Don't Rule Out Simple Models Prematurely: a Large Scale Benchmark Comparing Linear and Non-linear Classifiers in OpenML

Benjamin Strang[1], Peter van der Putten[2],
Jan N. van Rijn[1,3], and Frank Hutter[1]

[1] University of Freiburg, Germany
{benjamin.strang@students,vanrijn@cs,fh@cs}.uni-freiburg.de
[2] Leiden University, The Netherlands
p.w.h.van.der.putten@liacs.leidenuniv.nl
[3] Columbia University, USA
j.n.vanrijn@columbia.edu

Abstract. A basic step for each data-mining or machine learning task is to determine which model to choose based on the problem and the data at hand. In this paper we investigate when non-linear classifiers outperform linear classifiers by means of a large scale experiment. We benchmark linear and non-linear versions of three types of classifiers (support vector machines; neural networks; and decision trees), and analyze the results to determine on what type of datasets the non-linear version performs better. To the best of our knowledge, this work is the first principled and large scale attempt to support the common assumption that non-linear classifiers excel only when large amounts of data are available.

Keywords: Linear Classifiers, Meta-Learning, Benchmarking

1 Introduction

The experiments in many academic machine learning papers are designed to answer *which* particular method works better, typically by introducing a new algorithm and demonstrating success over a set of baselines or benchmarks. In a recent paper, Sculley et al. (2018) pinpoint this as a problem: 'Empirical studies have become challenges to be won, rather than a process for developing insight and understanding.' [22] To counteract this, we propose to answer the question *when* certain methods work better. Furthermore, we propose to add referenceable large scale empirical support for rules of thumb that are frequently used by data miners in real world applications. Meta learning studies can achieve these goals and thereby help turn machine learning from what has recently been called alchemy [16] into more of a principled engineering science. In this paper we will investigate when non-linear models outperform linear models. This may appear as a somewhat strange research question in this day and age, but linear models are still frequently used in practice since they are simpler, typically computationally more efficient and (due to their simplicity) often easier to interpret than modern non-linear models, such as deep learning models. With EU regulations on algorithmic decision-making and a "right to an explanation" [9] which came into effect on May 25, 2018, especially this often belittled dimension of interpretability is bound to become one of the most important deciding factors in day-to-day machine learning business. Furthermore, as our experiments demonstrate, it is not a given that a non-linear classifier will outperform a linear one at a statistically significant level. The underlying problem may simply be linear, or more commonly, insufficient data is available to estimate complex relationships reliably; furthermore, non-linear methods run a larger risk of overfitting given that they are typically higher variance methods [15].

Our contributions are as follows: (i) We run a large scale meta learning experiment on 299 datasets from OpenML [18,24], and compare linear vs. non-linear variants of neural networks, support vector machines and decision trees. Based on these datasets we give an indication when non-linear models may work better, and how often. (ii) We train a meta model to predict when non-linear models work better based on dataset characteristics. (iii) All experimental data and results are made available through OpenML, and the code used is made available as a Jupyter notebook. (iv) Whilst we address a very common topic in modeling, to the best of our knowledge this study is at least an order of magnitude larger than other studies on this topic in terms of number of datasets included.

2 Related Work

We review the literature on some exemplar studies that either discuss the question whether to use a linear or non-linear classifier, or use large scale experimentation to answer general scientific questions. Due to the broadness of these subject areas, this list is by no means complete.

Linear vs. Non-linear classifiers Various studies exist that compare linear classifiers and non-linear classifiers. Typically the comparison of linear and non-linear classifiers is performed in the context of a special modeling task, e.g., electricity consumption forecasting [10], CO_2 emissions [13], aggregate retail sales [4], EEG signal classification [7], corporate distress diagnosis [1], macroeconomic time series forecasting [23], routing [21] and epidemiological data [8]. These studies have in common that they are small scale experiments limiting the performance comparison to a special field of application and a small number of datasets. A simple general conclusion regarding the classification performance of linear and non-linear models cannot be drawn from the aforementioned related work as the final conclusions of these studies differ regarding classification performance. While in some studies [4,7,10,13] non-linear models in the form of neural networks or support vector machines achieved a better performance, some research groups find that non-linear components or methods are of no benefit or worse than the particular linear modeling approach [8,21].

