A Numeric Comparison of Feature Selection Algorithms for Supervised Learning

G. Palombo*

University of Milano-Bicocca

I. Narsky

California Institute of Technology

* Work done at Caltech

OUTLINE

- Feature Selection in Supervised Learning
- Statistical Packages: StatPatternRecognition, R, Weka
- Data
- Analysis and Results
- Conclusions

THE RESULTS OF THIS ANALYSIS HAVE BEEN SUBMITTED FOR PUBLICATION TO THE JOURNAL "STATISTICS AND COMPUTING".

SUPERVISED LEARNING (SL)

- The task is to separate signal from background by a learning process on training sets where signal and background are known.
- Many different classifiers (NN, RF, SVM, BDT, k-NN, etc…) have been developed to accomplish this task.
- The best classifier depends on the characteristics of the problem.

FEATURE SELECTION (FS)

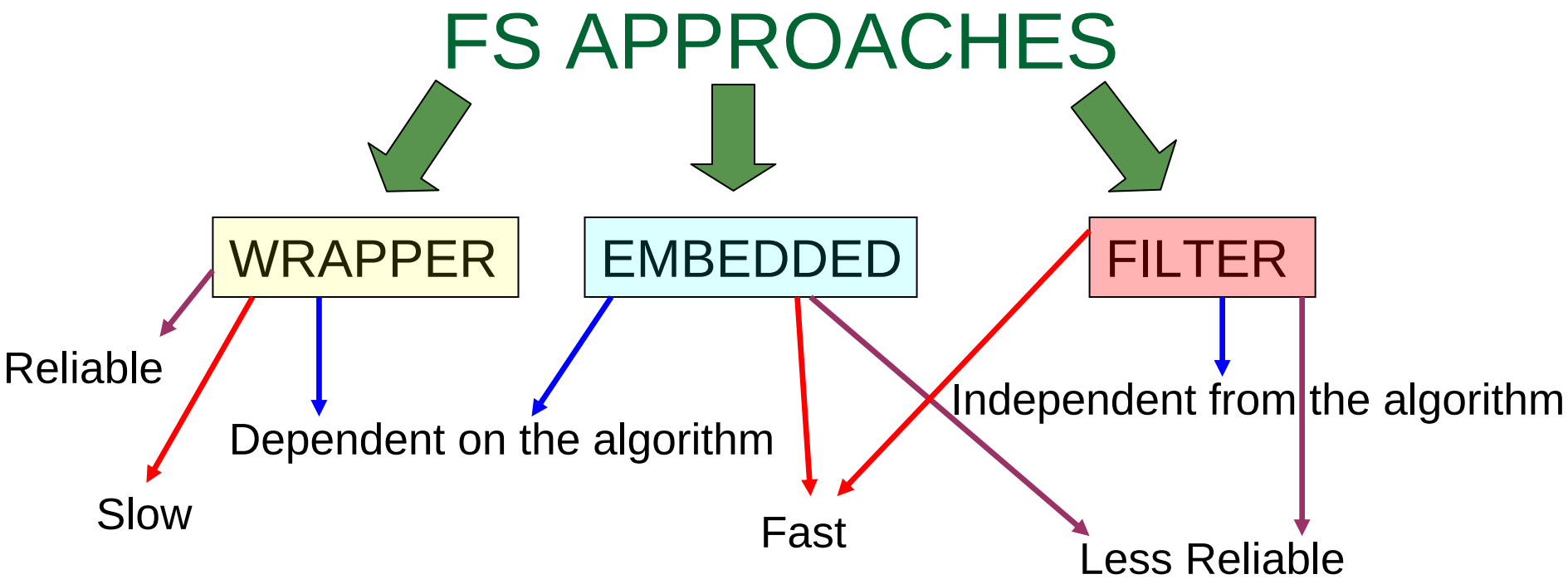
- Selection of the most powerful discriminating features (variables)

- Usually not all the features are useful for the classification problem. In modern applications, where the number of instances (events) can be huge, it is important to evaluate whether it is possible to find irrelevant features.

- FS addresses the problem of reducing the feature set to the smallest subset that gives the same or better quality of separation between signal and background as the full set does.

WHY FEATURE SELECTION ?

- Reduces data analysis time.
- Easier analyzing and interpreting the results.
- Can improve performance (for some algorithms).



WHAT TO CONSIDER

Predictive power of the selected subset

Size of the subset

Speed of the FS algorithm

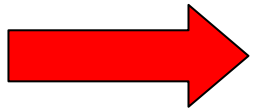
No method is universally better

CPU time and reliability are at odds. Again, no definitive rule. It depends on the goal of the analysis, the specific dataset, etc…

FS METHODS (1)

Filter

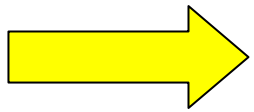
Correlations



The larger the correlation with the class label, the more important the feature.

