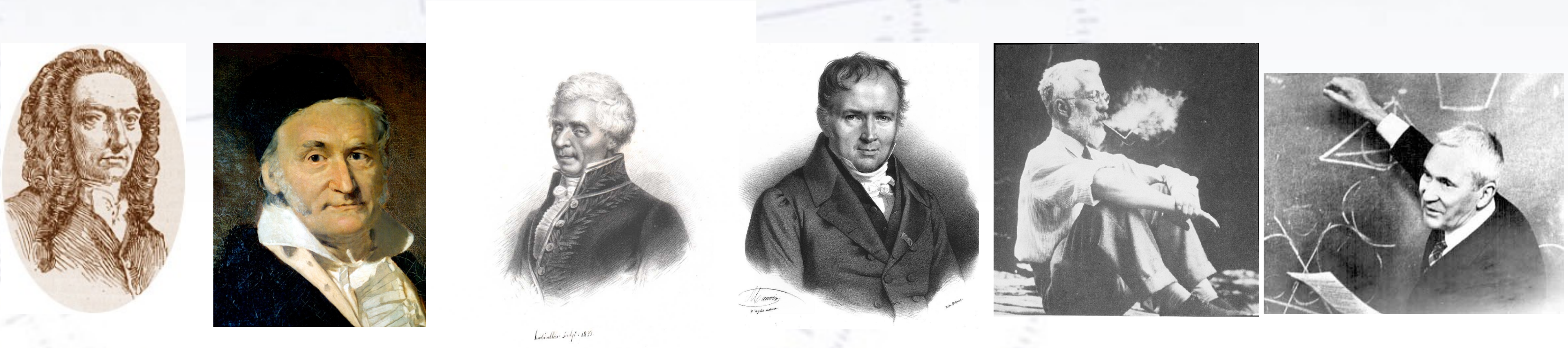
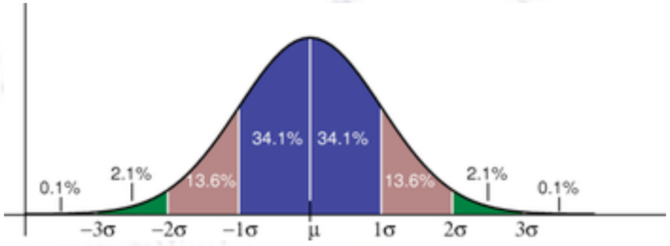
Big Data Analysis

Method performance overview



Troels C. Petersen (NBI)



"Statistics is merely a quantisation of common sense - Big Data is a sharpening of it!"

179 methods vs. 121 data sets

“Tree learning comes closest to meeting the requirements for serving as an off-the-shelf procedure for data mining", because it:

- is invariant under scaling and various other transformations of feature values,
- is robust to inclusion of irrelevant features,
- produces inspectable models.
- HOWEVER… they are seldom accurate (i.e. most performant)!

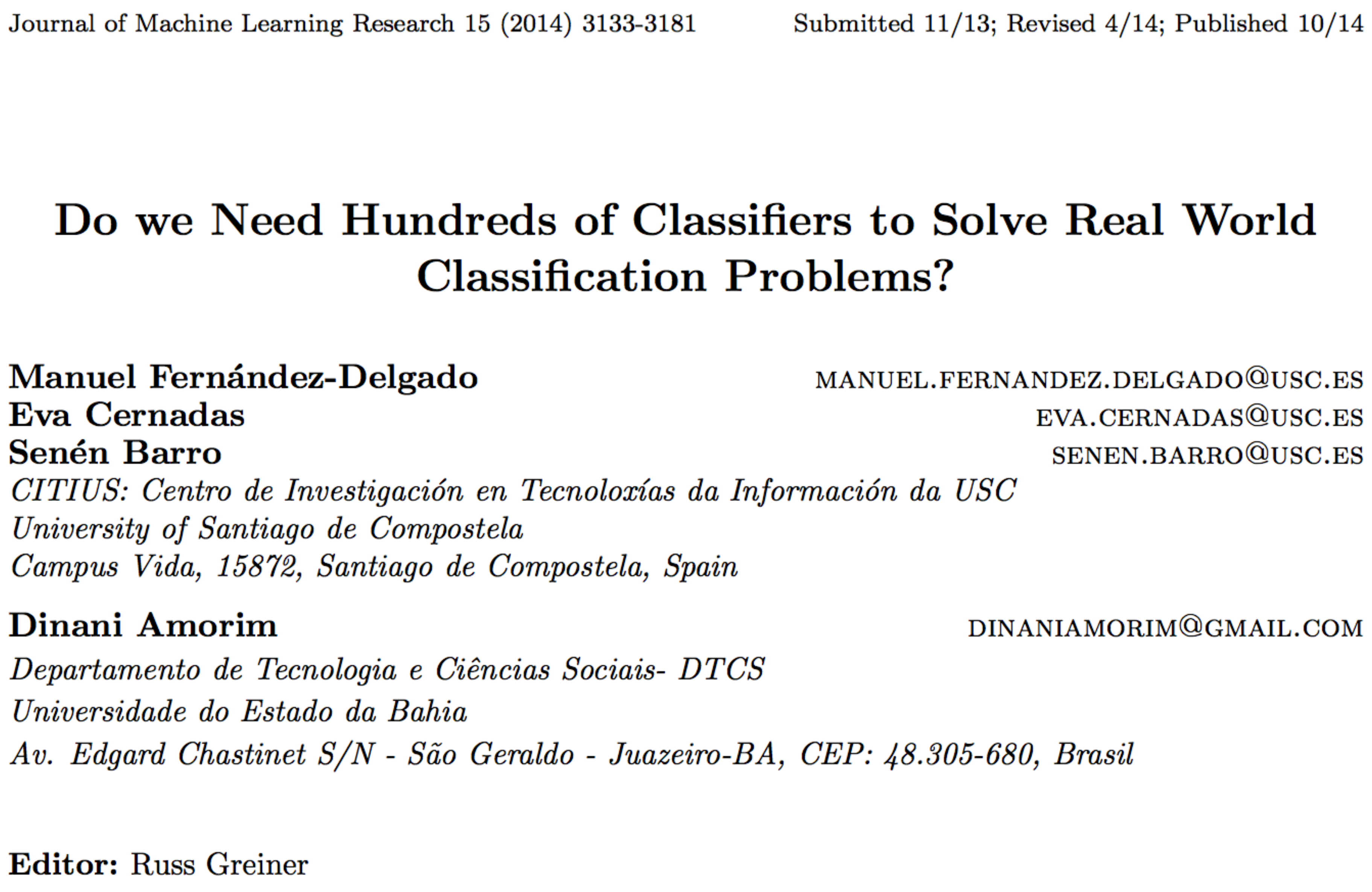
[Trevor Hastie, Professor of Mathematics & Statistics, Stanford University]

In a quite interesting paper, four authors investigated the performance of many Machine Learning (ML) methods (179 in total) on a large variety of data sets (121 in total).