Large Scale Experimentation OpenML [18,24] offers infrastructure to conduct large scale experiments which provide a solid empirical foundation for answering scientific questions. For each dataset, it contains a range of scientific tasks and meta-features, and it also allows for uploading new experimental results. In the past, several large scale experiments have been conducted by various research groups using this infrastructure. Flach and Kull (2015) use the experimental results on OpenML to study characteristics of precision recall curves over 886 classification datasets [6]. Post et al. (2016) researched in which cases feature selection improves classification performance on 399 classification datasets [14]. Olier et al. (2018) developed an algorithm selection method for QSAR's, and demonstrated its applicability on 2,700 QSAR problems [11]. Indeed, empirical results are typically more credible when based on a large number of datasets.

3 Background

This work aims to answer the basic scientific question when to use a linear or non-linear classifier by large scale experimentation. To achieve this, care needs to be taken to build on a solid infrastructure and experimental setup. In this section, we review the methods we used.

Datasets We prefer quality over quantity. As such, rather than using all of the thousands of datasets on OpenML, we selected a (still large) set of diverse datasets, i.e., the OpenML100 [3], which provides 100 datasets carefully selected from the OpenML overall dataset repository. The OpenML100 is designed to contain datasets that have a real world concept (rather than artificially generated data), have a meaningful classification task, and are introduced by a scientific publication. While the OpenML100 imposes dimensionality restrictions on the datasets (i.e., 500–100,000 data points, 1–5,000 features), we also report on results on datasets outside this range. Like for the OpenML100, highly unbalanced datasets with a minority class to majority class ratio of less than 0.05 were excluded. As this additional set of datasets is not as curated as the OpenML100, these results are reported separately. The total number of datasets used is 299.

Classifiers This study considers support vector machines (SVMs), neural networks and decision trees, each of them in a linear and a non-linear variant. SVMs natively support the notion of (non-)linearity by means of their kernel;

we use either a linear or an RBF kernel. For neural networks, next to a standard feed-forward network with hidden layers and non-linear (sigmoid) activation functions, as a linear variant we consider a linear model with no hidden layers trained by stochastic gradient descent. For decision trees, as a linear version we consider a decision stump (a decision tree with depth 1); this is arguably a very limited linear model, as it can only represent decision boundaries perpendicular to one of the axes. This means that even when a dataset is linearly separable, a decision stump might not be able to model it perfectly; however, we decided to still include this as a representative of the popular class of tree-based models.[1]

[1] In this study, we do not compare (still quite interpretable) decision trees against (more powerful, yet less interpretable) random forests in order to limit ourselves purely to a comparison of linear vs. non-linear models.

Table 1. Hyperparameters optimized by random search.

Classifier	parameters
SVM (linear)	C ($2^{-5} \dots 2^{15}$, log-scale), dual (boolean), imputation strategy, tol ($10^{-5} \dots 10^{-1}$, log-scale)
SVM (non-linear)	C ($2^{-5} \dots 2^{15}$, log-scale), gamma ($2^{-15} \dots 2^{3}$, log-scale), imputation strategy, tol ($10^{-5} \dots 10^{-1}$, log-scale), shrinking (boolean)
NN (linear)	alpha ($10^{-7} \dots 10^{-1}$, log-scale), imputation strategy, learning rate ('optimal', eta $= 1/(\alpha \cdot (t + t_0))$), tol ($10^{-5} \dots 10^{-1}$, log-scale), penalty (l2, l1, elasticnet)
NN (non-linear)	alpha ($10^{-7} \dots 10^{-1}$), early stopping (boolean), hidden layer size (32, 64, 128), imputation strategy, initial learning rate ($10^{-5} \dots 10^{0}$, log-scale), num. hidden layers (1, 2), tol ($10^{-5} \dots 10^{-1}$)
DT (stump)	criterion (gini, entropy), imputation strategy, max. features ($0.1, 0.2, 0.3, \dots 1.0$)
DT (non-linear)	criterion (gini, entropy), imputation strategy, max. depth ($2, 3, 5, 7, 10$), max. features ($0.1, 0.2, 0.3, \dots 1.0$)

Hyperparameter Optimization The performance of machine learning classifiers highly depends on hyperparameter optimization, which can often make the difference between mediocre and state-of-the-art performance. In this study, to minimize the bias resulting from the choice of a particular hyperparameter optimization method, we use the simplest option: random search [2].