Wrapper

Add n Rem r



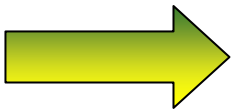
Start with a null set. Add n variables at a time. For each added variable, train the classifier and compute FOM. Choose the variables that improve FOM most. Then remove r variables that improve FOM least. Continue as long as it is possible to improve FOM.

FS METHODS (2)

Embedded

FOM Importance 

Works with tree-based algorithms. Adding up the improvement in the FOM over all the splits gives a feature importance estimate.

Permutations 

A trained classifier is applied to instances not included in the data used for training and its performance is evaluated. Then, this classifier is applied to these instances with class labels randomly permuted across each variable in turn and the change in the performance measure due to this permutation for each variable is estimated. The higher is this change, the more important is the feature.

STATISTICAL ANALYSIS TOOLS

- The package mainly used in this analysis is StatPatterRecogniton (SPR), a C++ package for SL developed at Caltech by Ilya Narsky (http://www.hep.caltech.edu/~narsky/spr.html).
- SPR is distributed under General Public License off Sourceforge (http://sourceforge.net/projects/statpatrec).
- Two versions of the package: standalone version which uses ASCII text for input and output data, and a ROOT-dependent version.

SPR CLASSIFIERS

- LDA and QDA
- Decision splits
- Bump Hunter
- Decision Trees (2 flavors)
- Feedforward backpropagation neural net with a logistic activation function
- Several flavors of boosting: discrete AdaBoost, real AdaBoost, epsilon-boost, and arc-x4 algorithm (Breiman version of boosting)
- Bagging
- Random Forest
- Bagging and boosting an arbitrary sequence of classifiers
- Algorithm for combining classifiers trained on subsets of input variables
- Multiclass learners: Allwein-Schapire-Singer and Binary Encoder

SPR: OTHER TOOLS

- Crossvalidation
- Bootstrap
- FS methods (the 4 previously decribed and others) and variable transformation
- Computation of data moments (mean, variance, covariance, correlations betwen variables, etc…)
- Up to 10 FOMs to optimize the classifiers
- Arbitrary grouping of input classes in 2 categories (signal and background)

STATISTICAL PACKAGES

SPR

- ✓ Supported on Unix
- ✓Command line
- ✓ C++
- ✓ Faster on big datasets
- ✓ Many different FOMs
- ✓ More flexible: boosting and bagging an arbitrary sequence of classifiers, generalized forward addiction, multiclass learner with any kind of classifier, etc…

R

http://cran.r-project.org/

- ✓ Supported on many platforms
- ✓ Command line
- ✓ R or implemented in C, C++ and interfaced into R
- ✓ Implemented in R is slow, but easy to interpretate. Otherwise, faster but less interpretable.
- ✓ Extensive on-line documentation
- ✓ Huge number of statistical tools
- ✓ Less flexible

WEKA

http://www.cs.waikato.ac.nz/~ml/weka/

- ✓ Supported on many platforms
- ✓ Command line and GUI
- ✓ Java
- ✓ Slow on big datasets
- ✓ Easy graphical interface, fast to learn
- ✓ Less flexible

SPR better for more complex analyses.
R / WEKA good for easier analyses.

GOAL OF THE ANALYSIS AND DATA

- Comparing FS methods implemented in SPR to each other and with other methods implemented in statistical packages R and Weka.
- Comparing our results with previously published results for the same datasets.
- HEP datasets usually have many events with few input variables, but typically these datasets are not public.
- We use the datasets Magic Telescope, Cardiac Arrhythmia, WDBC, WBC, Colic Horse which are (with the exception of Magic Telescope) much smaller than typical HEP dataset. But they are all available publicly at:

  *www.ics.**uci**.edu/~mlearn/ML**Repository**.html*

DATASET

- Magic Gamma-ray Telescope

  Monte Carlo simulated events, 19020 events and 10 features. Binary classification.
- Cardiac Arrhythmia

  Classification of the patient into one of the 12 classes of cardiac arrhythmia, 420 events and 261 features.
- Wisconsin Diagnostic Breast Cancer (WDBC)

  Binary problem (benign or malignant tumor), 569 events and 30 features.
- Wisconsin Breast Cancer (WBC)

  Binary problem (benign or malignant tumor), 699 events and 9 features.
- Colic Horse

