B. Neill, and Rema Padman. Machine learning approaches for early DRG classification and resource allocation. *INFORMS Journal on Computing*, 27(4):718–734, 2015. doi: 10.1287/ijoc.2015.0655. URL http://search.ebscohost.com/login.aspx?direct=true&AuthType=ip&db=bsu&AN=114699847&site=ehost-live.

David Gómez and Alfonso Rojas. An Empirical Overview of the No Free Lunch Theorem and Its Effect on Real-World Machine Learning Classification. *Neural Computation*, 28(1):216–228, January 2016. ISSN 0899-7667. doi: 10.1162/NECO_a_00793. URL https://doi.org/10.1162/NECO_a_00793.

Isabelle Guyon, André, and Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

M.A. Hall and G. Holmes. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data Engineering*, 15(6):1437–1447, November 2003. doi: 10.1109/TKDE.2003.1245283.

Mark A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, Hamilton, New Zealand, April 1999.

Anne-Claire Haury, Pierre Gestraud, and Jean-Philippe Vert. The Influence of Feature Selection Methods on Accuracy, Stability and Interpretability of Molecular Signatures. *PLOS ONE*, 6(12):e28210, December 2011. doi: 10.1371/journal.pone.0028210. URL http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0028210.

Robert C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–90, 1993. URL http://link.springer.com/article/10.1023/A:1022631118932.

Konstantin Hopf. *Predictive Analytics for Energy Efficiency and Energy Retailing*, volume 36 of *Contributions of the Faculty Information Systems and Applied Computer Sciences of the Otto-Friedrich-*

University Bamberg. University of Bamberg, 1 edition, 2019. ISBN 978-3-86309-669-4. URL https://doi.org/10.20378/irbo-54833.

Jianping Hua, Waibhav D. Tembe, and Edward R. Dougherty. Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recognition*, 42(3):409–424, March 2009. doi: 10.1016/j.patcog.2008.08.001. URL http://linkinghub.elsevier.com/retrieve/pii/S0031320308003142.

Earl B Hunt, Janet Marin, and Philip J Stone. *Experiments in induction.* Academic Press, Oxford, 1966.

Iñaki Inza, Pedro Larrañaga, Rosa Blanco, and Antonio J. Cerrolaza. Filter versus wrapper gene selection approaches in DNA microarray domains. *Artificial Intelligence in Medicine*, 31(2):91–103, June 2004. doi: 10.1016/j.artmed.2004.01.007. URL http://www.sciencedirect.com/science/article/pii/S0933365704000193.

A. Jain and D. Zongker. Feature selection: evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997.

Alexandros Kalousis, Julien Prados, and Melanie Hilario. Stability of Feature Selection Algorithms: A Study on High-dimensional Spaces. *Knowledge and Information Systems*, 12(1):95–116, May 2007. doi: 10.1007/s10115-006-0040-8.

Harsurinder Kaur, Husanbir Singh Pannu, and Avleen Kaur Malhi. A systematic review on imbalanced data challenges in machine learning: Applications and solutions. *ACM Computing Surveys*, 52(4):79:1–79:36, 2019. doi: 10.1145/3343440. URL https://doi.org/10.1145/3343440.

Eamonn Keogh and Abdullah Mueen. Curse of Dimensionality. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 257–258. Springer, Boston, MA, 2011. ISBN 978-0-387-30768-8. doi: 10.1007/978-0-387-30164-8_192. URL http://link.springer.com/referenceworkentry/10.1007/978-0-387-30164-8_192.

Taghi M. Khoshgoftaar, Kehan Gao, Ye Chen, and Amri Napolitano. Comparing Feature Selection Techniques for Software Quality Estimation Using Data-Sampling-Based Boosting Algorithms. *International Journal of Reliability, Quality and Safety Engineering*, 22(03):1550013, May 2015. doi: 10.1142/S0218539315500138.

Kenji Kira and Larry A. Rendell. A practical approach to feature selection. In *Proceedings of the ninth international workshop on Machine learning*, pages 249–256. Morgan Kaufmann, Aberdeen, Scotland, United Kingdom, 1992.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *The proceedings of the 14th International Conference in AI*, volume 14, pages 1137–1145. Morgan Kaufmann, 1995.