We use a budget of 250 iterations (in the case of the decision stump only 60 iterations due to the limited hyperparameter space). While more powerful optimization methods, such as Bayesian optimization, are sometimes orders of magnitudes faster, random search is simpler, trivially available in any programming language, almost parameter-free, and robustly applicable across various types of hyperparameter spaces, making it a straight-forward simple choice when we can afford to evaluate a large number of configurations.

In order to determine which hyperparameters are important to optimize, we followed the recommendations of [19]. Table 1 shows the hyperparameters and ranges over which we performed random search. We note in particular that this search space includes regularization hyperparameters, such as the penalty parameter C in SVMs and the strength α of the L_2 regularizer in neural networks.

Evaluation We use nested 10-fold cross-validation for evaluating the classifiers. For each of the 10 outer cross-validation folds, the hyperparameter optimization used an internal 3-fold cross-validation procedure on the training portion to determine the best hyperparameters. The model is then re-fitted using the best found hyperparameters on the full training set of that cross-validation fold, and this is used to make predictions for the test set.

4 Linear versus Non-linear

This section aims to answer the primary question of this work, i.e., when to use linear and non-linear classifiers.

Setup For each of the three classifier families, we perform hyperparameter optimization and measure the performance in terms of predictive accuracy of both the linear and non-linear classifier using 10-fold nested cross-validation. As this yields 10 individual scores, we can also perform a statistical test, which determines whether the results are statistically significant ($\alpha = 0.05$); for this, we used the Wilcoxon signed rank test as recommended by Demšar [5], since non-parametric tests do not depend on the assumption of normally-distributed data. The data is pre-processed using imputation, one-hot-encoding, variance threshold and feature scaling to unit variance. Of course, the values for these operations are inferred on the training set and applied to the test set. All classifiers, as well as the random search module, are as implemented in Scikit-learn version 0.19.1 [12]. Each algorithm had a maximum run time of 96 hours on a 20 core Intel Xeon E5-2630v4, i.e., a maximum of 1,920 CPU hours per run. Tasks which ran out of time were not evaluated. Figure 1 shows boxplots of the ratio of the run time for the linear and non-linear algorithms per dataset. The run time was measured over the full random search procedure.

Results The results are provided in an OpenML study[2], to which a Jupyter Notebook is attached. This section summarizes the results as three case studies. For each family of classifiers we present: (i) A table with summarizing statistics, both on the OpenML100 and on the complete set. (ii) A scatter plot showing for each dataset whether the linear or non-linear variant performed better, with the number of data points and the number of features as axes. (iii) A figure plotting the difference of mean predictive accuracy per dataset. It sorts the datasets by difference in mean accuracy scores. A positive difference indicates a better performance of the linear model.

The results of the SVM case study are presented in Table 2, Figure 2 and Figure 3; the results of the neural network case study are presented in Table 3, Figure 5 and Figure 4; finally, the results of the decision tree case study are presented in Table 4, Figure 6 and Figure 7.

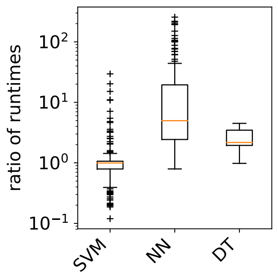

Fig. 1. Ratios of wall clock run times. Ratios larger than 1.0 indicate that the linear model needed less time.

