LETTER **Communicated by David Wolpert**

An Empirical Overview of the No Free Lunch Theorem and Its Effect on Real-World Machine Learning Classification

David Gómez
david.gomez.guillen@entel.upc.edu
Alfonso Rojas
alfonso@entel.upc.edu
Telematics Engineering Department, Polytechnical University of Catalonia, Barcelona 08034, Spain

A sizable amount of research has been done to improve the mechanisms for knowledge extraction such as machine learning classification or regression. Quite unintuitively, the no free lunch (NFL) theorem states that all optimization problem strategies perform equally well when averaged over all possible problems. This fact seems to clash with the effort put forth toward better algorithms. This letter explores empirically the effect of the NFL theorem on some popular machine learning classification techniques over real-world data sets.

1 Introduction

In the relatively recent history of machine learning, one important goal has always been to provide the maximum predictive power on a particular data domain.[1] Since this domain is usually fixed in the form of a data set, the potential for prediction improvement lies in either data preprocessing or more sophisticated algorithms. The former is a customized procedure tailored to the specific needs of each data set, so to tackle general problems, a large amount of research has been done to obtain better algorithms.

Unfortunately, an objective evaluation of machine learning classifiers is not as straightforward as it could be. In 1996, Wolpert (1996a, 1996b) wrote about what came to be known as the no free lunch theorem, with another clarification oriented toward searching algorithms on (Wolpert, 2013). In a machine learning context, the NFL theorem implies that all learning algorithms perform equally well when averaged over all possible data sets. This nonintuitive idea meant that looking for a general, highly predictive algorithm is not feasible. It is also surprising, since it is well known in empirical research that some algorithms perform consistently much better than others.

[1] Other possible goals include simplicity, interpretability, or accountability.

doi:10.1162/NECO_a_00793

This letter's overview was compiled from results gathered on the development of a custom machine learning library for a software project of one of the authors (Gómez, 2014). Section II describes the relevant methodology used in the original study: the implemented algorithms, the data sets that were used, and other important details. From all the results, figures, and tables, section 3 focuses and comments on those relevant for our discussion on the NFL. Finally, we summarize our findings in section 4 and compare them to other similar research.

2 Methodology

During our machine learning library development, we implemented and tested eight well-known algorithms, not including other custom methods beyond the scope of this review. The tests were performed by averaging 20 executions of each individual algorithm on every one of the chosen data sets, using a holdout model validation scheme with two-thirds of the data for training and the remaining third for testing. The reported results include the average accuracy result and the 95% confidence interval to see its variance.

Besides the different classifiers, the original study also includes results about execution time, validation scheme behavior, and feature selection. The focus in this letter, though, is on the algorithm and data set performance, complemented with some results from other areas when necessary. (For more information, see Gómez, 2014).

Since the study was quite extensive, the training hyperparameters were fixed to the default values specified in Table 1. Time and hardware support were in limited supply, so a full-featured cross-validation hyperparameter tuning procedure was not possible to do for many experiments in the report. Nevertheless, the goal was not to obtain the maximum accuracy in each execution but to get an approximation in order to compare the different models, so the conclusions can still be valid if highly parametric classifiers such as neural networks are carefully considered.

The objective of this study is to use the results from that report to observe the effect of the assumptions and structural properties of both data sets and algorithms and discuss which underlying mechanisms exist to cause those effects.

For the purposes of testing the correctness and efficiency of the library, six data sets were used as a benchmark, chosen to represent a wide enough spectrum of complexity, size, and domain (medicine, finance, social science, visual recognition). All of them are available at the UCI machine learning repository (Lichman, 2013) and are described in Table 2.

The reason to develop a new machine learning library instead of reusing another one was twofold. On the one hand, existing Java libraries did not have the necessary technical or license requirements to use in our project, but most important, by writing the code from scratch, we were able to learn

Table 1: Classifiers Used and Default Parameters Used.

