CS6510
Applied Machine Learning

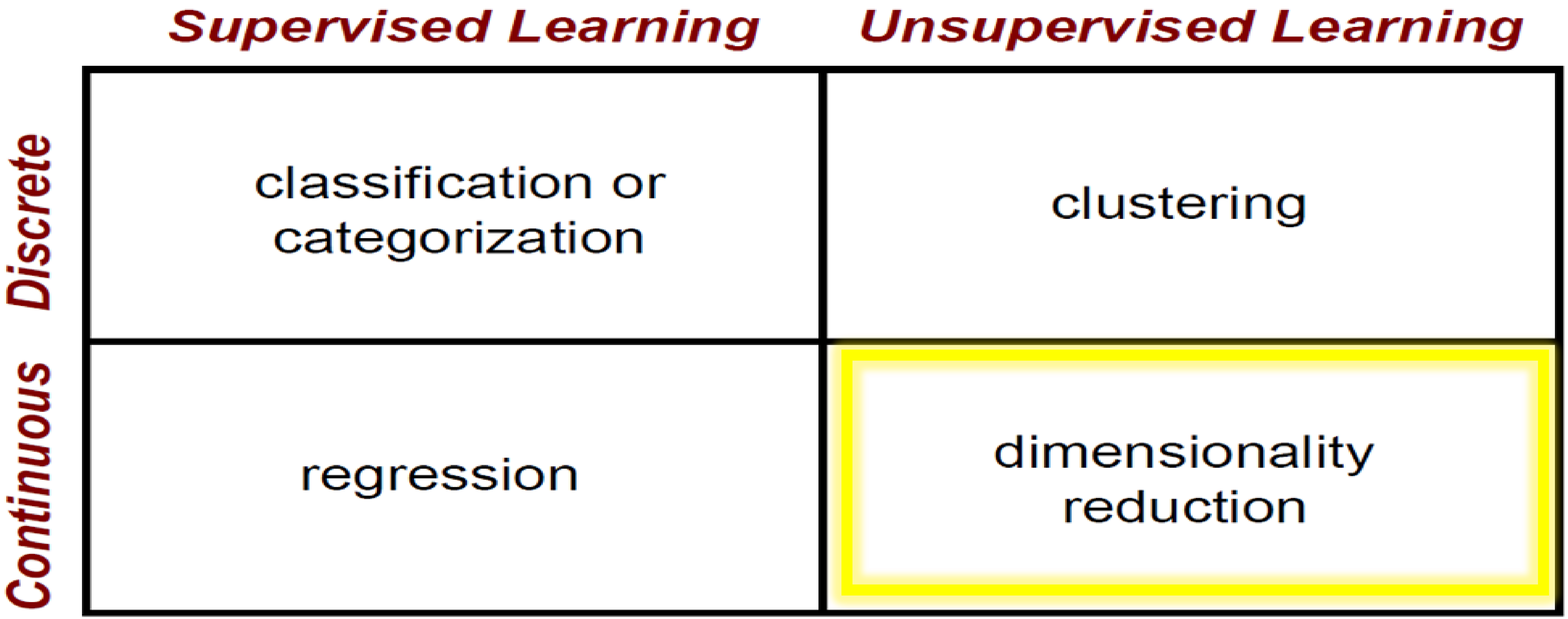
Feature Selection

12 Nov 2016

Vineeth N Balasubramanian

आई आई टी हैदराबाद
IIT Hyderabad

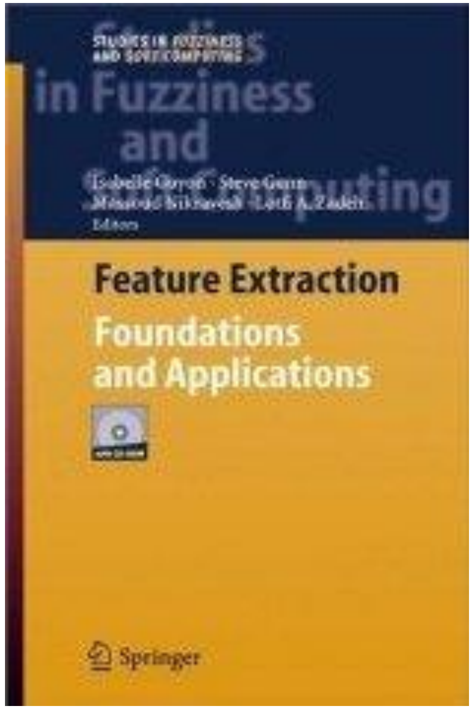
ML Problems



	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Acknowledgement

- Grateful to Isabelle Guyon for slides
  - http://clopinet.com/isabelle/Projects/Vilanova/