Discussion Many of the results are as expected, which validates the experimental setup. The absolute statistics tables (Tables 2, 3 and 4) and difference plots (Figures 3, 4 and 7) show that the non-linear classifier performs better more frequently than the linear one. Especially for decision trees this difference is eminent, arguably because of the limited representation of the decision stump. However, the linear classifier also sometimes performs better, and in many cases there is no significant difference. Specifically, for all datasets, only in half the cases SVMs yielded statistically significantly better results with the non-linear kernel than with the linear one. However, we note that failure to detect a significant difference does not imply that there is no such difference: our statistical

[2] `https://www.openml.org/s/123`

Support Vector Machine Case Study

Table 2. General Statistics on Performance

	all datasets				OpenML100 datasets			
	absolutely		significantly		absolutely		significantly	
result	number	%	number	%	number	%	number	%
linear better	121	41	14	5	19	20	2	2
equal / neither ... nor	19	6	218	74	6	6	44	46
non-linear better	154	52	62	21	70	74	49	52
	294	100	294	100	95	100	95	100

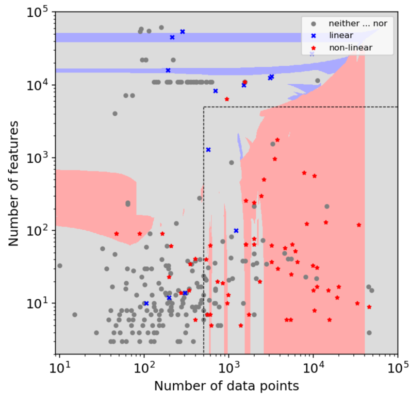

Fig. 2. Scatterplot showing whether linear or non-linear performs statistically better; each dot represents a dataset.

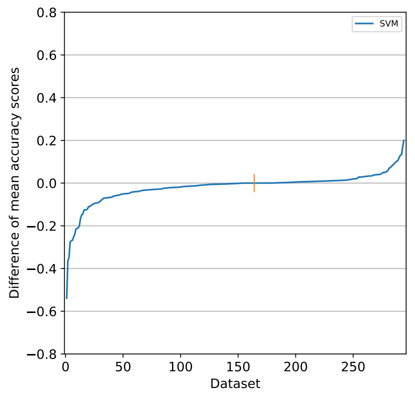

Fig. 3. Accuracy difference per dataset between linear and non-linear. Positive values indicate linear performed better.

tests are only based on 10 samples (one per cross-validation fold) and thus have limited power; thus, our results should not be overinterpreted.

The scatter plots (Figure 2, 5 and 6) reveal general trends which classifier performs better on datasets with specific characteristics. We plot this against the number of data points and the number of features; the background color shows which type of classifier is dominant in a region (based on a k-nearest-neighbour model with $k = 5$). Note that the background coloring looks a bit peculiar because it is determined in Euclidean space and represented in log space. Also note that some datasets have similar dimensions, causing several dots to overlap. For all classifier types the non-linear models are dominant in the regions with a large number of data points. However, when applied to a data set with few data points, the implementations of linear SVMs and neural networks we used do not perform worse than their non-linear counter parts at a statistically significant level. This result indicates that the optimal choice is not always clear-cut, and in case of doubt it may be preferable to use the linear version (since it is less likely to overfit, faster to run, and yields more interpretable results). We would like to highlight that, based on our data, we can only draw conclusions based

Neural Network Case Study

Table 3. General Statistics on Performance

	all datasets				OpenML100 datasets			
	absolutely		significantly		absolutely		significantly	
result	number	%	number	%	number	%	number	%
linear better	75	32	4	2	11	16	2	3
equal / neither ... nor	13	6	181	78	5	7	30	45
non-linear better	143	62	46	20	51	76	35	52
	231	100	231	100	67	100	67	100

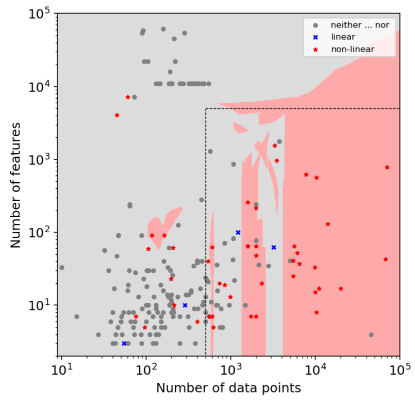

Fig. 4. Scatterplot showing whether linear or non-linear performs statistically better; each dot represents a dataset.