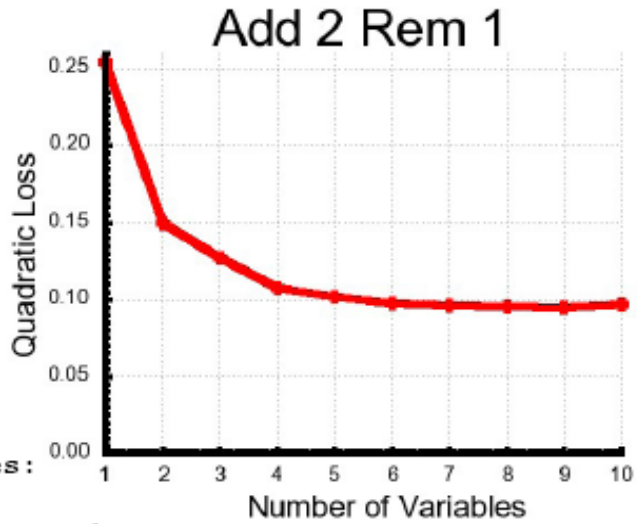
  Binary problem (whether the lesion is surgical), 368 events and 22 features.

RESULTS FROM MAGIC DATASET

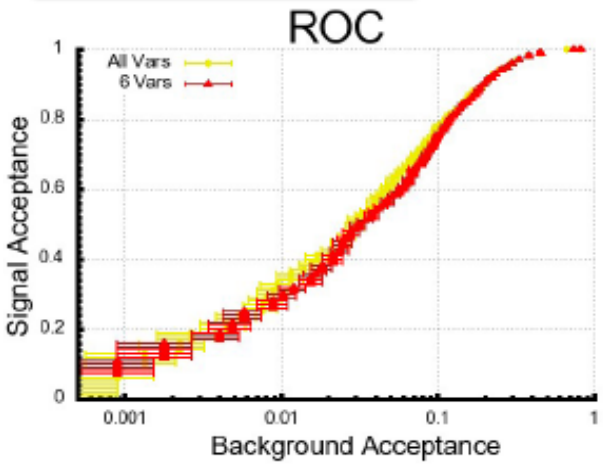
19020 events

10 input variables:
- Alpha
- Size
- Length
- Width
- Conc
- Conc1
- M3Long
- Dist
- Asym
- M3Trans

Using the best model:
Random Forest of 200 trees with 3 variables randomly selected for each split



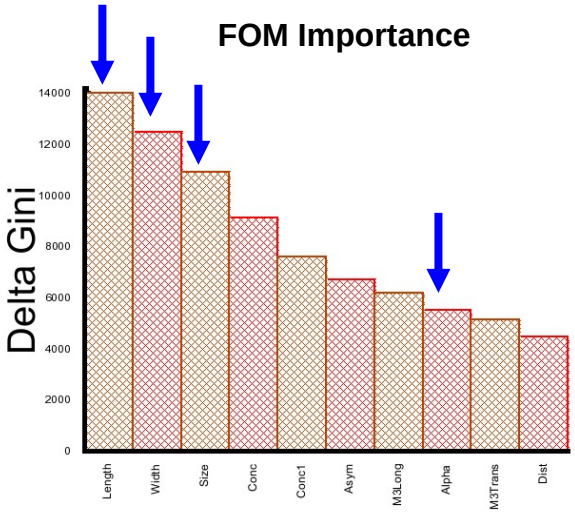
Variables:
1 Alpha
2 Alpha Length
3 Alpha Size Width
4 Alpha Length Size Width
5 Alpha Conc Length Size Width
6 Alpha Conc Dist Length Size Width
7 Alpha Conc Dist Length M3Long Size Width