Igor Kononenko. Estimating attributes: Analysis and extensions of RELIEF. In *Machine Learning: ECML-94*, pages 171–182, Berlin, Heidelberg, April 1994. Springer. doi: 10.1007/3-540-57868-4_57. URL https://link.springer.com/chapter/10.1007/3-540-57868-4_57.

Gang Kou, Pei Yang, Yi Peng, Feng Xiao, Yang Chen, and Fawaz E. Alsaadi. Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. *Applied Soft Computing*, 86:105836, 2020. ISSN 1568-4946. doi: 10.1016/j.asoc.2019.105836.

Mathias Kraus, Stefan Feuerriegel, and Asil Oztekin. Deep learning in business analytics and operations research: Models, applications and managerial implications. *European Journal of Operational Research*, 281(3):628–641, 2020. doi: 10.1016/j.ejor.2019.09.018. URL http://www.sciencedirect.com/science/article/pii/S0377221719307581.

Mineichi Kudo and Jack Sklansky. Comparison of Algorithms that Select Features for Pattern Classifiers. *Pattern Recognition*, 33(1):25–41, 2000.

M. Kukar, I. Kononenko, C. Groselj, K. Kralj, and J. Fettich. Analysing and improving the diagnosis of ischaemic heart disease with machine learning. *Artificial Intelligence in Medicine*, 16(1):25–50, May 1999.

Miron B. Kursa. *praznik: Tools for Information-Based Feature Selection*, 2020. URL https://CRAN.R-project.org/package=praznik. R package version 8.0.0.

Carmen Lai, Marcel JT Reinders, Laura J van't Veer, and Lodewyk FA Wessels. A comparison of univariate and multivariate gene selection techniques for classification of cancer datasets. *BMC Bioinformatics*, 7(1):235, 2006. doi: 10.1186/1471-2105-7-235. URL http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-7-235.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. doi: 10.1038/nature14539. URL https://www.nature.com/articles/nature14539.

In Lee and Yong Jae Shin. Machine learning for enterprises: Applications, algorithm selection, and challenges. *Business Horizons*, 63(2):157–170, 2020. doi: 10.1016/j.bushor.2019.10.005. URL http://www.sciencedirect.com/science/article/pii/S0007681319301521.

Alexandra L'Heureux, Katarina Grolinger, Hany F. Elyamany, and Miriam A. M. Capretz. Machine Learning With Big Data: Challenges and Approaches. *IEEE Access*, 5:7776–7797, 2017. doi: 10.1109/ACCESS.2017.2696365. URL http://ieeexplore.ieee.org/document/7906512/.

Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature Selection: A Data Perspective. *ACM Computing Surveys*, 50(6):94:1–94:45, December 2017. ISSN 0360-0300. doi: 10.1145/3136625.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002. URL https://CRAN.R-project.org/doc/Rnews/.

Huan Liu and Rudy Setiono. Chi2: Feature selection and discretization of numeric attributes. In *TAI '95 Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*, page 88. IEEE, 1995. ISBN 0-8186-7312-5.

Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao. Feature Selection: An Ever Evolving Frontier in Data Mining. In *Proceedings of the Fourth International Workshop on Feature Selection in Data Mining, Journal of Machine Learning Research*, volume 10, pages 4–13, Hyderabad, India, 2010. URL http://www.jmlr.org/proceedings/papers/v10/liu10b/liu10b.

Huiqing Liu, Jinyan Li, and Limsoon Wong. A Comparative Study on Feature Selection and Classification Methods Using Gene Expression Profiles and Proteomic Patterns. *Genome Informatics*, 13:51–60, 2002. doi: 10.11234/gi1990.13.51.

Meng Liu, Chang Xu, Yong Luo, Chao Xu, Yonggang Wen, and Dacheng Tao. Cost-Sensitive Feature Selection by Optimizing F-Measures. *IEEE Transactions on Image Processing*, 27(3):1323–1335, March 2018. doi: 10.1109/TIP.2017.2781298. URL https://ieeexplore.ieee.org/document/8170306/.