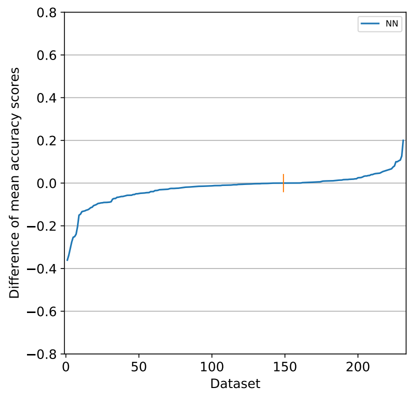

Fig. 5. Accuracy difference per dataset between linear and non-linear. Positive values indicate linear performed better.

on the scikit-learn implementations and the datasets we used; it remains an open question whether similar results would hold when using more advanced regularization schemes for the non-linear classifiers and other real-world data sets.

5 Learning when to use what classifier

In this section we present the results of an algorithm selection experiment. The relevance is three-fold: (i) this experiment adds credibility to the analysis in Section 4, by evaluating on a hidden test set (ii) this experiment is implicitly a deeper variant of the analysis in Section 4, i.e., by looking at a larger set of data characteristics, and (iii) the results of this experiment could be used to automatically select between a linear and non-linear classifier.

Setup The algorithm selection framework [17] consists of the following components: (i) a set of previously encountered datasets $\mathcal{D}$, (ii) a set of data characteristics $\mathcal{F}$ (also called meta-features), (iii) a set of algorithms $\mathcal{A}$, and (iv) measured

Decision Tree Case Study

Table 4. General Statistics on Performance

	all datasets				OpenML100 datasets			
	absolutely		significantly		absolutely		significantly	
result	number	%	number	%	number	%	number	%
linear better	79	26	4	1	10	10	0	0
equal / neither ... nor	13	4	162	54	0	0	17	17
non-linear better	207	69	133	44	90	90	83	83
	299	100	299	100	100	100	100	100

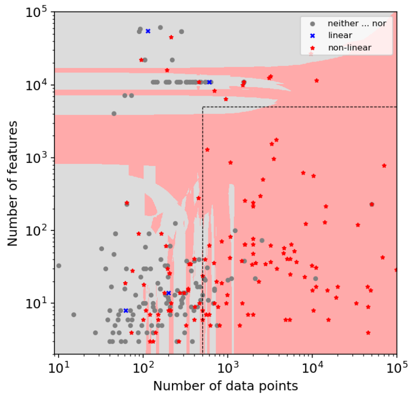

Fig. 6. Scatterplot showing whether linear or non-linear performs statistically better; each dot represents a dataset.

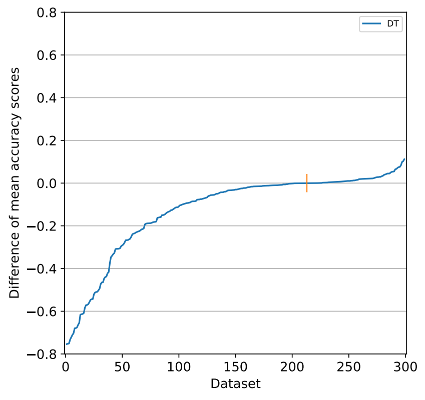

Fig. 7. Accuracy difference per dataset between linear and non-linear. Positive values indicate linear performed better.

performance p of the algorithms $\mathcal{A}$ on datasets $\mathcal{D}$. For any new dataset $\mathcal{D}'$ (not in $\mathcal{D}$) the task is to predict which algorithm from $\mathcal{A}$ maximizes performance measure p. In our case, $\mathcal{D}$ is the set of datasets on which both versions of a classifier terminated, $\mathcal{A}$ is the linear and the non-linear classifier of a given type, $\mathcal{F}$ is a set of meta-features selected from OpenML (which we define more precisely shortly) and performance measure p is predictive accuracy. We train a random forest (100 trees) on the set of meta-features to predict whether the optimized linear classifier or optimized non-linear classifier will perform statistically better. Cases where there is no statistical difference are assigned to the 'prefer linear' class. Hence, this is a binary decision problem.