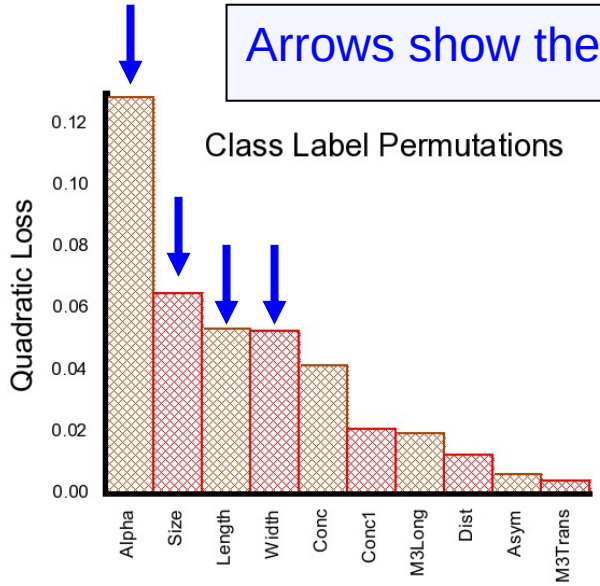


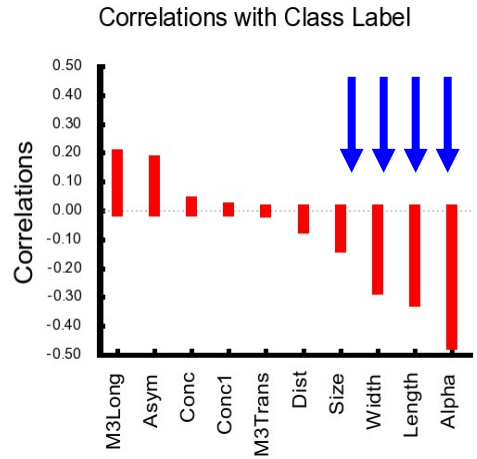
Conclusion:
- **4 variables can be eliminated without noticeable loss of predictive power**
- **Alpha Length Size Width are the 4 most powerful variables.**

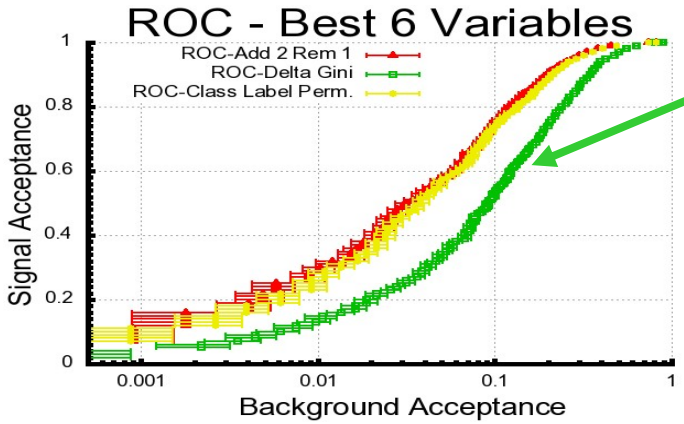
AND OTHER FS METHODS?

Arrows show the 4 most powerful variables.









FOM Importance does not select the most powerful variable and has the worst performance. All other methods achieve comparable performance selecting the 4 most powerful variables.

ARRHYTHMIA DATASET (1)

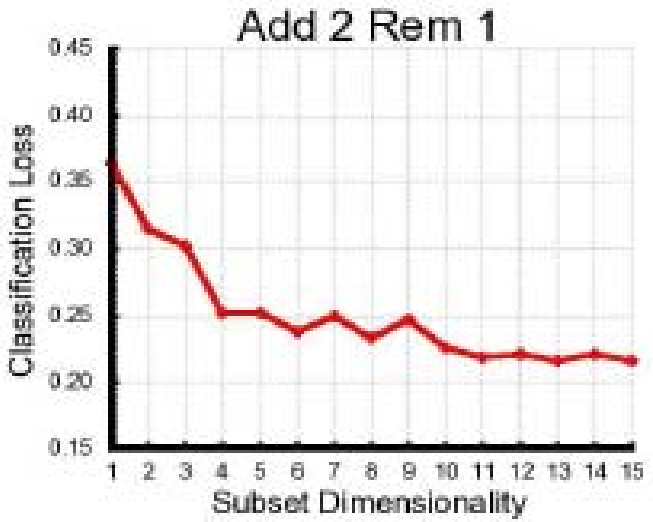
- Multi-class classification problem – 12 classes.
- We use Allwein-Shapire-Singer algorithm to reduce a multi-class problem to a set of binary ones and then convert the solutions to these binary problems into an overall multi-class classification label.
- Binary classifiers are built such that they first separate class “Normal” (the most numerous class) from all other classes which were more likely to be misclassified as “Normal”. Thereafter, they separate the remaining classes among themselves.
- We give different weights to different binary classifiers to achieve the best possible accuracy.
- High number of features (261): the advantage of a generalized forward addiction selection, which takes into account the combined effect of several variables, is more evident.

ARRHYTHMIA DATASET (2)

420 events

Matrix of binary classifiers (simple tres). Columns are the classifers and rows the classes and weights (last one).

row/col	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	0	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
2	-1	1	1	0	-1	-1	-1	-1	-1	-1	-1	-1	0	0
3	0	0	0	0	1	-1	-1	-1	-1	-1	-1	-1	-1	0
4	-1	0	0	0	-1	1	-1	-1	-1	-1	-1	-1	-1	1
5	-1	0	0	0	-1	-1	1	-1	-1	-1	-1	-1	-1	0
6	0	0	0	0	-1	-1	-1	1	-1	-1	-1	-1	-1	0
7	0	0	0	0	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
8	0	0	0	0	-1	-1	-1	-1	1	-1	-1	-1	-1	0
9	0	0	0	0	-1	-1	-1	-1	-1	1	-1	-1	-1	0
10	0	0	0	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	0
14	-1	0	0	0	-1	-1	-1	-1	-1	-1	-1	1	0	0
16	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	1	0
W	1	7	2	5	1	1	1	1	1	1	1	1	1	5