The purpose was to see, if there was any general pattern, and if some type of classifiers were more suited for some problems than others.

Their findings were written up in the following paper…

179 methods vs. 121 data sets



Journal of Machine Learning Research 15 (2014) 3133-3181 Submitted 11/13; Revised 4/14; Published 10/14

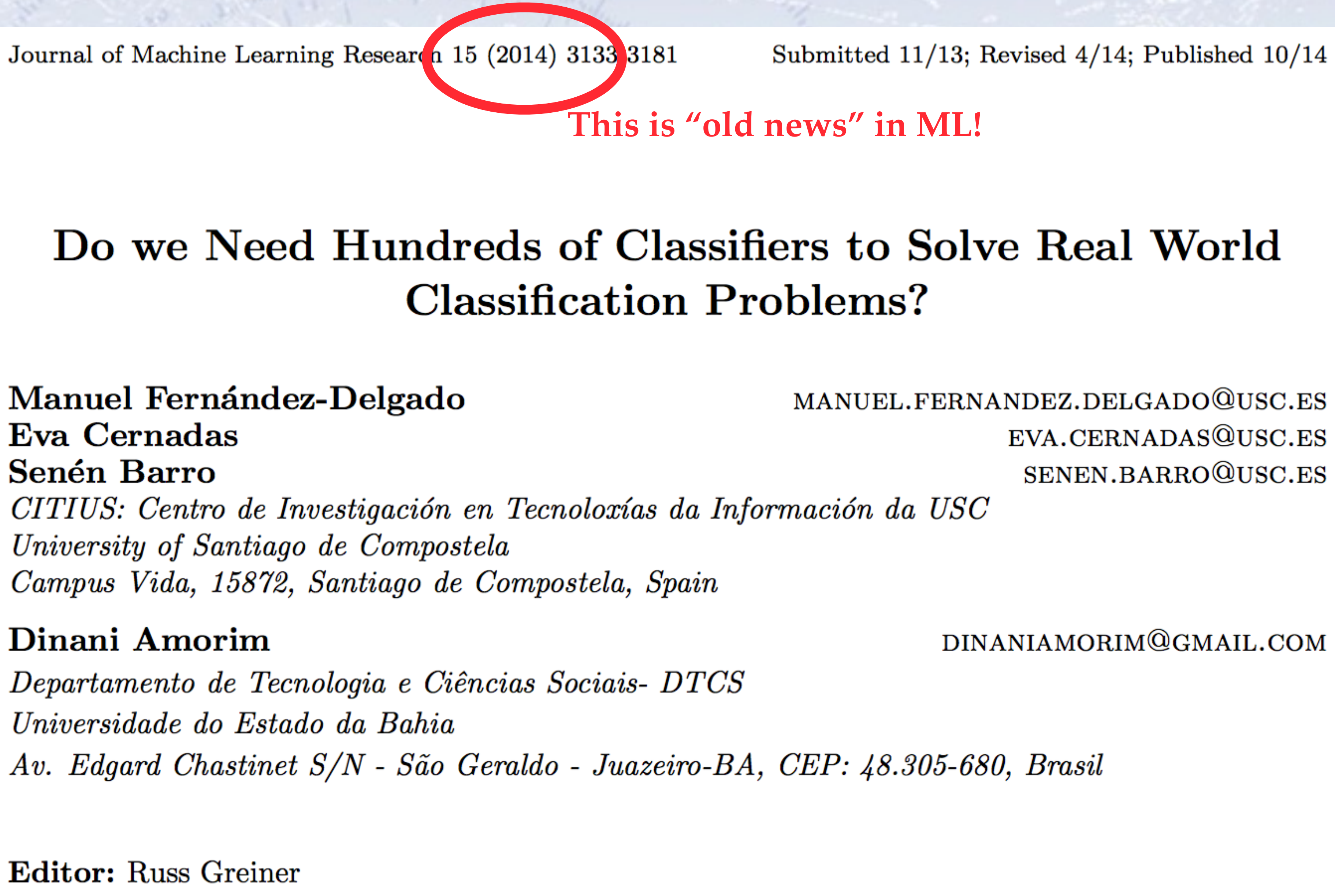
Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?

Manuel Fernández-Delgado MANUEL.FERNANDEZ.DELGADO@USC.ES
Eva Cernadas EVA.CERNADAS@USC.ES
Senén Barro SENEN.BARRO@USC.ES
CITIUS: Centro de Investigación en Tecnoloxías da Información da USC
University of Santiago de Compostela
Campus Vida, 15872, Santiago de Compostela, Spain

Dinani Amorim DINANIAMORIM@GMAIL.COM
Departamento de Tecnologia e Ciências Sociais- DTCS
Universidade do Estado da Bahia
Av. Edgard Chastinet S/N - São Geraldo - Juazeiro-BA, CEP: 48.305-680, Brasil

Editor: Russ Greiner

179 methods vs. 121 data sets



Journal of Machine Learning Research 15 (2014) 3133-3181 Submitted 11/13; Revised 4/14; Published 10/14

This is “old news” in ML!