Classifier	Default Parameters
Naive Bayes (John & Langley, 1995)	
C4.5 decision trees (Quinlan, 1993)	Prepruning
Neural networks (Rumelhart, Hinton, & Williams, 1988)	One hidden layer (#attributes + #classes)/2 nodes Learning rate = 0.3 Momentum = 0.2 500 epochs
k-nearest neighbors (Altman, 1992)	$k = 3$ Euclidean distance
Logistic regression (Cox, 1958)	
Random C4.5 forest (Breiman, 2001)	20 trees
AdaBoost.M1 (Freund & Schapire, 1996)	Using decision stumps
Stacking (Wolpert, 1992)	using: Naive Bayes Random C4.5 forest Neural network 3-NN Combiner: logistic regression

and control their inner workings in more detail. Knowing more about these algorithms makes it possible to adjust, optimize, and use them in a more effective way for our specific purposes.

In order to evaluate these classifiers' implementation, the machine learning suite Weka (Hall et al., 2009) was used to compare the results and determine whether the accuracy of our models is similar. The assessment was positive, and it was concluded that all the algorithms work correctly, all of them having similar accuracy results to Weka's implementation with the small exception of decision trees. In this last case, the design choice to implement prepruning instead of postpruning makes our decision trees slightly less accurate—a few percentage points at most.

3 Results

The average accuracy over all six data sets is pictured in Figure 1. These results are mostly consistent with previous research, but according to the NFL theorem on a sufficiently large pool of data sets, they should be similarly accurate. The most sensible explanation is that the classification problems in the real world usually belong to a particular subgroup of these possible, theoretical data sets.

Some common structural properties or assumptions are general and applicable to all kinds of sets, like Occam's razor or the independent and

Table 2: Data Sets Used and Relevant Notes about Them.

Data Set	Notes
Audiology	Very imbalanced classes (see Figure 5)
	Many imbalanced features
	Large number of classes (24)
	(Relatively) large number of features (69)
	All features nominal
	Large amount of missing values
Column	Slightly imbalanced classes
	Low number of features (6)
	Low number of classes (3)
	All features nominal
	No missing values
Breast cancer	Slightly imbalanced classes
	Binary class
	All features nominal
	Some missing values
	Difficult to classify[a]
Multiple features (Fourier)	Balanced classes (see Figure 5)
	(Relatively) large set (2000 instances)
	All features numeric
	No missing values
	Significant difference in difficulty between classes
German credit	Slightly imbalanced classes
	Binary class
	Mix of nominal (13) and numeric (7) features
	No missing values
	Difficult to classify[b]
Nursery	Large set (12,960 instances)
	Mix of balanced (3) and imbalanced (2) classes
	No missing values
	All features nominal
	All possible instance permutations available (*census*)
	Easy to classify[c]

[a] Results from other research on the breast cancer set are available at http://www.fizyka.umk.pl/kis/projects/datasets.html#Ljubljana. We define difficulty as the inability to build a model that improves beyond the baseline accuracy by any significant amount, as best described by the κ statistic (Carletta, 1996).

[b] Results from other research on German credit set are available at Huang, Chen, and Wang (2006).

[c] Results from other research on nursery set are available at Kumar, Nitin, and Chauhan (2012).

identical distribution of the samples. From all the data-dependent structural properties in the data sets, the most critical assumptions we have found in our analysis are two: determinism and the Pareto principle.

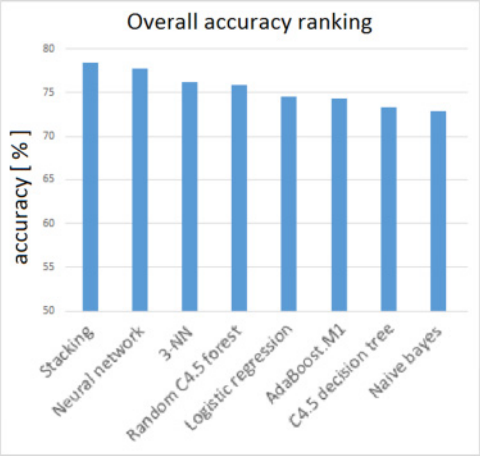

Figure 1: Average model accuracy over all six data sets.