Studies in Fuzziness and Soft Computing

Isabelle Guyon · Steve Gunn · Masoud Nikravesh · Lotfi A. Zadeh
Editors

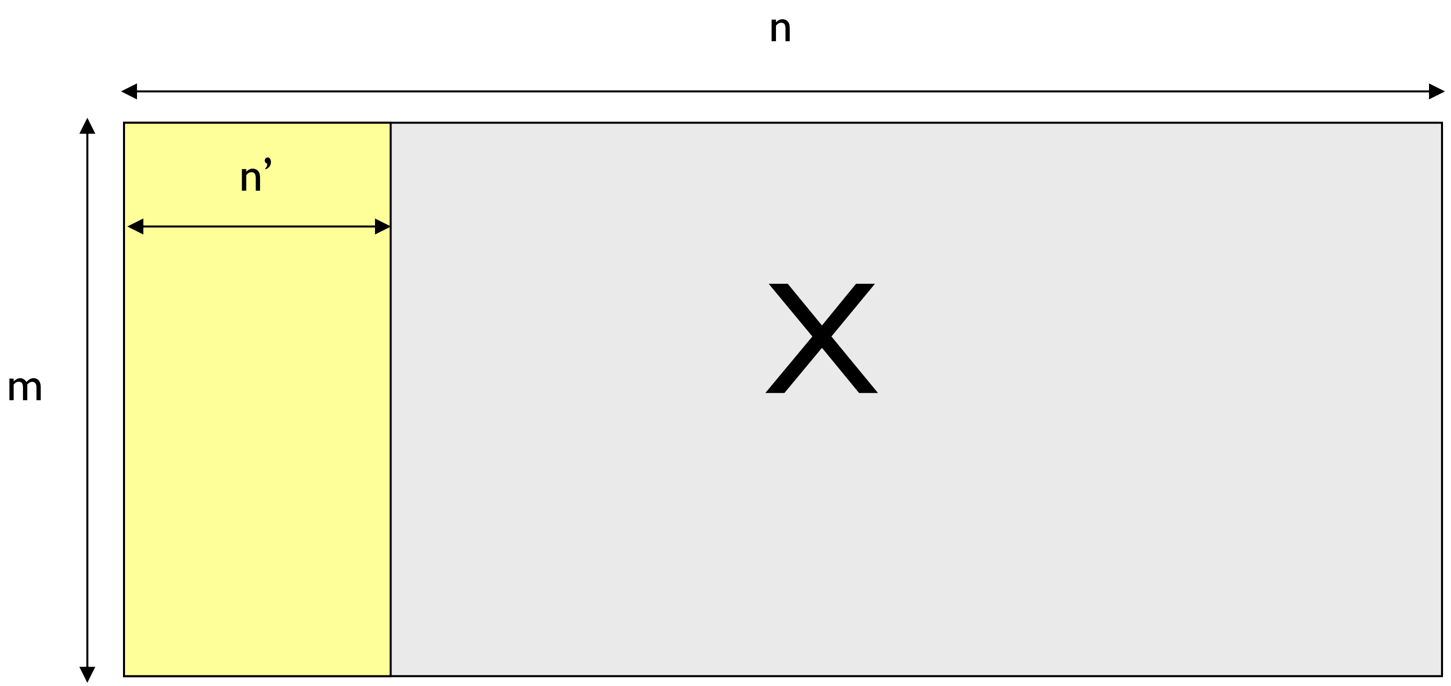
Feature Extraction
Foundations and Applications

Springer

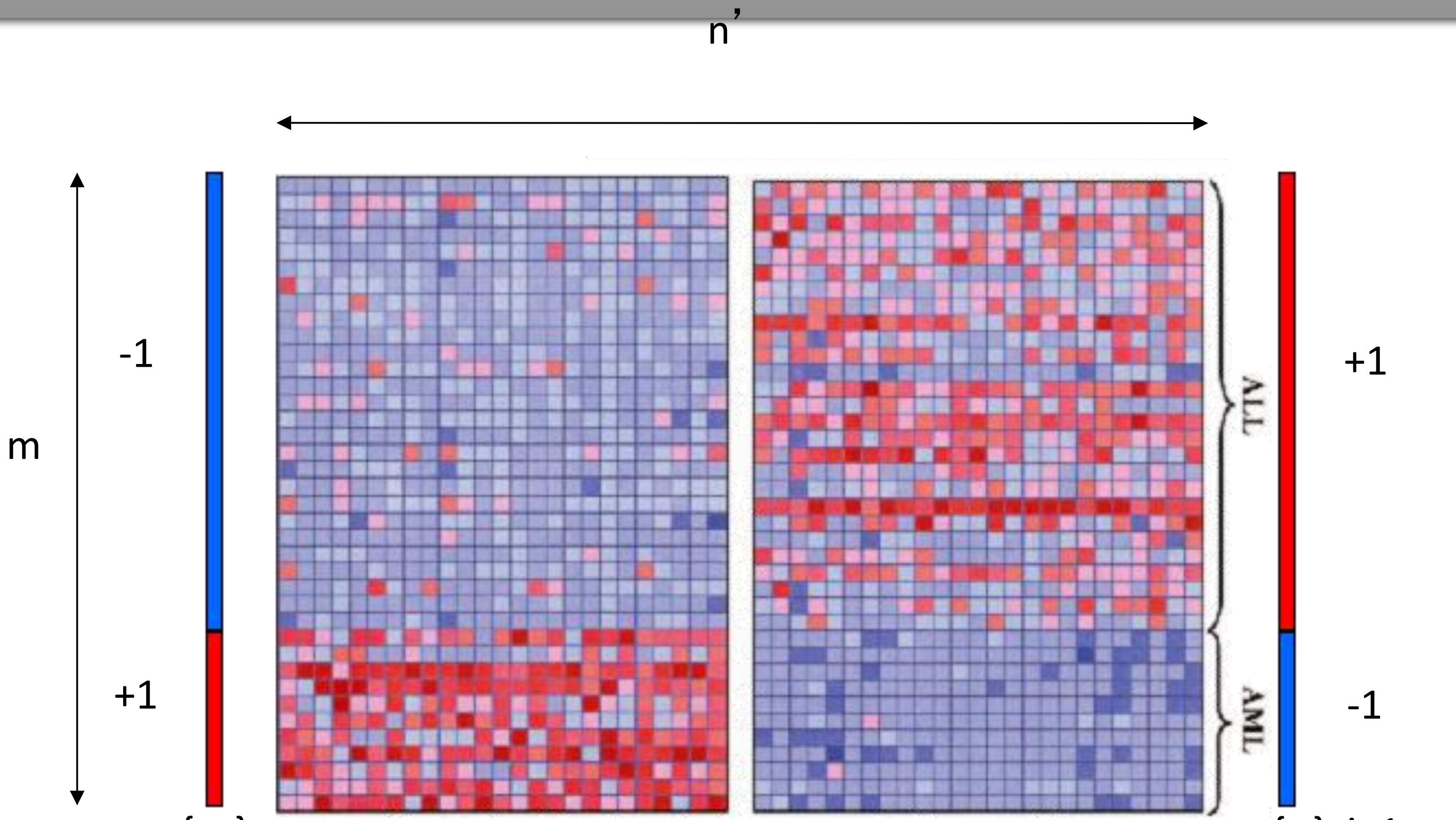
Feature Extraction,
Foundations and Applications
I. Guyon et al, Eds.
Springer, 2006.
http://clopinet.com/fextract-book