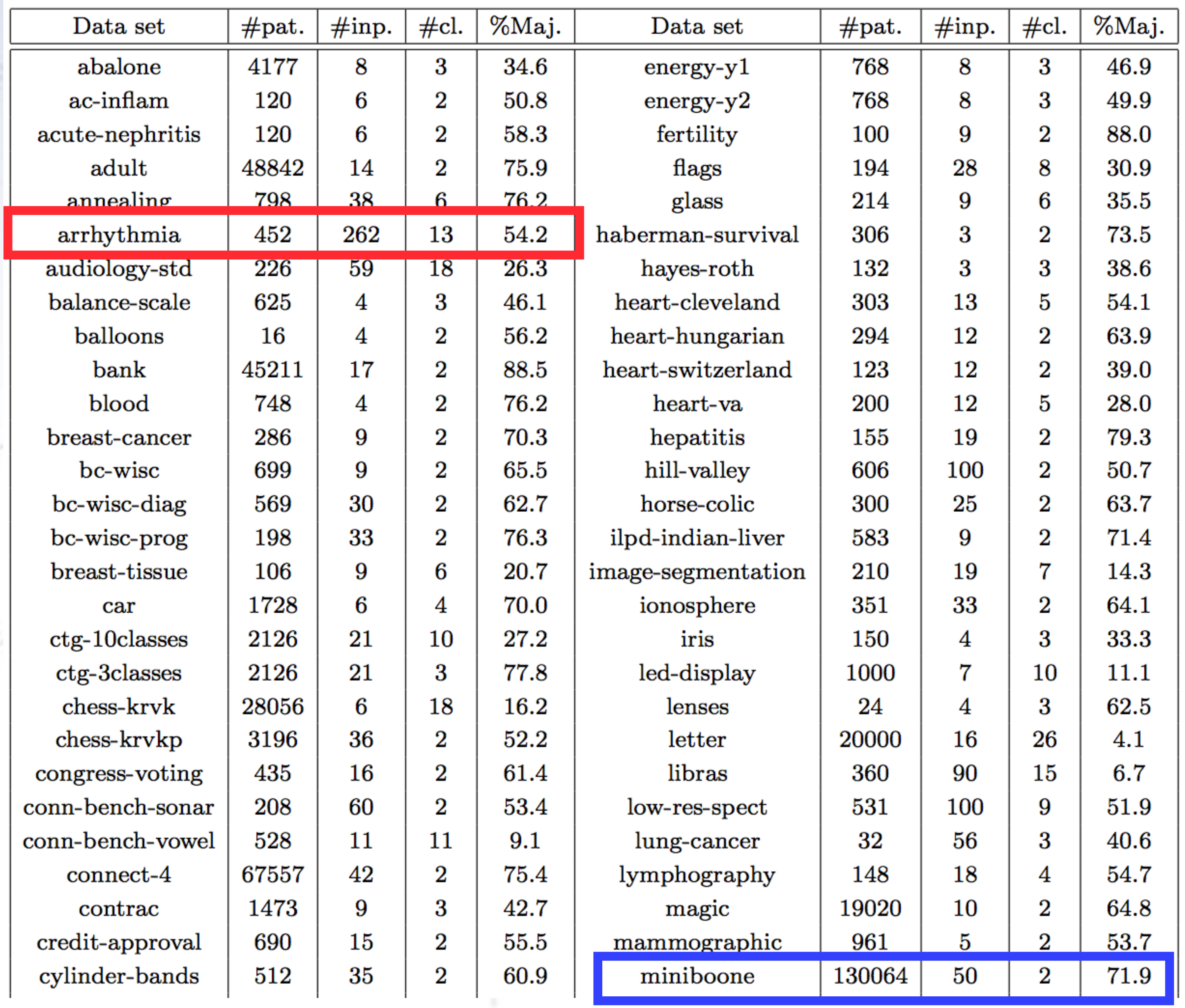
Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?

Manuel Fernández-Delgado MANUEL.FERNANDEZ.DELGADO@USC.ES
Eva Cernadas EVA.CERNADAS@USC.ES
Senén Barro SENEN.BARRO@USC.ES
CITIUS: Centro de Investigación en Tecnoloxías da Información da USC
University of Santiago de Compostela
Campus Vida, 15872, Santiago de Compostela, Spain

Dinani Amorim DINANIAMORIM@GMAIL.COM
Departamento de Tecnologia e Ciências Sociais- DTCS
Universidade do Estado da Bahia
Av. Edgard Chastinet S/N - São Geraldo - Juazeiro-BA, CEP: 48.305-680, Brasil

Editor: Russ Greiner

What are the data sets?