3.1 Determinism. Determinism implies that the same input values should belong to the same class. Causality and determinism are the basis of science, even if noise is usually an unavoidable and unfortunate reality in practice. In this case, noise-resistant algorithms such as naive Bayes tend to work better than, for instance, decision trees that are prone to overfit the model unless properly pruned.

The comparison between the results of a fully deterministic data set (nursery, due to all possible instances being accounted for) and a noisy one (credit rating, probably due to hidden variables) can tell us more. Observing the decision tree's behavior, we see that it has below-baseline performance on the credit set but performs much better (compared to other algorithms) on the nursery set (center right and bottom right in figure 2, respectively). By performing feature selection in the former, the next section will show how the noise of useless attributes is filtered out and the result improves beyond the baseline accuracy, to a similar level to the rest of algorithms.

Many of the drawbacks posed by decision trees are reduced by using them in ensemble methods, such as bagging or random forests. These aggregations are able to cancel some of the noise and offer consistent improvement over individual trees, as pictured in Figure 3.

3.2 Pareto Principle. Most information in a data set is often contained in only a few features. Of course, features are initially chosen by the data collector, and many times they do not know which ones will be important; that is why they might use machine learning in the first place.

That means some attributes tend to be useless and add noise without providing much valuable information in return, disrupting the performance

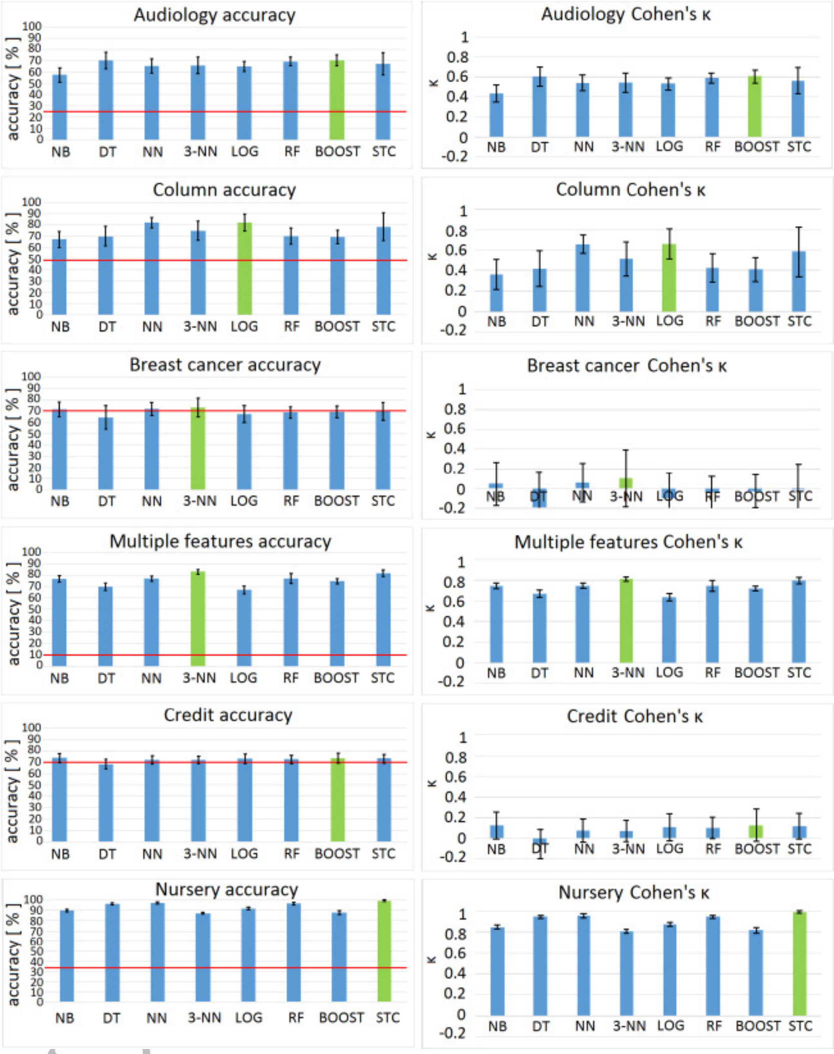

Figure 2: Average model accuracy for each tested data set (left) along with the average Cohen's kappa coefficient (right) to provide a more realistic outlook on the real helpfulness of the models.

of some classifiers, like decision trees, if they are not filtered out. Figure 4, for instance, shows how decision trees are specially sensitive to this problem and how the model average accuracy is considerably improved by filtering out the useless features.