Table 5 shows the sets of meta-features that we consider, per category. In our experiment we consider the following subset of these: (i) simple, containing just the features that can be computed in a single pass over the dataset, (ii) no landmarkers, which contains meta-features from the categories simple, statistical and information theoretic, and (iii) all, containing all meta-features in this table. Indeed, calculating meta-features comes at a certain cost and in particular calculating the landmarkers might impose a high run time. However, note that

Table 5. Meta Features

Category	Meta-features
Simple	Number Of Features, Number Of Data Points, Dimensionality, Default Accuracy, Number Of Data Points With Missing Values, Percentage Of Data Points With Missing Values, Number Of Missing Values, Percentage Of Missing Values, Number Of Numeric Features, Percentage Of Numeric Features, Number Of Symbolic Features, Percentage Of Symbolic Features, Number Of Binary Features, Percentage Of Binary Features, Majority Class Size, Majority Class Percentage, Minority Class Size, Minority Class Percentage, Number Of Classes, Minority Majority Ratio
Statistical	Mean Means Of Numeric Attributes, Mean Std Of Numeric Attributes, Mean Kurtosis Of Numeric Attributes, Mean Skewness Of Numeric Attributes
Information Theoretic	Class Entropy, Mean Attribute Entropy, Mean Mutual Information, Equivalent Number Of Attributes, Mean Noise To Signal Ratio
Landmarkers	Decision Stump Error Rate, Decision Stump Kappa, Decision Stump AUC, Naive Bayes Error Rate, Naive Bayes Kappa, Naive Bayes AUC, 1-NN ErrRate, 1-NN Kappa, 1-NN NAUC

Table 6. Accuracy and AUROC scores for different sets of meta-features. The majority class is the combined set of 'linear statistically better' and 'no statistical difference'. Results over all datasets.

classifier	default	simple		no landmarkers		all	
family	accuracy	accuracy	AUC	accuracy	AUC	accuracy	AUC
SVM	0.789	0.844	0.874	0.844	0.897	**0.861**	0.908
DT	0.555	0.839	0.895	0.826	0.902	**0.856**	0.913
ANN	0.801	0.857	0.846	**0.870**	0.827	0.861	0.852
Total dataset	0.708	0.803	0.837	0.801	0.836	**0.805**	0.841

Table 7. Accuracy and AUROC scores for different sets of meta-features. The majority class is 'non-linear statistically better'. Results over the OpenML 100.

classifier	default	simple		no landmarkers		all	
family	accuracy	accuracy	AUC	accuracy	AUC	accuracy	AUC
SVM	0.516	0.611	0.745	0.621	0.719	**0.674**	0.789
DT	0.830	0.840	0.869	0.840	0.795	**0.870**	0.883
ANN	0.522	0.657	0.716	0.642	0.677	**0.672**	0.715
Total dataset	0.637	0.740	0.798	0.756	0.796	**0.760**	0.804

the algorithms in $\mathcal{A}$ are both optimized using 250 iterations of random search, and the landmarkers are (by design) ran with a given set of hyperparameters. This justifies the use of applying landmarkers. We evaluate the meta-model in a leave-one-out fashion, training the model on all but one dataset and test it on this left-out dataset, in order to assess the performance of the meta-model.

Results Table 6 and Table 7 show the performance results of the meta-learning experiment evaluated for all completed datasets and for the completed datasets of the OpenML100 repository, respectively. As baseline the default accuracy (obtained by always predicting the majority class) is listed for each subset. The 'classifier family' column shows which classifier type was used, the 'default accuracy' column shows the default accuracy, and the other columns show the accuracy and AUROC score of the meta-model for each set of meta-features. The set of meta-features with the highest accuracy score is typeset in bold.