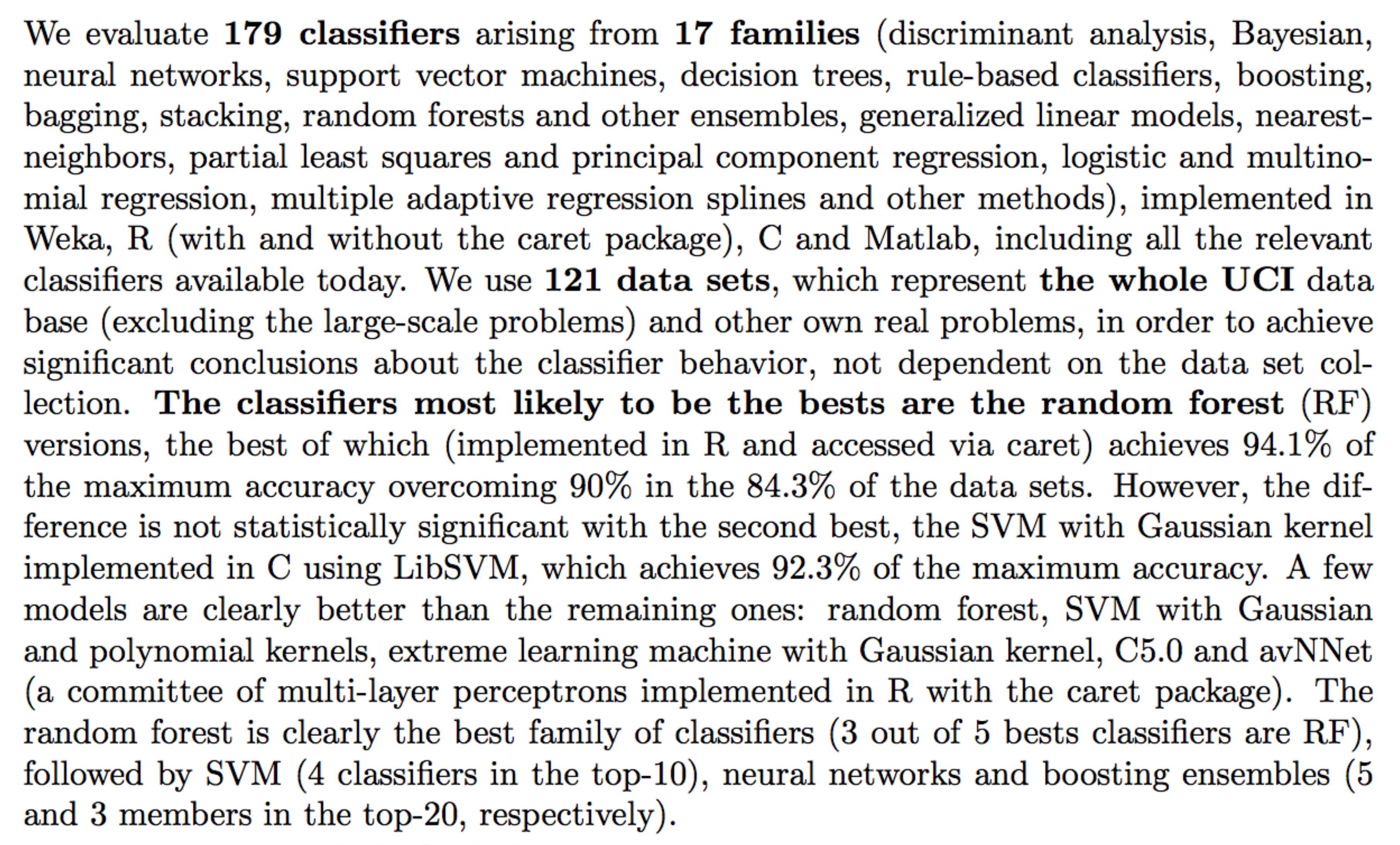
Data set	#pat.	#inp.	#cl.	%Maj.	Data set	#pat.	#inp.	#cl.	%Maj.
abalone	4177	8	3	34.6	energy-y1	768	8	3	46.9
ac-inflam	120	6	2	50.8	energy-y2	768	8	3	49.9
acute-nephritis	120	6	2	58.3	fertility	100	9	2	88.0
adult	48842	14	2	75.9	flags	194	28	8	30.9
annealing	798	38	6	76.2	glass	214	9	6	35.5
arrhythmia	452	262	13	54.2	haberman-survival	306	3	2	73.5
audiology-std	226	59	18	26.3	hayes-roth	132	3	3	38.6
balance-scale	625	4	3	46.1	heart-cleveland	303	13	5	54.1
balloons	16	4	2	56.2	heart-hungarian	294	12	2	63.9
bank	45211	17	2	88.5	heart-switzerland	123	12	2	39.0
blood	748	4	2	76.2	heart-va	200	12	5	28.0
breast-cancer	286	9	2	70.3	hepatitis	155	19	2	79.3
bc-wisc	699	9	2	65.5	hill-valley	606	100	2	50.7
bc-wisc-diag	569	30	2	62.7	horse-colic	300	25	2	63.7
bc-wisc-prog	198	33	2	76.3	ilpd-indian-liver	583	9	2	71.4
breast-tissue	106	9	6	20.7	image-segmentation	210	19	7	14.3
car	1728	6	4	70.0	ionosphere	351	33	2	64.1
ctg-10classes	2126	21	10	27.2	iris	150	4	3	33.3
ctg-3classes	2126	21	3	77.8	led-display	1000	7	10	11.1
chess-krvk	28056	6	18	16.2	lenses	24	4	3	62.5
chess-krvkp	3196	36	2	52.2	letter	20000	16	26	4.1
congress-voting	435	16	2	61.4	libras	360	90	15	6.7
conn-bench-sonar	208	60	2	53.4	low-res-spect	531	100	9	51.9
conn-bench-vowel	528	11	11	9.1	lung-cancer	32	56	3	40.6
connect-4	67557	42	2	75.4	lymphography	148	18	4	54.7
contrac	1473	9	3	42.7	magic	19020	10	2	64.8
credit-approval	690	15	2	55.5	mammographic	961	5	2	53.7
cylinder-bands	512	35	2	60.9	miniboone	130064	50	2	71.9

The data sets are all quite smallish (< 150000 entries), with only 7 / 56 being above 10000 entries!

There are most often between 4-100 input parameters.

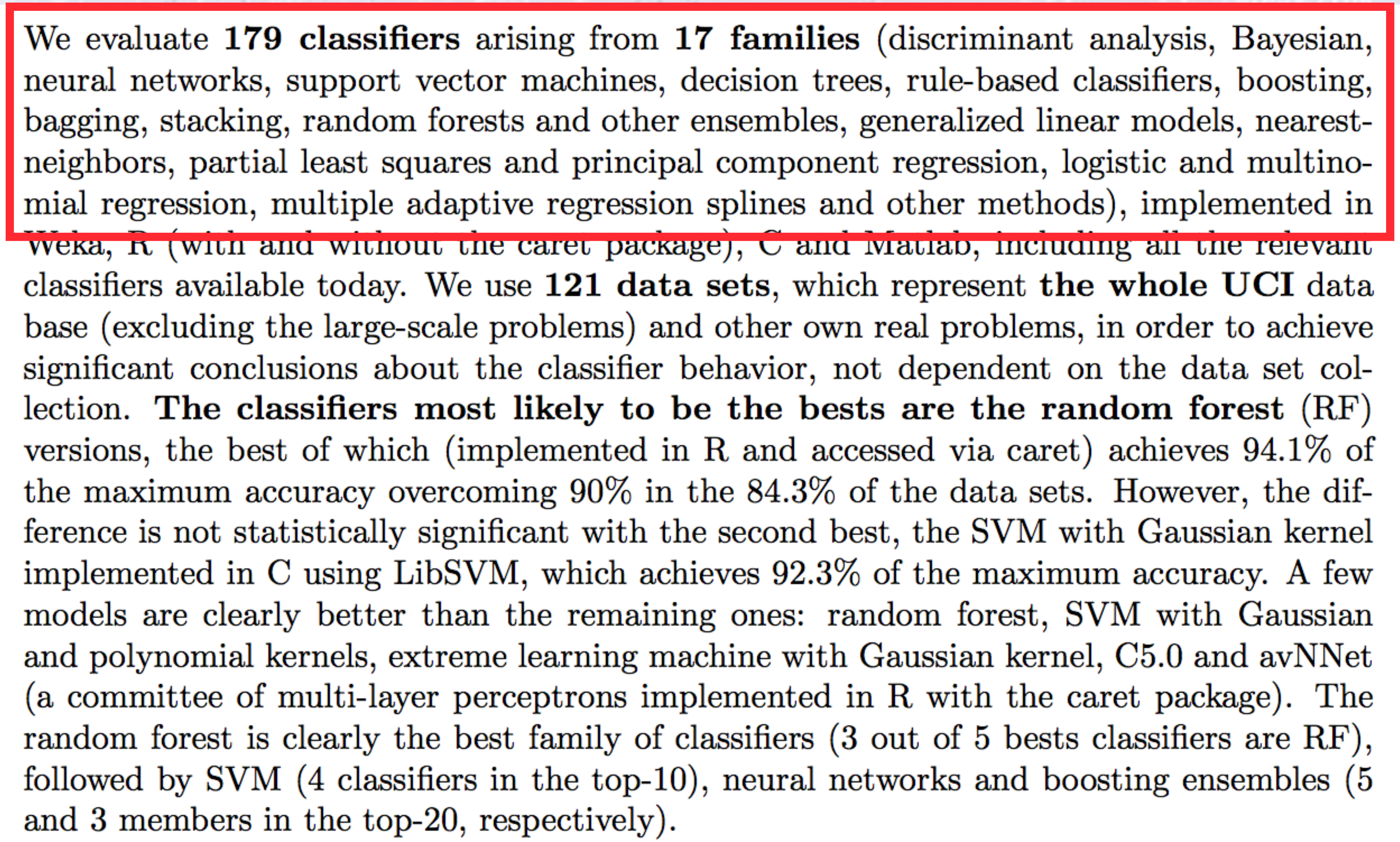
The standard problem is to divide into two classes.