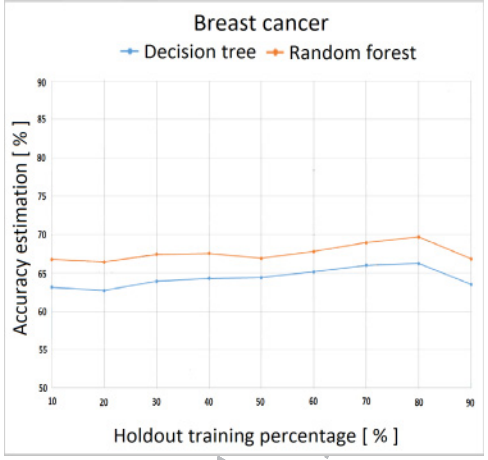

Figure 3: Average accuracy comparison between single decision trees and random forest, for different holdout percentages, for the breast cancer data set. In this case, forests reveal approximately a 4% improvement over single trees.

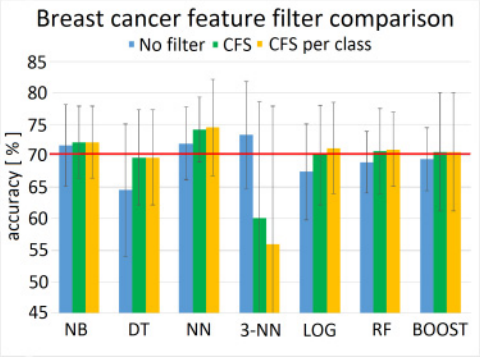

Figure 4: Breast cancer set accuracy per classifier, with different feature selection methods. Decision trees in particular perform notably better after CFS feature selection (Hall, 2000), while the kNN classifier, a local approximator, is heavily penalized.

Also related to this principle is the fact that many natural data sets have intrinsically imbalanced classes. We can see an example of that in the audiology data set class distribution in Figure 5 (left), where many classes are barely available. Even when balance exists, it is many times artificially implemented due to a lack of previous knowledge (see the multiple features set class balance on the right in Figure 5, where the same number of samples

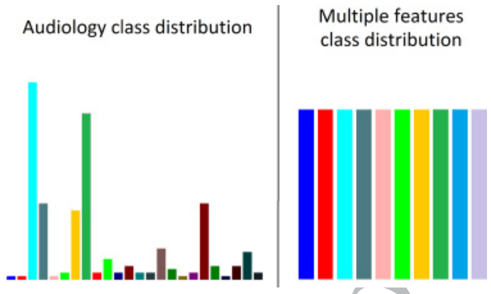

Figure 5: Audiology (left) and multiple features (right) data set class distributions. Highly imbalanced sets are usually more difficult to learn properly, and the overall accuracy estimation might not be representative of all classes.

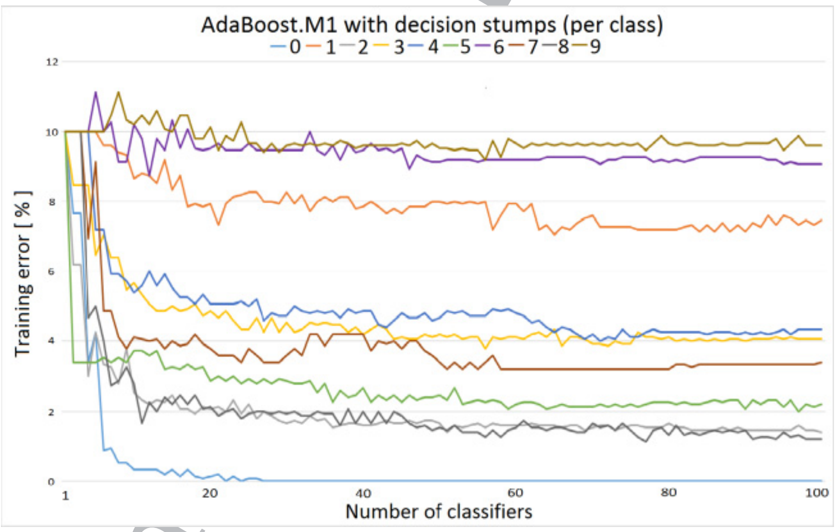

Figure 6: Training error for AdaBoost.M1 on the multiple features data set per digit. Excluding the first few points, the error rate is always below the baseline error (10%), yet when all these internal classifiers are combined, the average error is around 25%.