250 features out of 261 not needed

Multiclass algorithm set to Allwein-Schapire-Singer

Variable	Add(+)/Rem(-)	Loss
HeartRate	+	0.37381
V1W.RApex	+	0.314286
V1W.RApex	-	0.378571
V1A.RApex	+	0.316667
V3A.Q	+	0.278571
V3A.Q	-	0.314286
AVRA.T	+	0.27619
V3A.R	+	0.242857
V3A.R	-	0.297619
V3W.R	+	0.245238
V1A.QRSA	+	0.235714
V1A.QRSA	-	0.269048
V3A.T	+	0.235714
DIIW.Q	+	0.22619
V3A.T	-	0.242857
V3W.S	+	0.213889
DIW.TRag	+	0.216667
V3W.S	-	0.233333
AVRW.TDiph	+	0.216667
V3W.S	+	0.202186
DIW.TRag	-	0.221429
DIW.PDiph	+	0.206612
V1A.Q	+	0.195055
AVRW.TDiph	-	0.228571
DIIA.P	+	0.207182
V3A.QRSTA	+	0.217033
V1A.Q	-	0.233333
DIIIW.Defl	+	0.211111
V2W.R	+	0.209945
DIIIW.Defl	-	0.219048
V4A.P	+	0.211538
V4W.R	+	0.211905
V4W.R	-	0.230952

Order in which the features are added and removed by Add2Rem1.

ARRHYTHMIA DATASET (3)

Accuracy with different FS methods.

Features set	accuracy (%)	p-value
Best 11 Add2Rem1	80.95	
Best 11 Add1Rem0	76.19	0.49 (-)
Best 3 Add2Rem1	71.90	0.04 (-)
Best 4 Add2Rem1	75.24	0.22 (-)
All 261 SPR	75.95	0.10 (-)
Best 25 ReliefF	72.14	0.20 (-)
Best 25 CFS-SF	73.33	0.54 (-)
Best 5 FCBF	69.05	0.01 (-)
Full Dataset *	68.80	
Best 25 ReliefF *	68.80	
Best 25 CFS-SF *	69.02	
Best 5 FCBF *	71.47	
Full Dataset 95-NN †	56.92 ± 7.70	
Best 96 95-NN †	56.92 ± 7.70	
Full DataSet 7-NN †	65.38 ± 7.20	
Best 96 5-NN †	63.65 ± 4.39	

P-value calculated through 10-fold cross-validated paired t test (T. G. Dietterich, Neural Computation 10, 1895-1923 (1998)).
(-) indicates that the best 11 variables model is better than the model it is compared with. P-value smaller or equal to 0.05 indicates that the model with 11 variables is statistically, at 0.05 level, better.

Results from SPR

Our results from Weka

Results from L. YU and H. Liu, Journ. Of Machine Learning Research 5, 1205-1224 (2004).

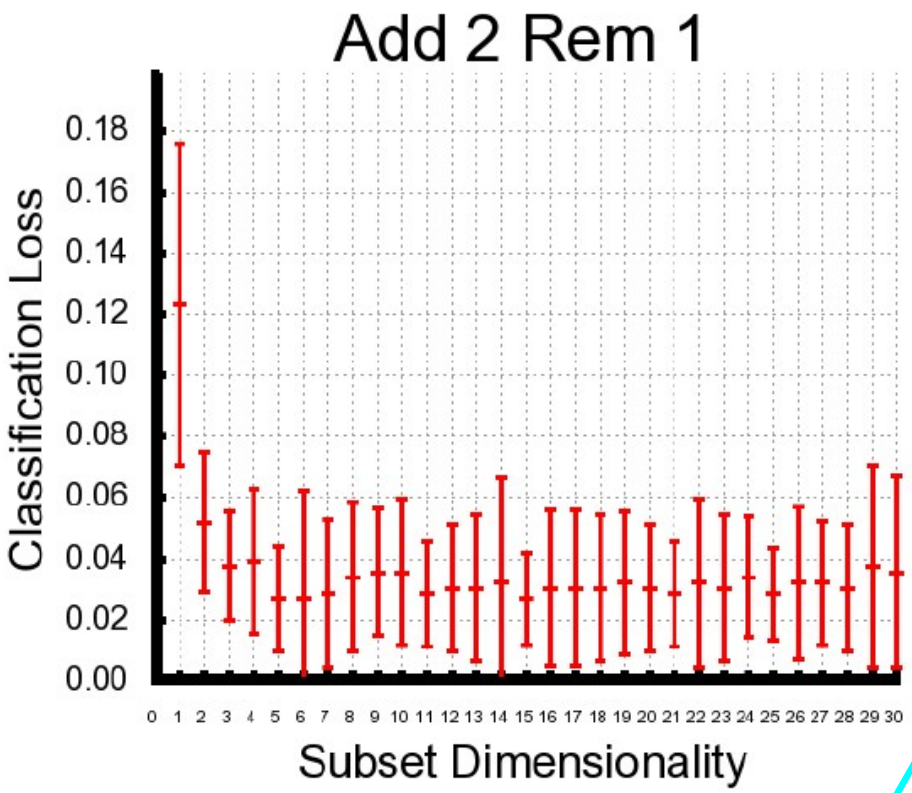
Results from H-L. Wei and S. Billings, IEEE Trans. Pattern Analysis and Machine Intel. 29 (1), 162-166 (2007).