179 methods vs. 121 data sets



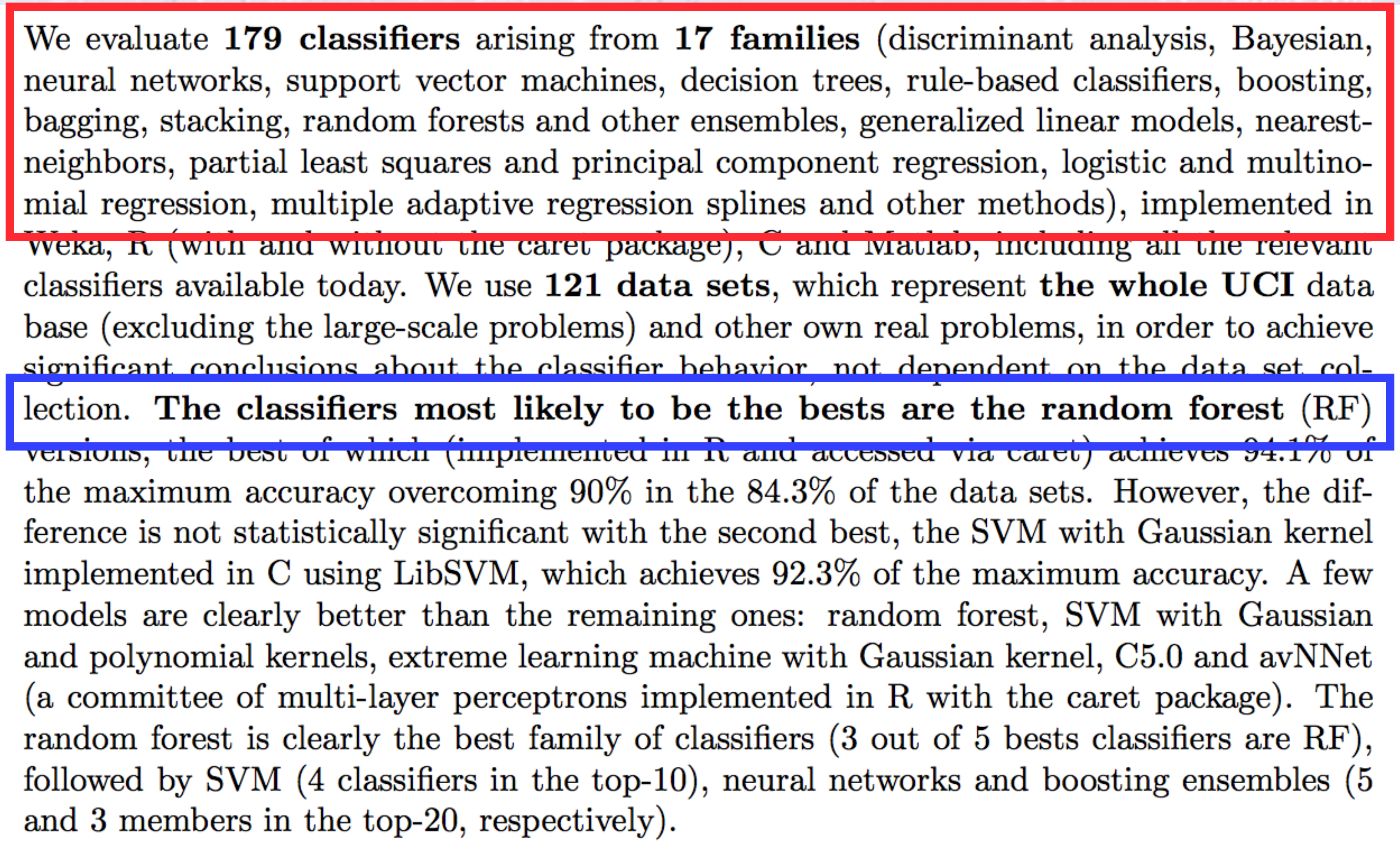
We evaluate **179 classifiers** arising from **17 families** (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, generalized linear models, nearest-neighbors, partial least squares and principal component regression, logistic and multinomial regression, multiple adaptive regression splines and other methods), implemented in Weka, R (with and without the caret package), C and Matlab, including all the relevant classifiers available today. We use **121 data sets**, which represent **the whole UCI** data base (excluding the large-scale problems) and other own real problems, in order to achieve significant conclusions about the classifier behavior, not dependent on the data set collection. **The classifiers most likely to be the bests are the random forest** (RF) versions, the best of which (implemented in R and accessed via caret) achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets. However, the difference is not statistically significant with the second best, the SVM with Gaussian kernel implemented in C using LibSVM, which achieves 92.3% of the maximum accuracy. A few models are clearly better than the remaining ones: random forest, SVM with Gaussian and polynomial kernels, extreme learning machine with Gaussian kernel, C5.0 and avNNet (a committee of multi-layer perceptrons implemented in R with the caret package). The random forest is clearly the best family of classifiers (3 out of 5 bests classifiers are RF), followed by SVM (4 classifiers in the top-10), neural networks and boosting ensembles (5 and 3 members in the top-20, respectively).