was chosen for each class). In these cases the most conservative action is assuming a noninformative prior (in Bayesian terms) and deciding that all classes are equally probable.

In the case of the multiple features set, AdaBoost.M1 exhibits clearly this effect (see Figure 6). This algorithm starts by training one-versus-all classifiers for each class (i.e., “Does this input represent digit *d*?”), building training sets with a 10%/90% class composition in this particular data set.

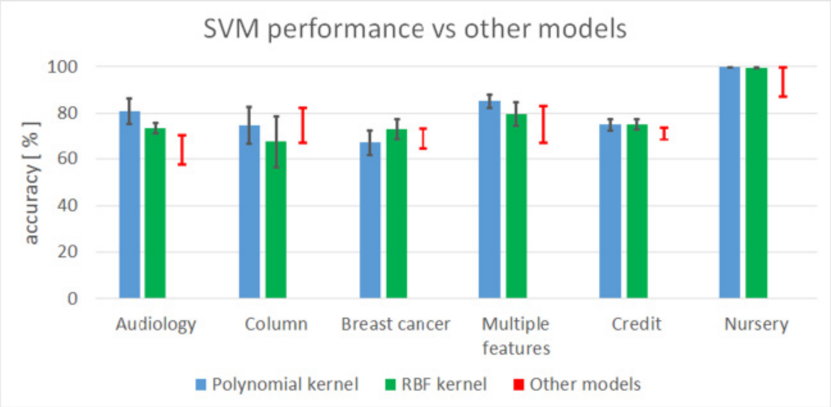

Figure 7: SVM models (with polynomial and RBF kernels) compared to the rest of the models introduced in Figure 2. The red interval represents the range (minimum and maximum) of the accuracies of the eight other models. In four out of the six sets, kernel machines surpass the maximum obtained by these previous classification algorithms.

The separate evaluation of these inner models can offer misleadingly high accuracies (over 90%), yet combining these classifiers returns an overall model with an accuracy around 75%.

This is expected. Since the goal is to minimize the error, there is always a biased prior toward the most populated class, so the inner models in an ensemble are not always representative of the original problem. This fact might determine which algorithm to use, since some of them (like kNN or SVM; Akbani, Kwek, & Japkowicz, 2004) are often significantly impaired by imbalanced classes.

Despite this last assertion, additional experiments we performed to test the capabilities of kernel machines (SVM in particular) show that even in imbalanced sets, these algorithms can outperform other popular methods. Figure 7 shows, for example, how an SVM on the audiology set can offer an increase of 10% over the best algorithm we tried in the original study.

3.3 Abstraction Layers. Beyond the traits we are able to discern in these admittedly simple (yet real) data sets, other aspects more relevant for current applications and research are worth discussing as well. Many state-of-the-art techniques in machine learning for complex data sets often focus on deep learning. The idea behind this philosophy is to break down complex problems in several abstraction layers to simplify the learning process; its adoption in recent years is widespread and well documented in a great variety of successful projects in companies such as Apple and Google (Vinyals, Toshev, Bengio, & Erhan, 2014).