WDBC DATASET

569 events

Classifier: RF with 50 trees, 2 instances per node and 23 feature randomly chosen.

Add 2 Rem 1 and Permutations achieve the best results.



Features set	accuracy (%)
Full Dataset Random Forest	96.25 ± 3.09
Full Dataset RuleFit	95.78
Best 3 Add2Rem1	96.07 ± 2.77
Best 3 Add1Rem0	94.89 ± 4.29
Best 3 Correlations	93.10 ± 2.65
Best 3 Permutations	96.21 ± 2.63
Best 3 RuleFit	93.85
Best 3 FOM Importance	91.25 ± 3.20
Full Dataset†	97.94 ± 1.67 (5-NN)
13 Features †	97.04 ± 1.65 (7-NN)

27 features out of 30 not needed

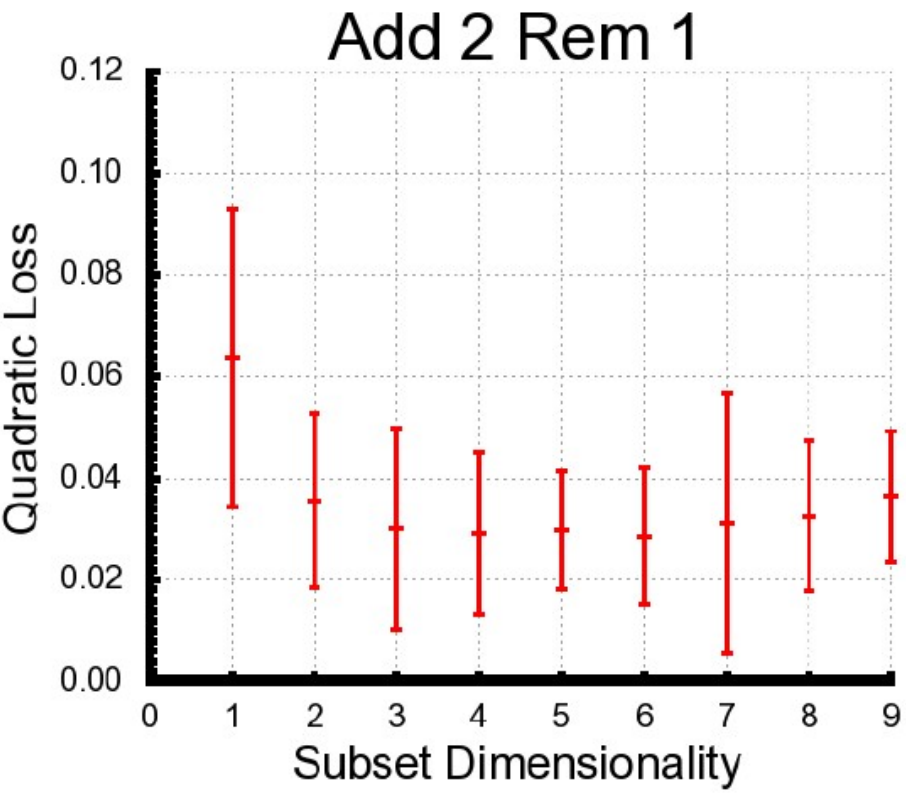
Results from H-L. Wei and S. Billings, IEEE Trans. Pattern Analysis and Machine Intel. 29 (1), 162-166 (2007).

WBC DATASET

699 events

Classifier: AdaBoost with binary splits with 100 cylcles.

Easy dataset: all the methods achieve comparable results.



Features set	accuracy (%)
Full Dataset Boosted Decision Split	96.24 ± 2.59
Full Dataset RuleFit	95.82
Best 3 Add2Rem1	95.77 ± 1.71
Best 3 Add1Rem0	95.77 ± 1.71
Best 3 Correlations	95.28 ± 2.54
Best 3 Permutations	95.72 ± 2.98
Best 3 RuleFit	95.61
Full Data Set †	98.16 ± 2.03 (5-NN)
4 Features †	97.42 ± 2.16 (15-NN)

6 features out of 9 not needed.