179 methods vs. 121 data sets



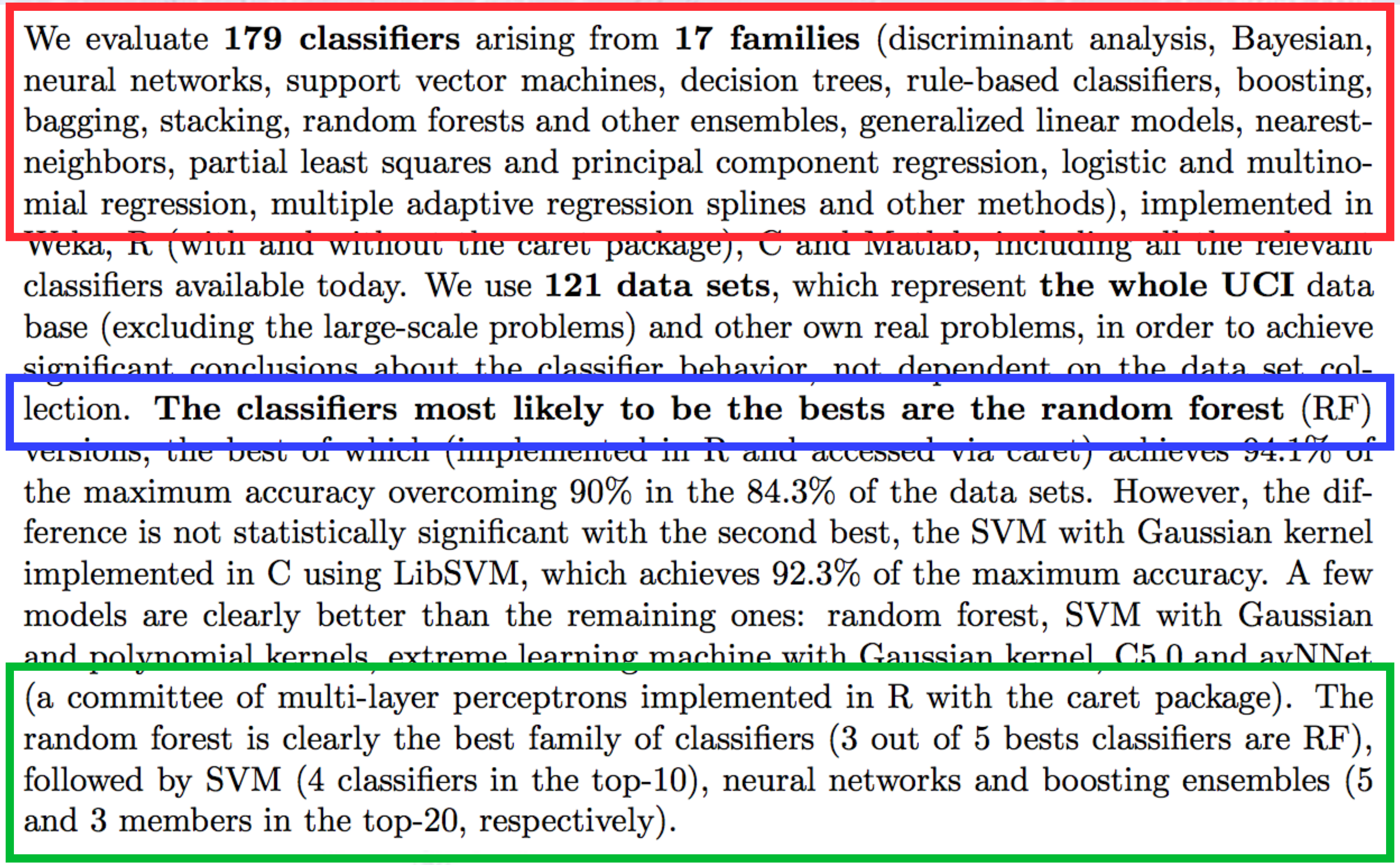
We evaluate **179 classifiers** arising from **17 families** (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, generalized linear models, nearest-neighbors, partial least squares and principal component regression, logistic and multinomial regression, multiple adaptive regression splines and other methods), implemented in Weka, R (with and without the caret package), C and Matlab, including all the relevant classifiers available today. We use **121 data sets**, which represent **the whole UCI** data base (excluding the large-scale problems) and other own real problems, in order to achieve significant conclusions about the classifier behavior, not dependent on the data set collection. **The classifiers most likely to be the bests are the random forest** (RF) versions, the best of which (implemented in R and accessed via caret) achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets. However, the difference is not statistically significant with the second best, the SVM with Gaussian kernel implemented in C using LibSVM, which achieves 92.3% of the maximum accuracy. A few models are clearly better than the remaining ones: random forest, SVM with Gaussian and polynomial kernels, extreme learning machine with Gaussian kernel, C5.0 and avNNet (a committee of multi-layer perceptrons implemented in R with the caret package). The random forest is clearly the best family of classifiers (3 out of 5 bests classifiers are RF), followed by SVM (4 classifiers in the top-10), neural networks and boosting ensembles (5 and 3 members in the top-20, respectively).

179 methods vs. 121 data sets



We evaluate **179 classifiers** arising from **17 families** (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, generalized linear models, nearest-neighbors, partial least squares and principal component regression, logistic and multinomial regression, multiple adaptive regression splines and other methods), implemented in Weka, R (with and without the caret package), C and Matlab, including all the relevant classifiers available today. We use **121 data sets**, which represent **the whole UCI** data base (excluding the large-scale problems) and other own real problems, in order to achieve significant conclusions about the classifier behavior, not dependent on the data set collection. **The classifiers most likely to be the bests are the random forest** (RF) versions, the best of which (implemented in R and accessed via caret) achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets. However, the difference is not statistically significant with the second best, the SVM with Gaussian kernel implemented in C using LibSVM, which achieves 92.3% of the maximum accuracy. A few models are clearly better than the remaining ones: random forest, SVM with Gaussian and polynomial kernels, extreme learning machine with Gaussian kernel, C5.0 and avNNet (a committee of multi-layer perceptrons implemented in R with the caret package). The random forest is clearly the best family of classifiers (3 out of 5 bests classifiers are RF), followed by SVM (4 classifiers in the top-10), neural networks and boosting ensembles (5 and 3 members in the top-20, respectively).