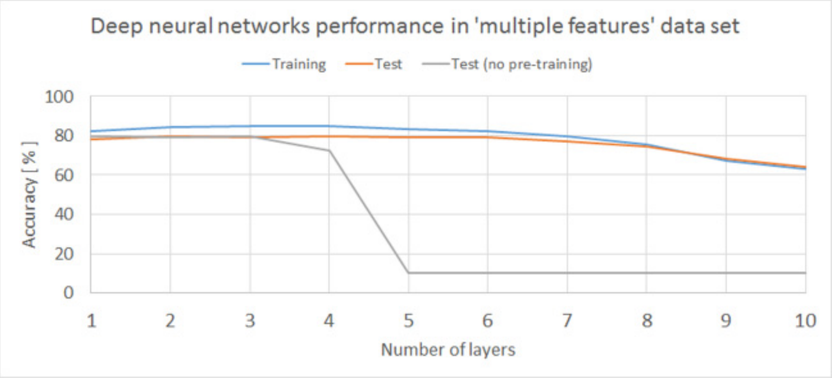

Figure 8: Results of deep neural networks on the multiple features data set. Without pretraining, after layer 5, the accuracy drops to the baseline of 10%, or what could be expected from a model choosing at random.

Despite the enormous success of these techniques, deep learning is also subject to the NFL limitations. It has been proven to work for sets with inherently hierarchical data from an abstraction point of view, like pictures in a facial detection system (pixels become edges, then features and finally faces), but without that kind of data structure, the added complexity may not be worth it.

In our experiments with these simple, small sets using deep neural networks, we found disappointing results: the models seem incapable of producing good results due to the limited number of examples and possibly the low data abstraction potential. As an example, in the largest of the sets tested (multiple features set with 2000 instances), we had the results shown in Figure 8. Among other observations, we see that:

- Regular backpropagation training on its own is ill suited for deep architectures, requiring some kind of pretraining to ease (Bengio, Lamblin, Popovici, & Larochelle, 2007).
- Accuracy slightly increases with a few layers but tends to eventually drop, as was previously known (Tesauro, 1992). Without pretraining, sufficiently deep networks become effectively random since the first layers are barely affected by backpropagation due to the vanishing gradient problem (Hochreiter, 1991).
- This last fact is unlikely to be due to overfitting, since the training and holdout accuracy are similar even on deep networks.
- Therefore, it seems reasonable to conclude that deep networks in this case are not capable of offering substantial improvements over shallow networks, even if more sophisticated training methods might improve the models a bit further.

As a corollary to deep learning exploitation of data structure, we also have to look at the kinds of data we are processing. Convolutional neural networks, for instance, are designed to take advantage of the specific structural properties that can be expected from images: locality, bidimensionality and, again, abstraction layers (Krizhevsky, Sutskever, & Hinton, 2012). Even with these traits, though, these networks can still be misled (Nguyen, Yosinski, & Clune, 2014), showing us that for their correct use, they need even further assumptions about the training images.

4 Conclusion

As usual in statistical modeling, we see that the most important point the NFL theorem warns us about is checking assumptions for our particular problem. Clearly, not all the algorithms will perform equally well on all problems: very specific, artificial data sets could be designed to be difficult to handle accurately for one particular classifier, as seen in Nguyen et al. (2014). But in the real world, we have seen (in this small sample of six and those in other research publications) that conclusions can be made that, while not completely general, are applicable to most data sets found in real practical problems.

While evaluating the average accuracy ranking for our six data sets, we noticed the effect of the NFL theorem and how assumptions are key to performance. On one hand there is Naive Bayes, a model that makes a strong assumption about feature independence that is rarely met, with the lowest score in the ranking. On the other hand, ensemble methods work really well by properly combining the strengths of different classifiers and compensating each other's weaknesses. In our experiments, the stacking classifier is rarely the most accurate on any set, but the results show that it has the best average performance.

One premise we were concerned about in section 2 is the lack of hyperparameter tuning in this study and how that could affect our conclusions. This concern was unwarranted. The most sensitive classifier to this problem, the neural network, has been ranked as the second-best model in our test. That shows that even when not properly optimized, neural networks are still very powerful classifiers.