Introduction to Feature Selection

- Thousands to millions of low level features: select the most relevant one to build better, faster, and easier to understand learning machines.

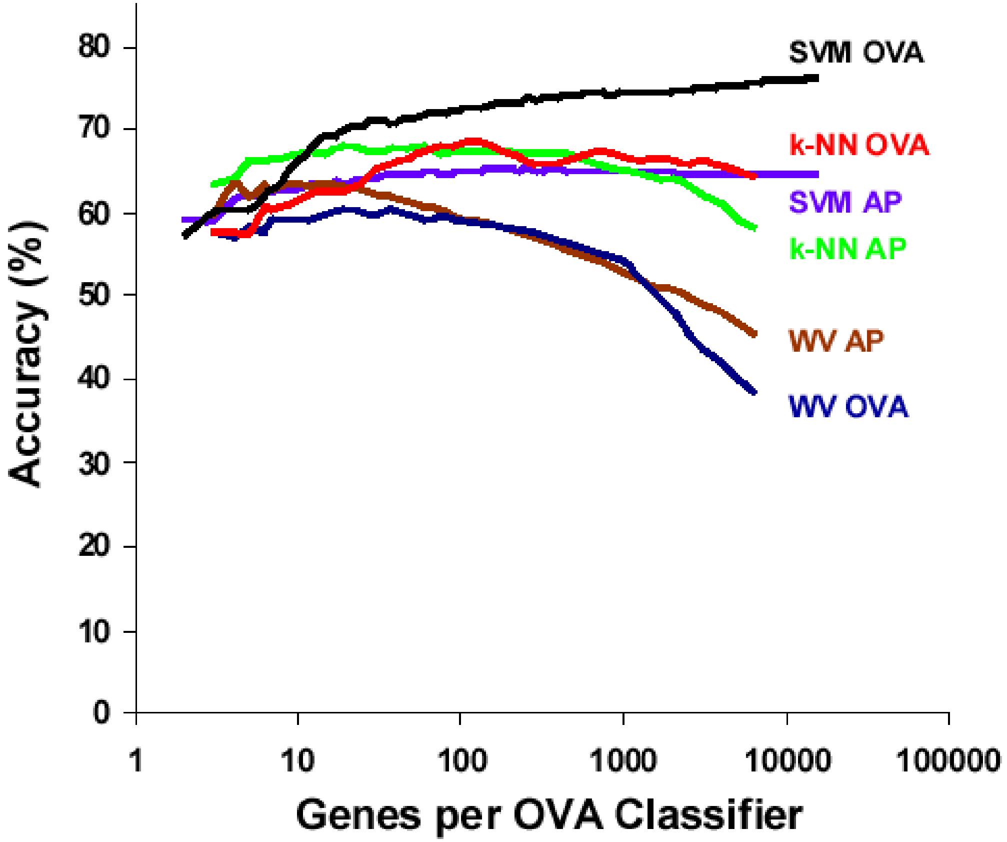


Example: Leukemia Diagnosis



$\{-y_i\}$ *Golub et al, Science Vol 286:15 Oct. 1999* $\{y_i\}$, i=1:m

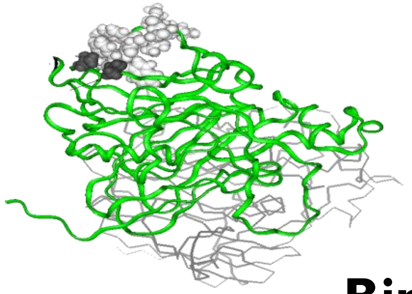
Example: Cancer Diagnosis



Differentiation of 14 tumors.