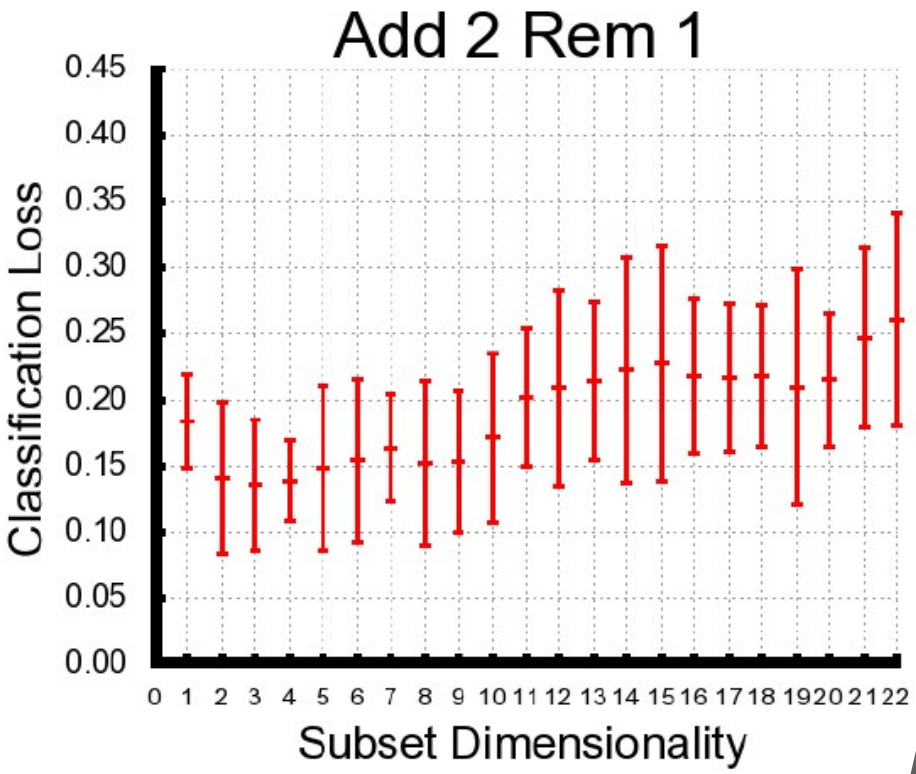
Results from H-L. Wei and S. Billings, IEEE Trans. Pattern Analysis and Machine Intel. 29 (1), 162-166 (2007).

COLIC DATASET

368 events

Classifier: Tree with 15 minimum instances per node.

Add n Rem r selects less variables achieving a comparable accuracy.



Method	accuracy (%)	No of sel. feat.
All	85.85 ± 5.25	
Add2Rem1	85.85 ± 5.25	2
Add1Rem0	85.85 ± 5.25	2
Correlations	82.05 ± 5.44	2
Permutations	78.59 ± 7.33	2
FOM Importance	61.36 ± 4.21	2
ID3	81.52 ± 2.0	17.4
ID3 HC-FSS	83.15 ± 1.1	2.8
ID3 BFS-FSS Forward	82.07 ± 1.5	3.4
ID3 BFS-FSS Backward	82.61 ± 1.7	7.2
C4.5 GA-Wrapper	82.4	13
C4.5 ReliefF-GA-Wrapper	83.8	10
C4.5 ReliefF	85.3	20
C4.5 All	85.3	22
C4.5-RelFss	85.9	4
C4.5-RelFss	84.5	12

D. Bell and H. Wang, ML 41, 175-195 (2000)

L-X. Zhang et al., Proc. Of the Second Intern. Conf. On ML and Cybernetics, Xi'an, (2003).

R. Kohavi and G. John, Artificial Intel. Jour, 97, 273-324 (1997).

20 features out of 22 not needed

CPU TIME

Dataset	No of instances	No of Classes	No of Sel. features/ Tot. features	Program	Method	Time
Magic	19020	2	6/10	SPR	Add 2 Rem 1	$238m$
					Add 1 Rem 0	$97m\,20s$
					Correlations	$2.53s$
					Permutations	$2m\,58s$
					FOM Importance	$2m\,05s$
Arrhythmia	420	12	11/261	SPR	Add 2 Rem 1	$44m\,30s$
					Add 1 Rem 0	$19m10s$
			5/261	Weka	FCBF	$2.11s$
			25/261		ReliefF	$9.01s$
			25/261		CFS-SF	$2.65s$
WDBC	569	2	3/30	SPR	Add 2 Rem 1	$7m\,52s$
					Add 1 Rem 0	$4m\,29s$
					Correlations	$0.250s$
					Permutations	$0.637s$
					FOM Importance	$1.02s$
				R	Rulefit	$2.81s$
WBC	699	2	3/9	SPR	Add 2 Rem 1	$12.7s$
					Add 1 Rem 0	$4.90s$
					Correlations	$0.078s$
					Permutations	$2.04s$
				R	Rulefit	$2.24s$
Colic	368	2	2/22	SPR	Add 2 Rem 1	$2.78s$
					Add 1 Rem 0	$1.89s$
					Correlations	$0.142s$
					Permutations	$0.171s$
					FOM Importance	$0.071s$

If used in conjunction with huge datasets and time consuming classifiers, wrapper methods are much slower. Add 2 Rem 1 has always the best performance and correlations, when used, is always the fastest one.