179 methods vs. 121 data sets



We evaluate **179 classifiers** arising from **17 families** (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, generalized linear models, nearest-neighbors, partial least squares and principal component regression, logistic and multinomial regression, multiple adaptive regression splines and other methods), implemented in Weka, R (with and without the caret package), C and Matlab, including all the relevant classifiers available today. We use **121 data sets**, which represent **the whole UCI** data base (excluding the large-scale problems) and other own real problems, in order to achieve significant conclusions about the classifier behavior, not dependent on the data set collection. **The classifiers most likely to be the bests are the random forest** (RF) versions, the best of which (implemented in R and accessed via caret) achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets. However, the difference is not statistically significant with the second best, the SVM with Gaussian kernel implemented in C using LibSVM, which achieves 92.3% of the maximum accuracy. A few models are clearly better than the remaining ones: random forest, SVM with Gaussian and polynomial kernels, extreme learning machine with Gaussian kernel, C5.0 and avNNet (a committee of multi-layer perceptrons implemented in R with the caret package). The random forest is clearly the best family of classifiers (3 out of 5 bests classifiers are RF), followed by SVM (4 classifiers in the top-10), neural networks and boosting ensembles (5 and 3 members in the top-20, respectively).

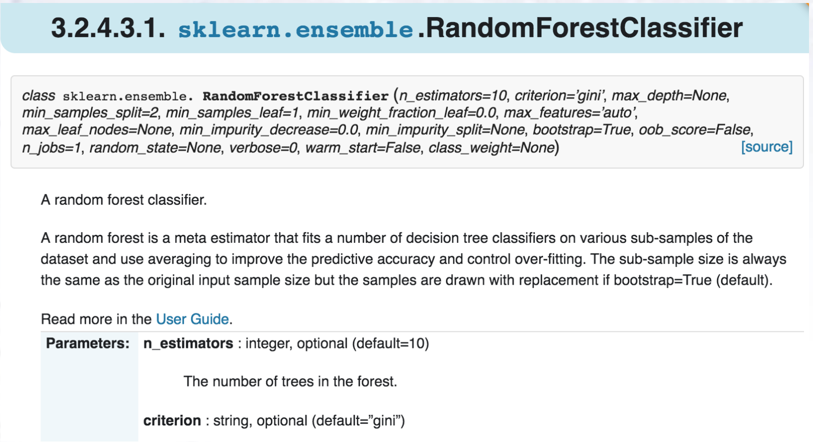
Random Forests implementations

Given the succes of the RandomForests algorithm, it has naturally been implemented in many languages (the original one being Fortran!!!).

I managed to find it in both Python and R:

Python: scikit-learn package

R: randomForests package



3.2.4.3.1. sklearn.ensemble.RandomForestClassifier

class sklearn.ensemble. **RandomForestClassifier** (*n_estimators=10, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None*) [source]

A random forest classifier.

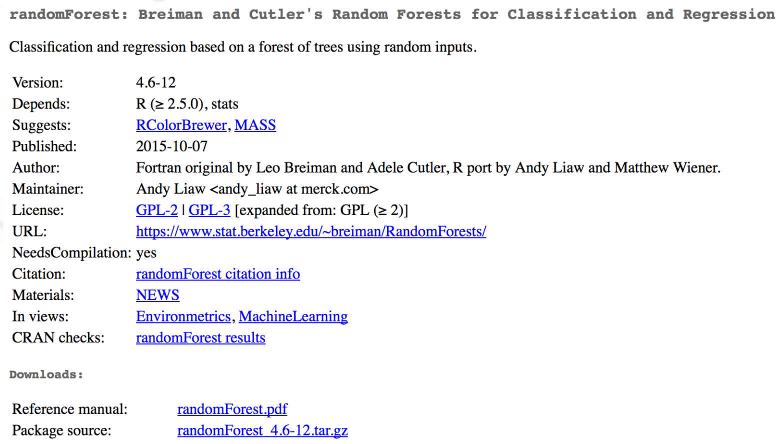
A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True (default).

Read more in the User Guide.

Parameters: **n_estimators** : integer, optional (default=10)

The number of trees in the forest.

criterion : string, optional (default="gini")



randomForest: Breiman and Cutler's Random Forests for Classification and Regression

Classification and regression based on a forest of trees using random inputs.

Version:	4.6-12
Depends:	R (≥ 2.5.0), stats
Suggests:	RColorBrewer, MASS
Published:	2015-10-07
Author:	Fortran original by Leo Breiman and Adele Cutler, R port by Andy Liaw and Matthew Wiener.
Maintainer:	Andy Liaw <andy_liaw at merck.com>
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)]
URL:	https://www.stat.berkeley.edu/~breiman/RandomForests/
NeedsCompilation:	yes
Citation:	randomForest citation info
Materials:	NEWS
In views:	Environmetrics, MachineLearning
CRAN checks:	randomForest results

Downloads:

Reference manual:	randomForest.pdf
Package source:	randomForest_4.6-12.tar.gz

The results in more detail…

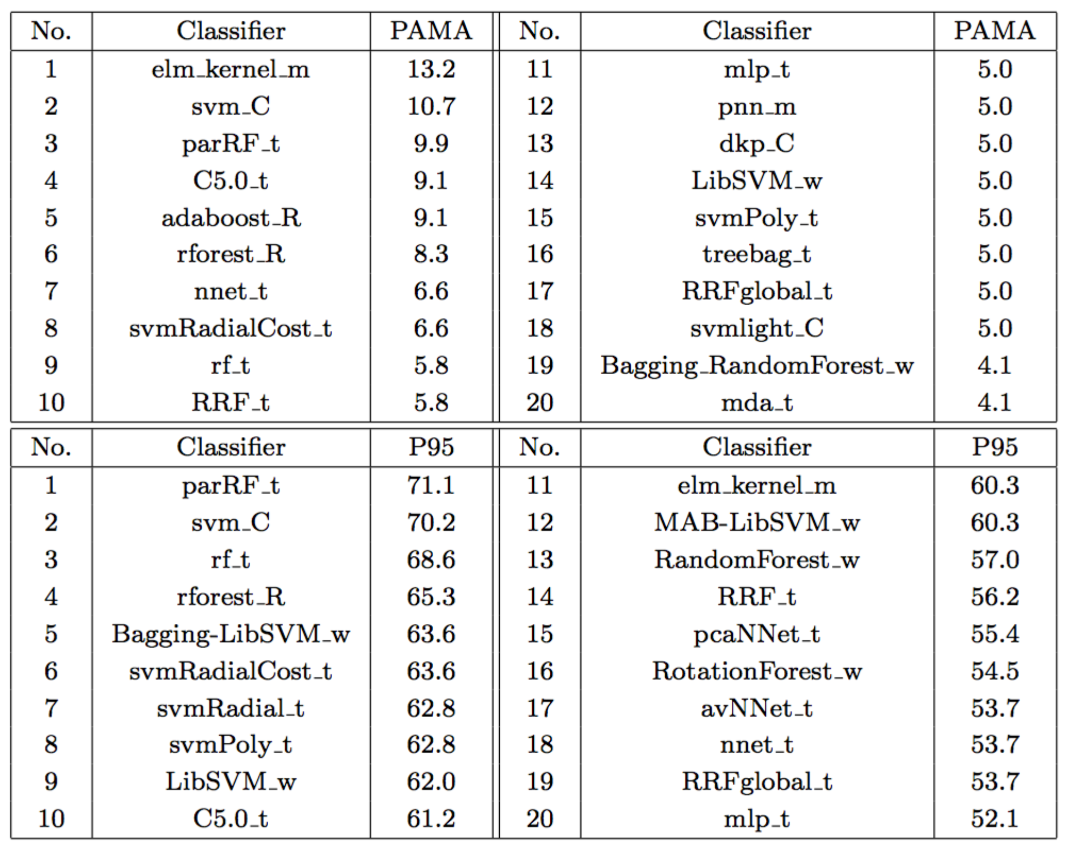
The many good algorithms are ranked according to probability of achieving:

- Maximum Accuracy (PAMA)
- 95% accuracy on all data sets (P95)

As can be seen, the Random Forest (parRF_t) is not the most likely to be the best.
Rather it is the one, which most often is ranked high.

But this just shows, that there is no guarantee that parRF_t is the most powerful method. In fact far from it.

This is a general problem, which must be considered…



No.	Classifier	PAMA	No.	Classifier	PAMA
1	elm_kernel_m	13.2	11	mlp_t	5.0
2	svm_C	10.7	12	pnn_m	5.0
3	parRF_t	9.9	13	dkp_C	5.0
4	C5.0_t	9.1	14	LibSVM_w	5.0
5	adaboost_R	9.1	15	svmPoly_t	5.0
6	rforest_R	8.3	16	treebag_t	5.0
7	nnet_t	6.6	17	RRFglobal_t	5.0
8	svmRadialCost_t	6.6	18	svmlight_C	5.0
9	rf_t	5.8	19	Bagging_RandomForest_w	4.1
10	RRF_t	5.8	20	mda_t	4.1

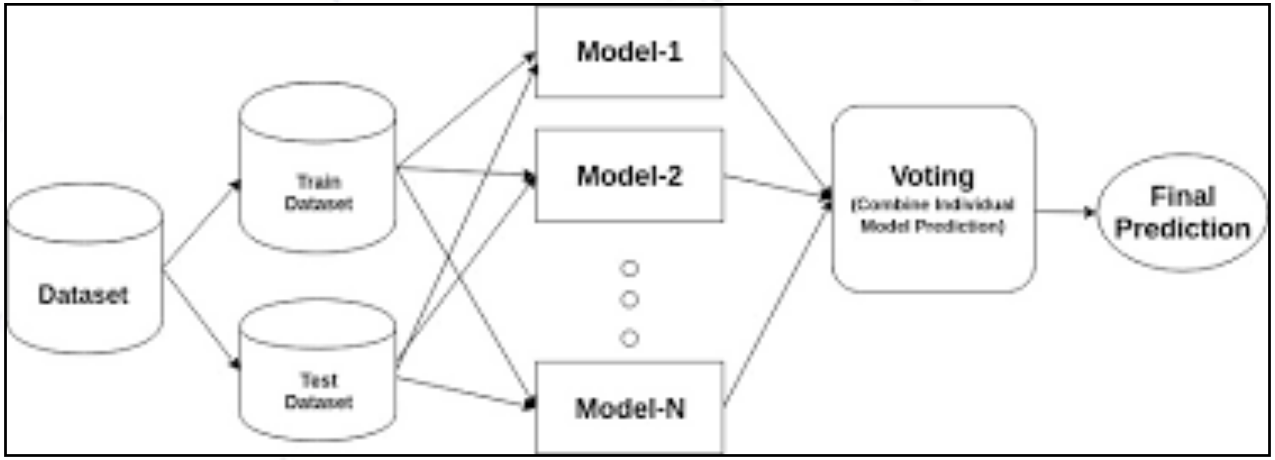
No.	Classifier	P95	No.	Classifier	P95
1	parRF_t	71.1	11	elm_kernel_m	60.3
2	svm_C	70.2	12	MAB-LibSVM_w	60.3
3	rf_t	68.6	13	RandomForest_w	57.0
4	rforest_R	65.3	14	RRF_t	56.2
5	Bagging-LibSVM_w	63.6	15	pcaNNet_t	55.4
6	svmRadialCost_t	63.6	16	RotationForest_w	54.5
7	svmRadial_t	62.8	17	avNNet_t	53.7
8	svmPoly_t	62.8	18	nnet_t	53.7
9	LibSVM_w	62.0	19	RRFglobal_t	53.7
10	C5.0_t	61.2	20	mlp_t	52.1

Ensemble method

Different methods have different advantages, and for that reason the very best performance is often obtained by “ensemble methods”.

Here, several different ML methods are used on data, and subsequently their results are combined in a new “ensemble” ML algorithm (or by voting!), which benefits from all the advantages.

These have lately been the most performant methods (i.e. winning competitions). However, they are cumbersome (you have to optimise many methods), and typically a single method reaches close to the information limit.


Dataset
Train Dataset
Test Dataset
Model-1
Model-2
Model-N
Voting
(Combine Individual Model Prediction)
Final Prediction