Ramaswamy et al, PNAS, 2001

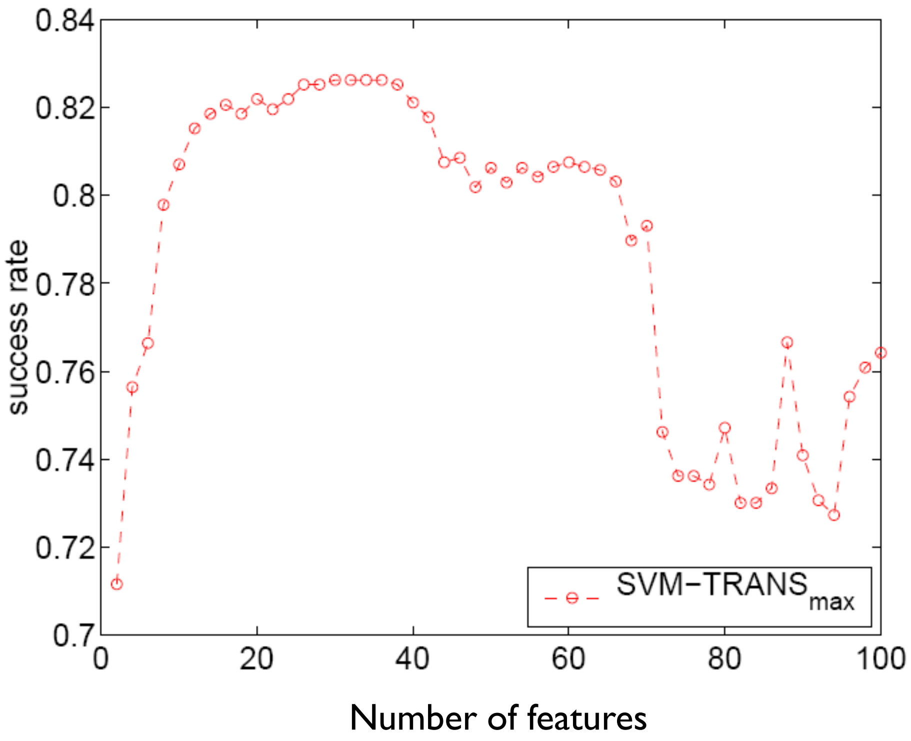
Example: Drug Screening



Binding to Thrombin (DuPont Pharmaceuticals)

- 2543 compounds tested for their ability to bind to a target site on thrombin, a key receptor in blood clotting; 192 "active" (bind well); the rest "inactive". Training set (1909 compounds) more depleted in active compounds.

- 139,351 binary features, which describe three-dimensional properties of the molecule.



Weston et al, Bioinformatics, 2002

Example: Text Filtering

Reuters: 21578 news wire, 114 semantic categories.

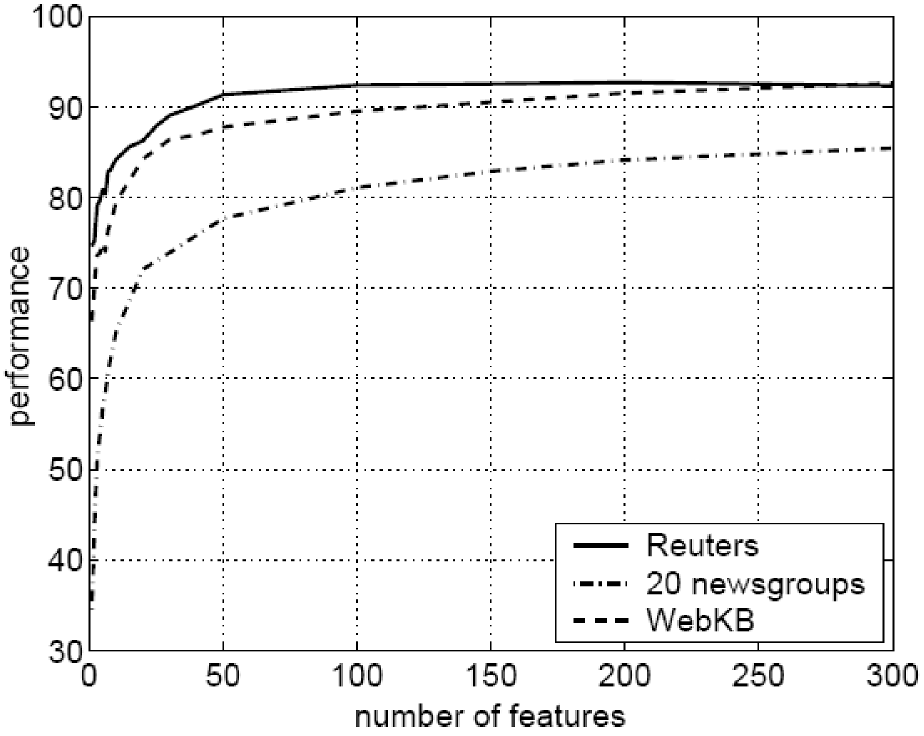
20 newsgroups: 19997 articles, 20 categories.

WebKB: 8282 web pages, 7 categories.

Bag-of-words: >100000 features.

Top 3 words of some categories:

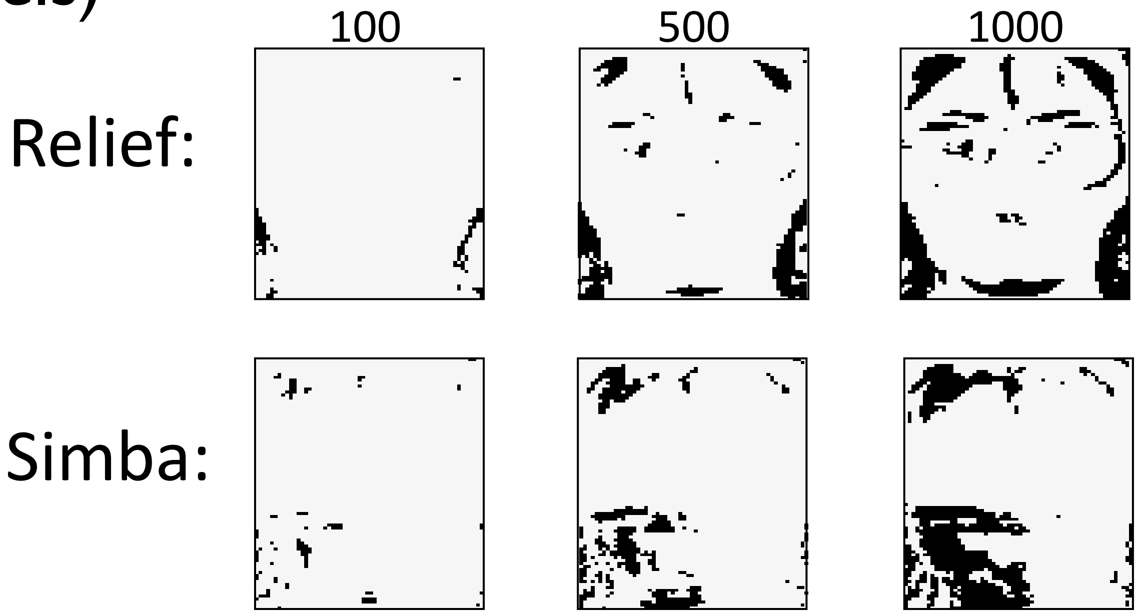
- **Alt.atheism**: atheism, atheists, morality
- **Comp.graphics**: image, jpeg, graphics
- **Sci.space**: space, nasa, orbit
- **Soc.religion.christian**: god, church, sin
- **Talk.politics.mideast**: israel, armenian, turkish
- **Talk.religion.misc**: jesus, god, jehovah



Bekkerman et al, JMLR, 2003

Example: Face Recognition

- Male/female classification
- 1450 images (1000 train, 450 test), 5100 features (images 60x85 pixels)



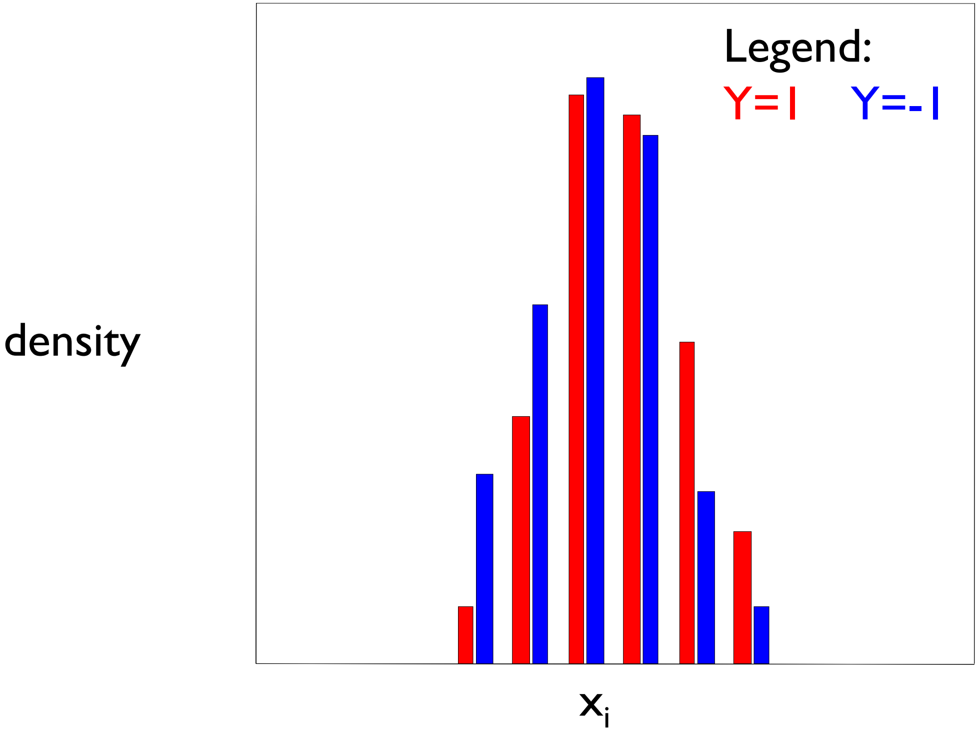
Navot-Bachrach-Tishby, ICML 2004

Categorization of Methods

- **Univariate method**: considers one variable (feature) at a time.
- **Multivariate method:** considers subsets of variables (features) together.
- **Filter method:** ranks features or feature subsets independently of the predictor (classifier).
- **Wrapper method:** uses a classifier to assess features or feature subsets.