SCALING CPU-TIME

SCALING CPU-TIME FOR TREES-BASED CLASSIFIERS

Time to train a tree

Size (N) vs Time (t) $N*\log(N) : N_1*\log(N_1) = t : t_1$

Dimensionality (D) vs Time (t) $D : D_1 = t : t_1$

Datasets >1000 events = no K-fold cross-validation needed $t_1 = t/K$

Bagging: Time needed to train one tree times number of trees in the ensemble

Boosting: A bit slower then bagging. Bagging + time needed to reweight the training data

Random Forest: Faster than bagging. Like bagging, but lower dimensionality

Wrapper Methods: The number of combinations evaluated depends on the method and when the FOM stops improving. For Add 1 Rem 0 all the possible variable combinations would be D*(D+1)/2 – 1.

Magic is a typical physics training set (10 features/20K data). We used a RF with many cycles (200). CPU-Time of FS methods for Magic can be considered as a benchmark for physics analysis.

CONCLUSIONS

- **We have compared 4 different FS methods to public datasets showing that comparable discriminating power can be achieved using a subset of features.**
- **From our analysis, the size of the reduced subset is often extremely small compared to the full set.**
- **Among included methods, Add N Rem R algorithm always selects the most powerful subset of variables.**
- **CPU-time for Add N Remove R is reasonable for datasets with 5k-20k events and 10-30 features.**
- **All other methods can often achieve good results. But their reliability does not seem to be guaranteed for every kind of data set.**

BACK UP

NO UNIVERSALLY BETTER CLASSIFIER

	Neural Net	RBF	SVM	Decision Tree	BDT and RF	k-NN
Predictive power	●	●	●	●	●	●
Ability to deal with irrelevant inputs	●	●	●	●	●	●
Interpretability	●	●	●	●	●	●
Curse of dimensionality	●	●	●	●	●	●
Computational scalability with adding new dimensions	●	●	●	●	●	●
Training stability	●	●	●	●	●	○
Response time	●	●	●	●	●	●

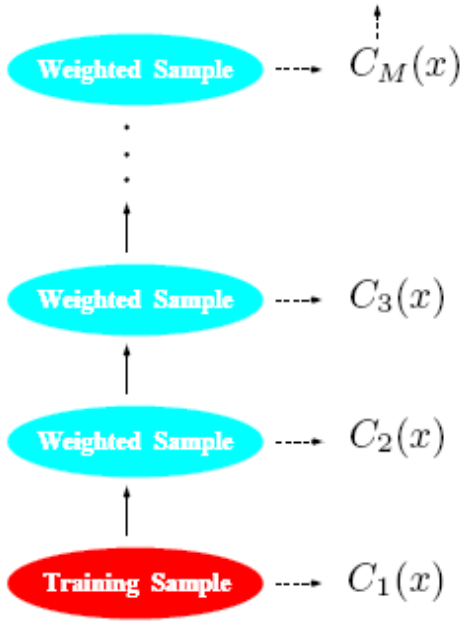
● good ● fair ● poor **Mostly copied from Hastie, Tibshirani & Friedman**

INVEST TIME IN FINDING THE BEST CLASSIFIER AND ITS BEST PARAMETERS FOR THAT SPECIFIC DATASET.

BOOSTING

BOOSTED TREES

- Average many trees, each grown to reweighted versions of the training data.
- Weighting *decorrelates* the trees, by focussing on regions missed by past trees.
- Final Classifier is weighted average of classifiers: $C(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m C_m(x)\right]$



BOOSTED SPLITS

Apply decision splits on input variables sequentially: split 1 on variable 1, split 2 on variable 2 etc; in the end goes back to variable 1 and starts over.

Boosted splits are robust and incredibly fast. However, typically give a worse model error than boosted trees

BAGGING & RANDOM FOREST

BAGGING

Bootstrapping of training points:
Draw N points out of sample of size N => one bootstrap replica

Build many decision trees on bootstrap replicas of the training sample and classify new data by the majority vote

RANDOM FOREST = bagging + random selection of input variables for each decision split