Both tables show a similar trend. In all cases, the meta-model performs consistently better than the default accuracy. From this we conclude that the meta-

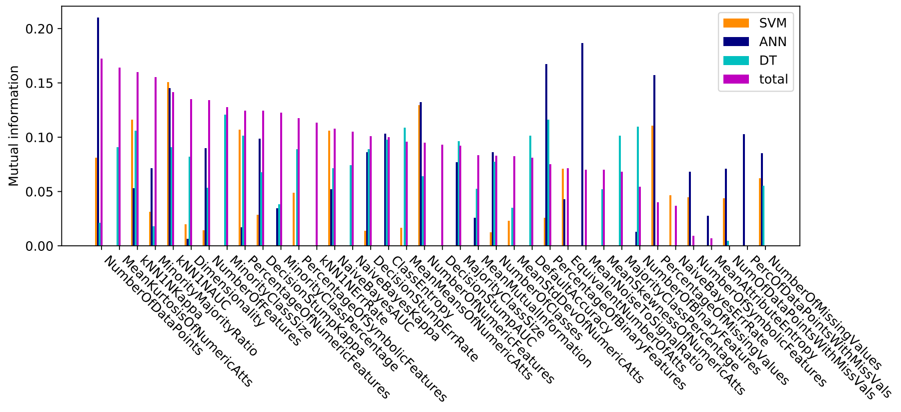

Fig. 8. Bar plot showing the mutual information for each meta feature. OpenML100 subset. Sorted by the mutual information values of the total dataset

features model something related to the linearity of the dataset. Interestingly, the majority class is different between the two meta-datasets. When considering only the OpenML100, the majority class is 'non-linear statistically better'; contrarily, when considering all datasets, the majority class is the combined set of 'linear statistically better' and 'no statistical difference'. This is most likely due to the slightly larger datasets in the OpenML100. The set of simple meta-features already makes a decent improvement compared to the default accuracy. Adding the set of statistical and information theoretic features adds only a little predictive power (as shown in the column 'no landmarkers'). Finally, adding the landmarker features (and thus having most information) results consistently in the highest accuracy. We analyze which features are most important for the meta-model. Features that are important to the meta-model have the potential to give more information about the linearity of a dataset and the dynamics between the linear and non-linear classifier. We use the mean mutual information measure, as described in [20], because this is a uni-variate measure and prevents biases incurred from correlations between features. The results are presented in Figure 8. The features that we analyzed in Section 4 (number of data points and number of features) appear to be quite important. Of the other features, the nearest neighbour based landmarkers seem important.

6 Conclusion

Motivated by the interpretability of linear models (which is important in the context of legal requirements explainability of automated decisions), as well as secondary niceties of linear models (such as ease of use and computational efficiency), this paper presented the results of a large scale experiment comparing the performance of linear and non-linear classifiers. Our main focus was to build large scale empirical support to determine the circumstances under which a given type of classifier is better. We considered three classifier families: SVMs, neural networks and decision trees, all as implemented in scikit-learn and represented by a corresponding linear and non-linear model. Unsurprisingly, non-linear models of each classifier family achieved a better performance on more datasets than their linear counterparts. However, for many datasets the performance difference was not significant, a finding that is highly relevant for practical applications. Meta-features related to dataset dimensionality (number of data points, number of features and the ratio of these) were the most relevant meta-features for deciding whether to choose a linear or a non-linear model. As expected, non-linear models typically exhibit a significantly better predictive performance if the dataset at hand has a large number of data points and few features.

In order to make this experiment reproducible, all results are available on OpenML and can be conveniently accessed through a Jupyter notebook. This also makes it convenient to change the experimental parameters, such as the set of datasets, displayed meta-features and optimization criterion, which all potentially influence the results. Future work will focus on a better understanding of the dynamics between meta-features and linearity of the dataset. One interesting direction would be to search for meta-features that better distinguish the datasets on which linear classifiers perform well. Furthermore, we would also like to perform these analysis on different evaluation measures, such as Area under the ROC Curve or F-measure. Having a publicly available meta-dataset enables the community to actively participate in this process.

In summary, as our title states: don't rule out simple models prematurely.

Acknowledgements This work has partly been supported by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme under grant no. 716721. The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant no INST 39/963-1 FUGG.