By contrasting these results with previous research, further evidence of the NFL theorem is found. Fernández-Delgado, Cernadas, Barro, and Amorim (2014) reach similar results to those obtained in this letter, with one glaring exception: stacking algorithms. While in our experiments the stacking implementation had the best average accuracy, their study concluded that it was one of the worst globally. The difference can be explained by the base classifiers: they use zeroR in their implementation, while we included those described in Table 1. For further evidence of stacking as a highly competitive technique, most machine learning competitions end with some kind of stacking (also known as *blending*) as the best classifiers, seen, for example, in the 2008 Netflix competition (Bell, Koren, & Volinsky, 2008). We see, then, that learning ensembles are very sensitive to the models they contain, and their performance can wildly vary if not well designed. Whether it is adequate will depend, once again, on the problem we are considering: a consequence of the NFL theorem.

Besides this small discrepancy, the other results of the letter are mostly coherent with ours: random forests, neural networks, and k-nn score high on their ranking, followed by boosting and logistic regression and, later, by naive Bayes, among others.

This comparison shows that the data and its preprocessing are as important as, if not more so than, the algorithm itself in determining the quality of the model. Data visualization or statistical techniques such as feature selection can be crucial to provide a better fit and obtain simpler and better models. Even if ultimately the classifier to use might be decided by performing some kind of empirical model selection (like cross-validation), knowing the advantages, disadvantages, assumptions, and caveats of the different algorithms can guide and speed up the learning process.

References

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *Proceedings of the 15th European Conference on Machine Learning* (pp. 39–50). New York: Springer.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician*, *46*, 175–185.

Bell, R. M., Koren, Y., & Volinsky, C. (2008). The BellKor 2008 solution to the Netflix Prize. http://www2.research.att.com/~Volinsky/netflix/Bellkor2008.pdf

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks (pp. 153–160). In B. Schölkopf, J. C. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems, 19*. Cambridge, MA: MIT Press.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, *22*, 249–254.

Cox, D. R. (1958). The regression analysis of binary sequences (with discussion). *Journal of the Royal Statistical Society: Series B*, *20*, 215–242.

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, *15*, 3133–3181.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Gómez, D. (2014). *Krimtrack: An integrated coordination system with a virtual machine learning advisor*. http://upcommons.upc.edu/pfc/handle/2099.1/24173?locale=en.

Hall, M. A. (2000). Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 359–366). San Mateo, CA: Morgan kaufmann.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009). The WEKA data mining software: An update. *SIGKDD Explorations Newsletter*, *11*(1), 10–18.

Hochreiter, S. (1991). *Untersuchungen zu dynamischen neuronalen Netzen.* Diploma thesis, Technische Universität München.

Huang, C.-L., Chen, M.-C., & Wang, C.-J. (2006). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications*, *33*(4).

John, G., & Langley, P. (1995). Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (pp. 338–345). San Mateo, CA: Morgan kaufmann.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In F. Pereira, C.J.C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems, 25* (pp. 1097–1105). Red Hook, NY: Curran.

Kumar, P., Nitin, V.K.S., & Chauhan, D. S. (2012). *A benchmark to select data mining based classification algorithms for business intelligence and decision support systems*. Computing Research Repository. abs/1210.3139.

Lichman, M. (2013). *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml.

Nguyen, A. M., Yosinski, J., & Clune, J. (2014). *Deep neural networks are easily fooled: High confidence predictions for unrecognizable images*. Computing Research Repository. abs/1412.1897.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Mateo, CA: Morgan kaufmann.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1988). Learning internal representations by error propagation. In J. A. Anderson & E. Rosenfeld (Eds.), *Neurocomputing: Foundations of research* (pp. 673–695). Cambridge, MA: MIT Press.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, *8*, 257–277.

Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2014). *Show and tell: A neural image caption generator*. Computing Research Repository. abs/1411.4555.

Wolpert, D. (1992). Stacked generalization. *Neural networks*, *5*, 241–259.

Wolpert, D. H. (1996a). The existence of a priori distinctions between learning algorithms. *Neural Computation*, *8*, 1391–1420.

Wolpert, D. H. (1996b). The lack of a priori distinctions between learning algorithms, *Neural Computation*, *8*, 1341–1390.

Wolpert, D. H. (2013). Ubiquity symposium: Evolutionary computation and the processes of life: What the no free lunch theorems really mean: How to improve search algorithms. *Ubiquity*, *2*, 2013, 1–15.

Received May 11, 2015; accepted September 2, 2015.