Gary Marcus. Deep learning: A critical appraisal, 2018.

David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2019. URL https://CRAN.R-project.org/package=e1071. R package version 1.7-3.

Patrick E. Meyer and Gianluca Bontempi. On the use of variable complementarity for feature selection in cancer classification. In Franz Rothlauf, Jürgen Branke, Stefano Cagnoni, Ernesto Costa, Carlos Cotta, Rolf Drechsler, Evelyne Lutton, Penousal Machado, Jason H. Moore, Juan Romero, George D. Smith, Giovanni Squillero, and Hideyuki Takagi, editors, *Applications of Evolutionary Computing*, Lecture Notes in Computer Science, pages 91–102. Springer, 2006. ISBN 978-3-540-33238-1. doi: 10.1007/11732242_9.

Patrick Emmanuel Meyer, Colas Schretter, and Gianluca Bontempi. Information-Theoretic Feature Selection in Microarray Data Using Variable Complementarity. *IEEE Journal of Selected Topics in Signal Processing*, 2(3):261–274, June 2008. doi: 10.1109/JSTSP.2008.923858. URL http://ieeexplore.ieee.org/document/4550559/.

Oliver Müller, Maria Fay, and Jan vom Brocke. The effect of big data and analytics on firm performance: An econometric analysis considering industry characteristics. *Journal of Management Information Systems*, 35(2):488–509, 2018. doi: 10.1080/07421222.2018.1451955.

Sarah Nogueira, Konstantinos Sechidis, and Gavin Brown. On the stability of feature selection algorithms. *Journal of Machine Learning Research*, 18:1–54, 2018.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and others. Scikit-learn. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

Hanchuan Peng, Fuhui Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, August 2005. doi: 10.1109/TPAMI.2005.159.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. doi: 10.1007/BF00116251. URL https://link.springer.com/article/10.1007/BF00116251.

John O. Rawlings, Sastry G. Pantula, and David A. Dickey. *Applied regression analysis: a research tool*. Springer texts in statistics. Springer, New York, 2nd ed edition, 1998. ISBN 978-0-387-98454-4.

Juha Reunanen. Overfitting in Making Comparisons Between Variable Selection Methods. *Journal of Machine Learning Research*, 3:1371–1382, March 2003. URL http://dl.acm.org/citation.cfm?id=944919.944978.

Marko Robnik-Šikonja. Experiments with Cost-Sensitive Feature Evaluation. In *Machine Learning: ECML 2003*, pages 325–336, Berlin, Heidelberg, September 2003. Springer. doi: 10.1007/978-3-540-39857-8_30. URL https://link.springer.com/chapter/10.1007/978-3-540-39857-8_30.

Marko Robnik-Šikonja and Igor Kononenko. Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, 53:23–69, 2003. doi: 10.1023/A:1025667309714.

Marko Robnik-Sikonja and Petr Savicky. *CORElearn: Classification, Regression and Feature Evaluation*, 2020. URL https://CRAN.R-project.org/package=CORElearn. R package version 1.54.2.

Piotr Romanski and Lars Kotthoff. *FSelector: Selecting Attributes*, 2018. URL https://CRAN.R-project.org/package=FSelector. R package version 0.31.

Asim Roy, Shiban Qureshi, Kartikeya Pande, Divitha Nair, Kartik Gairola, Pooja Jain, Suraj Singh, Kirti Sharma, Akshay Jagadale, Yi-Yang Lin, Shashank Sharma, Ramya Gotety, Yuexin Zhang,

Ji Tang, Tejas Mehta, Hemanth Sindhanuru, Nonso Okafor, Santak Das, Chidambara N. Gopal, Srinivasa B. Rudraraju, and Avinash V. Kakarlapudi. Performance comparison of machine learning platforms. *INFORMS Journal on Computing*, 31(2):207–225, 2019. doi: 10.1287/ijoc.2018.0825. URL https://pubsonline.informs.org/doi/10.1287/ijoc.2018.0825.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson education, 2016.

Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, January 2007. doi: 10.1093/bioinformatics/btm344. URL http://bioinformatics.oxfordjournals.org/content/23/19/2507.

Yvan Saeys, Thomas Abeel, and Yves van de Peer. Robust Feature Selection Using Ensemble Feature Selection Techniques. In *Machine Learning and Knowledge Discovery in Databases*, pages 313–325, Berlin, Heidelberg, September 2008. Springer. doi: 10.1007/978-3-540-87481-2_21. URL https://link.springer.com/chapter/10.1007/978-3-540-87481-2_21.

José A Sáez, Mikel Galar, JuliáN Luengo, and Francisco Herrera. Tackling the problem of classification with noisy data using multiple classifier systems: Analysis of the performance and robustness. *Information Sciences*, 247:1–20, 2013.

Vijay Bhaskar Semwal, Kaushik Mondal, and G. C. Nandi. Robust and accurate feature selection for humanoid push recovery and classification: deep learning approach. *Neural Computing and Applications*, 28(3):565–574, March 2017. ISSN 0941-0643, 1433-3058. doi: 10.1007/s00521-015-2089-3.

Galit Shmueli and Otto R. Koppius. Predictive analytics in information systems research. *MIS Quarterly*, 35(3):553–572, 2011.

Noelia Sánchez-Maroño, Amparo Alonso-Betanzos, and María Tombilla-Sanromán. Filter Methods for Feature Selection – A Comparative Study. In Hujun Yin, Peter Tino, Emilio Corchado, Will Byrne, and Xin Yao, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2007*, volume 4881, pages 178–187. Springer, Berlin, Heidelberg, 2007. ISBN 978-3-540-77225-5. doi: 10.1007/978-3-540-77226-2_19. URL http://link.springer.com/10.1007/978-3-540-77226-2_19.

Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 1. Wiley, New York, 1998.

Yap Bee Wah, Nurain Ibrahim, Hamzah Abdul Hamid, Shuzlina Abdul-Rahman, and Simon Fong. Feature Selection Methods: Case of Filter and Wrapper Approaches for Maximising Classification Accuracy. *Pertanika Journal of Science & Technology*, 26(1), 2018.

H. White. A Heteroskedasticity-Consistent Covariance Matrix and a Direct Test for Heteroskedasticity. *Econometrica*, (48):817–383, 1980.

David H. Wolpert. The Lack of A Priori Distinctions Between Learning Algorithms. *Neural Computation*, 8(7):1341–1390, October 1996. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.1996.8.7.1341. URL https://www.mitpressjournals.org/doi/abs/10.1162/neco.1996.8.7.1341.

Lynn Wu, Lorin Hitt, and Bowen Lou. Data analytics, innovation, and firm productivity. *Management Science*, 66(5):2017–2039, 2019. doi: 10.1287/mnsc.2018.3281.

Bing Xue, Mengjie Zhang, and Will N. Browne. A Comprehensive Comparison on Evolutionary Feature Selection Approaches to Classification. *International Journal of Computational Intelligence and Applications*, 14(02):1550008, June 2015. doi: 10.1142/S146902681550008X. URL https://www.worldscientific.com/doi/abs/10.1142/S146902681550008X.

Howard Hua Yang and John Moody. Data visualization and feature selection: New algorithms for nongaussian data. In *Advances in neural information processing systems*, pages 687–693, 2000. URL http://papers.nips.cc/paper/1779-data-visualization-and-feature-selection-new-algorithms-for-nongau

Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, volume 3, pages 856–863, 2003.

Achim Zeileis. Econometric Computing with HC and HAC Covariance Matrix Estimators. *Journal of Statistical Software*, 11(1):1–17, November 2004. doi: 10.18637/jss.v011.i10. URL https://www.jstatsoft.org/index.php/jss/article/view/v011i10. Number: 1.

Xingquan Zhu and Xindong Wu. Class noise vs. attribute noise: A quantitative study. *Artificial Intelligence Review*, 22(3):177–210, 2004. doi: 10.1007/s10462-004-0751-8. URL http://link.springer.com/article/10.1007/s10462-004-0751-8.