References

1. Altman, E.I., Marco, G., Varetto, F.: Corporate distress diagnosis: Comparisons using linear discriminant analysis and neural networks (the Italian experience). Journal of Banking & Finance 18(3), 505–529 (1994)
2. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of Machine Learning Research 13(Feb), 281–305 (2012)
3. Bischl, B., Casalicchio, G., Feurer, M., Hutter, F., Lang, M., Mantovani, R.G., van Rijn, J.N., Vanschoren, J.: OpenML Benchmarking Suites and the OpenML100. arXiv preprint arXiv:1708.03731 (2017)
4. Chu, C.W., Zhang, G.P.: A comparative study of linear and nonlinear models for aggregate retail sales forecasting. International Journal of Production Economics 86(3), 217–231 (2003)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7(Jan), 1–30 (2006)
6. Flach, P., Kull, M.: Precision-recall-gain curves: PR analysis done right. In: Advances in Neural Information Processing Systems. pp. 838–846 (2015)

7. Garrett, D., Peterson, D.A., Anderson, C.W., Thaut, M.H.: Comparison of linear, nonlinear, and feature selection methods for EEG signal classification. IEEE Trans. on Neural Systems and Rehabilitation Engineering 11(2), 141–144 (2003)
8. Gaudart, J., Giusiano, B., Huiart, L.: Comparison of the performance of multi-layer perceptron and linear regression for epidemiological data. Computational Statistics & Data Analysis 44(4), 547–570 (2004)
9. Goodman, B., Flaxman, S.: European Union regulations on algorithmic decision-making and a "right to explanation". ArXiv e-prints (Jun 2016)
10. Kaytez, F., Taplamacioglu, M.C., Cam, E., Hardalac, F.: Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares Support Vector Machines. International Journal of Electrical Power & Energy Systems 67, 431–438 (2015)
11. Olier, I., Sadawi, N., Bickerton, G.R., Vanschoren, J., Grosan, C., Soldatova, L., King, R.D.: Meta-QSAR: a large-scale application of meta-learning to drug design and discovery. Machine Learning pp. 1–27 (2018)
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
13. Pino-Mejías, R., Pérez-Fargallo, A., Rubio-Bellido, C., Pulido-Arcas, J.A.: Comparison of linear regression and artificial neural networks models to predict heating and cooling energy demand, energy consumption and CO_2 emissions. Energy 118, 24–36 (2017)
14. Post, M.J., van der Putten, P., van Rijn, J.N.: Does feature selection improve classification? A large scale experiment in OpenML. In: International Symposium on Intelligent Data Analysis. pp. 158–170. Springer (2016)
15. van der Putten, P., van Someren, M.: A bias-variance analysis of a real world learning problem: The CoIL Challenge 2000. Machine Learning 57(1), 177–195 (Oct 2004)
16. Rahimi, A., Recht, B.: Reflections on random kitchen sinks (2017)
17. Rice, J.R.: The Algorithm Selection Problem. Advances in Computers 15, 65118 (1976)
18. van Rijn, J.N.: Massively Collaborative Machine Learning. Ph.D. thesis, Leiden University (2016)
19. van Rijn, J.N., Hutter, F.: Hyperparameter importance across datasets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 2367–2376. ACM (2017)
20. Ross, B.C.: Mutual information between discrete and continuous data sets. PloS one 9(2), e87357 (2014)
21. Schütze, H., Hull, D.A., Pedersen, J.O.: A comparison of classifiers and document representations for the routing problem. In: Proceedings of the 18th annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 229–237. ACM (1995)
22. Sculley, D., Snoek, J., Wiltschko, A., Rahimi, A.: Winner's curse? On pace, progress, and empirical rigor. In: Proc. of ICLR 2018 (2018)
23. Swanson, N.R., White, H.: A model selection approach to real-time macroeconomic forecasting using linear models and artificial neural networks. The Review of Economics and Statistics 79(4), 540–550 (1997)
24. Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: networked science in machine learning. ACM SIGKDD Explorations Newsletter 15(2), 49–60 (2014)