Univariate Filter Methods

- Individual Feature Irrelevance



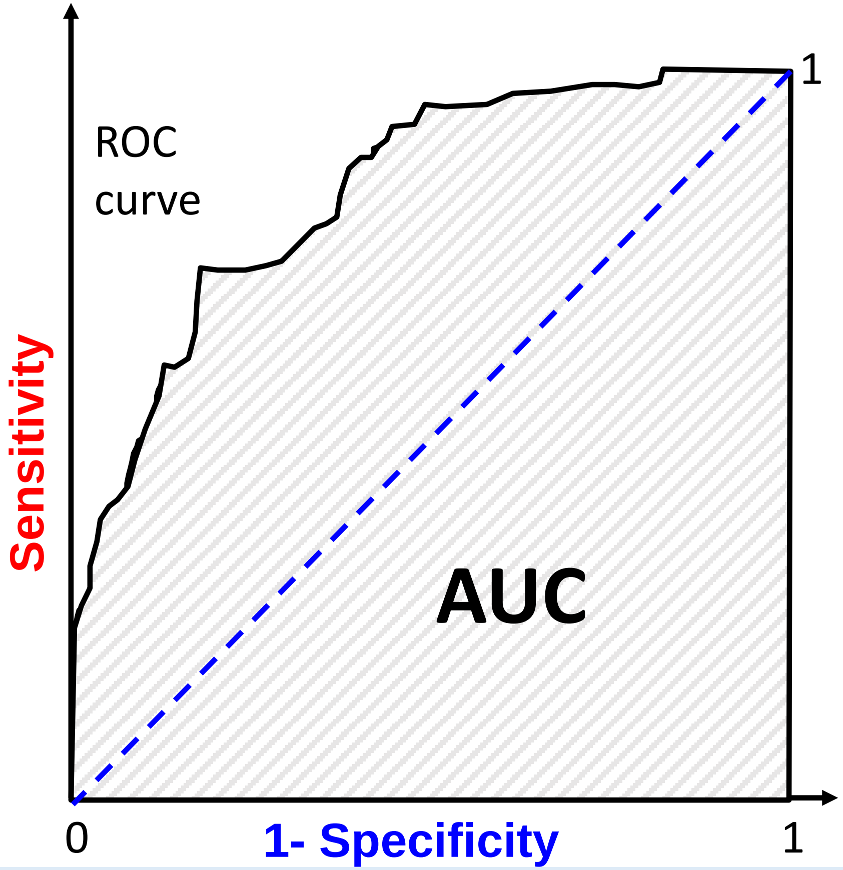
$P(X_i,Y) = P(X_i)\ P(Y)$

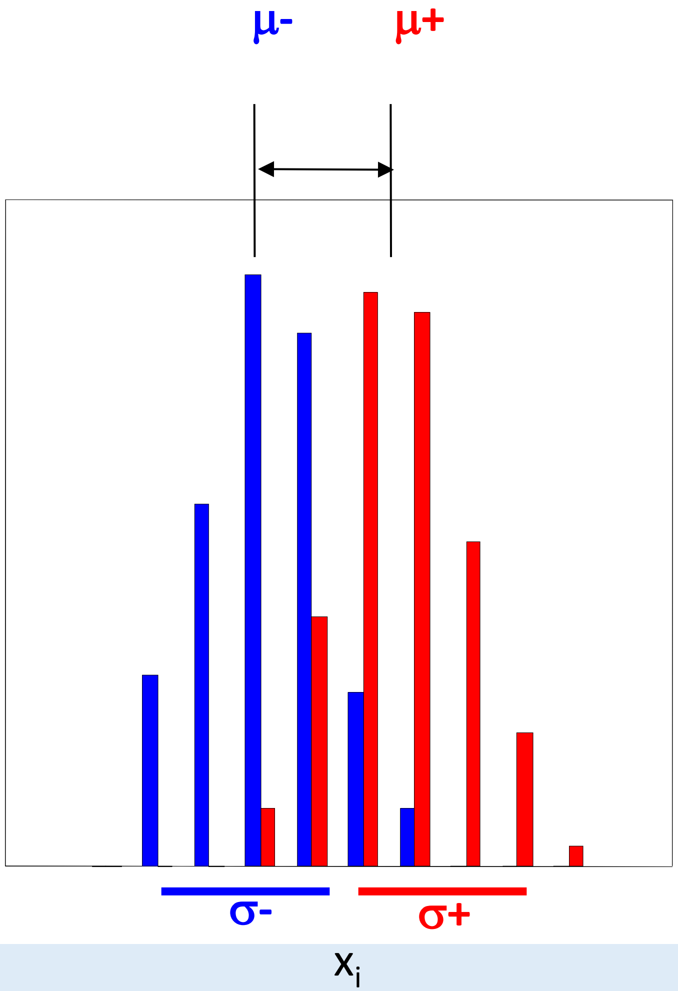
$P(X_i|\,Y) = P(X_i)$

$P(X_i|\,Y=1) = P(X_i|\,Y=-1)$

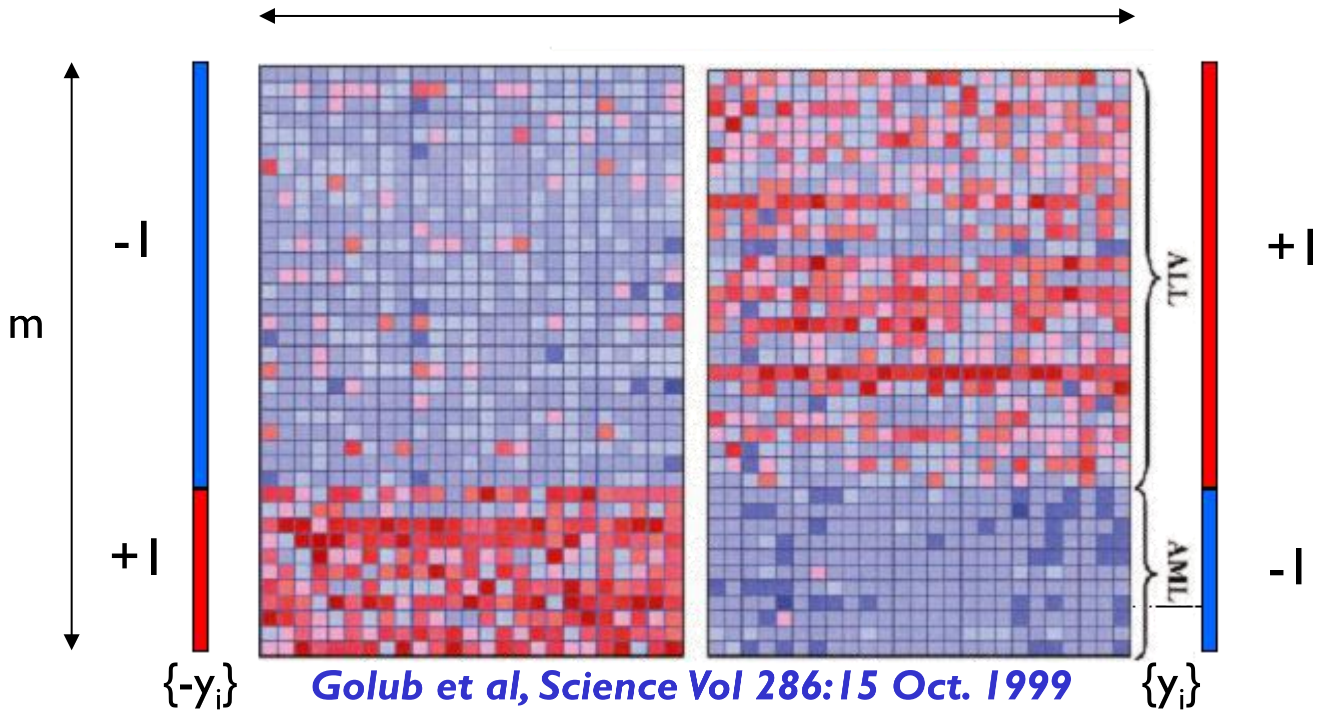
Univariate Filter Methods

- Individual Feature Irrelevance

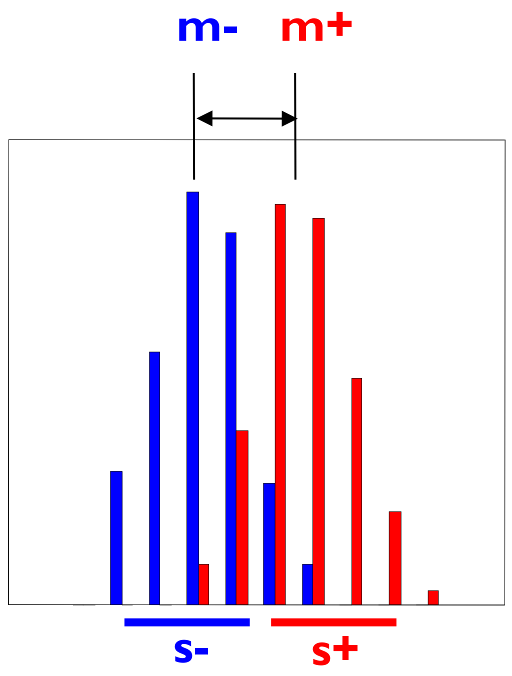




S2N



Golub et al, Science Vol 286:15 Oct. 1999



$$S2N = \frac{|m+ - m-|}{s+ + s-}$$

$S2N \cong R \sim \mathbf{x} \bullet \mathbf{y}$

after "standardization" $\mathbf{x} \leftarrow (\mathbf{x}-m_x)/s_x$

Univariate Dependence

- Independence:

$$P(X,Y) = P(X)\,P(Y)$$

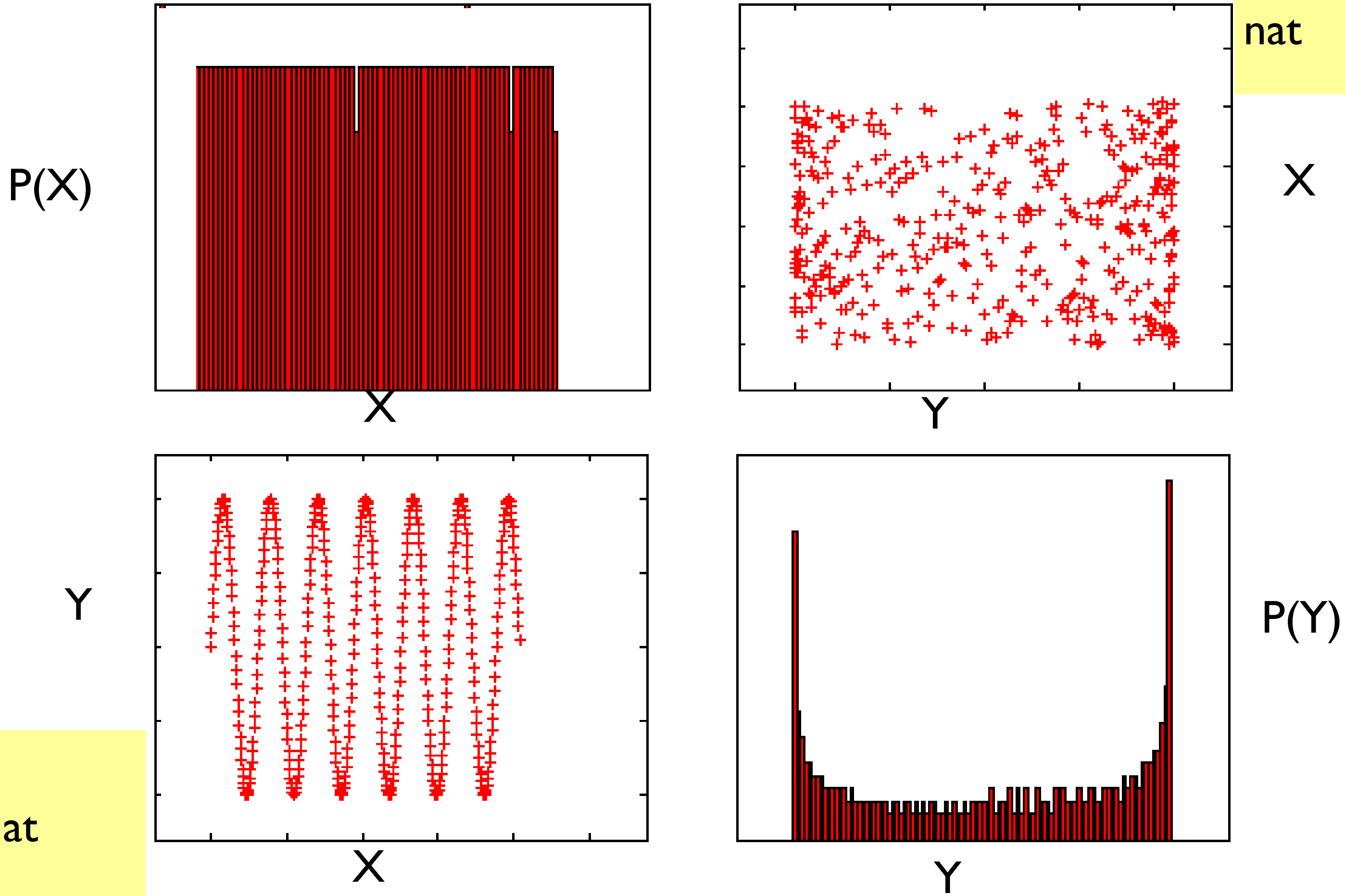
- Measure of dependence:

$$MI(X,Y) = \int P(X,Y) \log \frac{P(X,Y)}{P(X)P(Y)}\, dX\, dY$$

$$= KL\big(\, P(X,Y) \,||\, P(X)P(Y) \,\big)$$

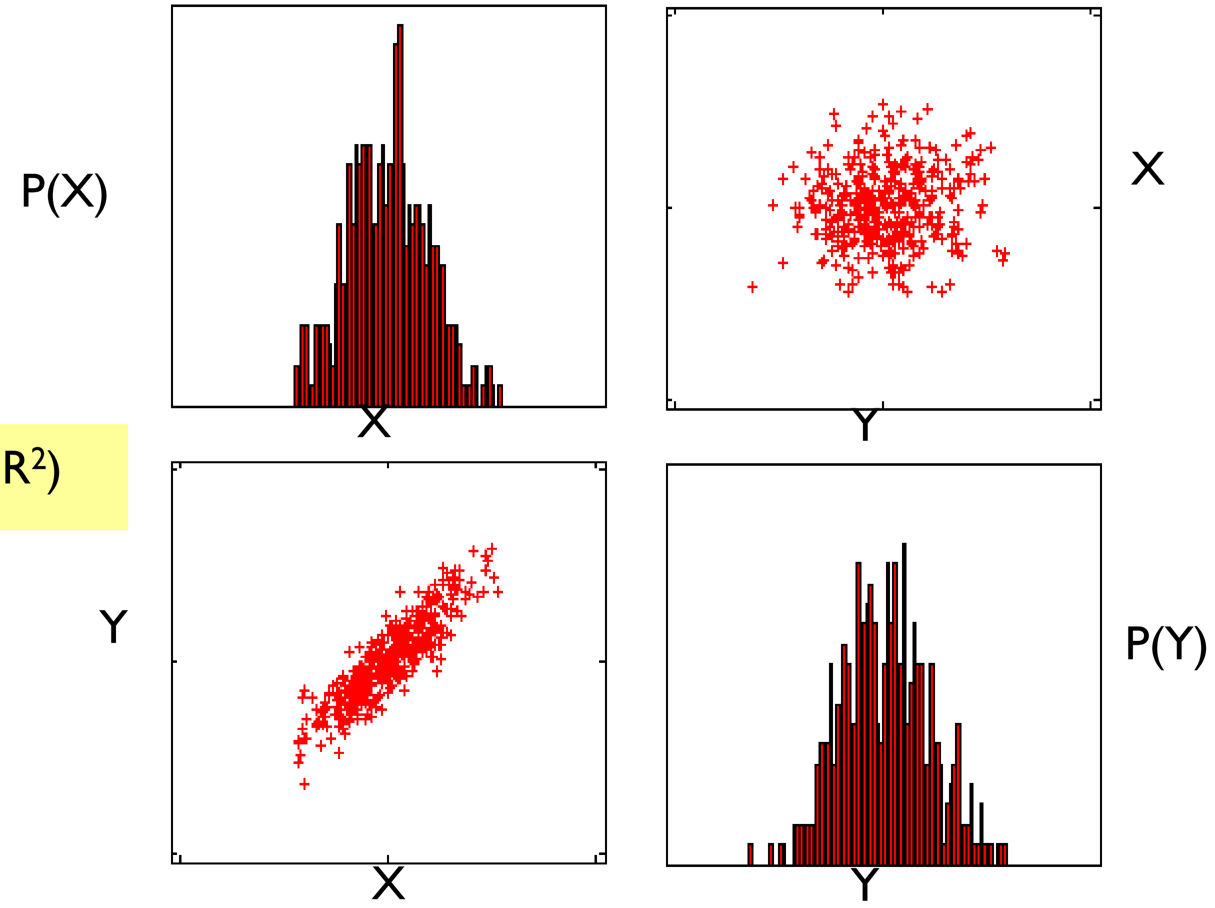
Correlation and Mutual Information

R=0.02 MI=1.03 nat



R=0.00$_{02}$
MI=1.65 nat

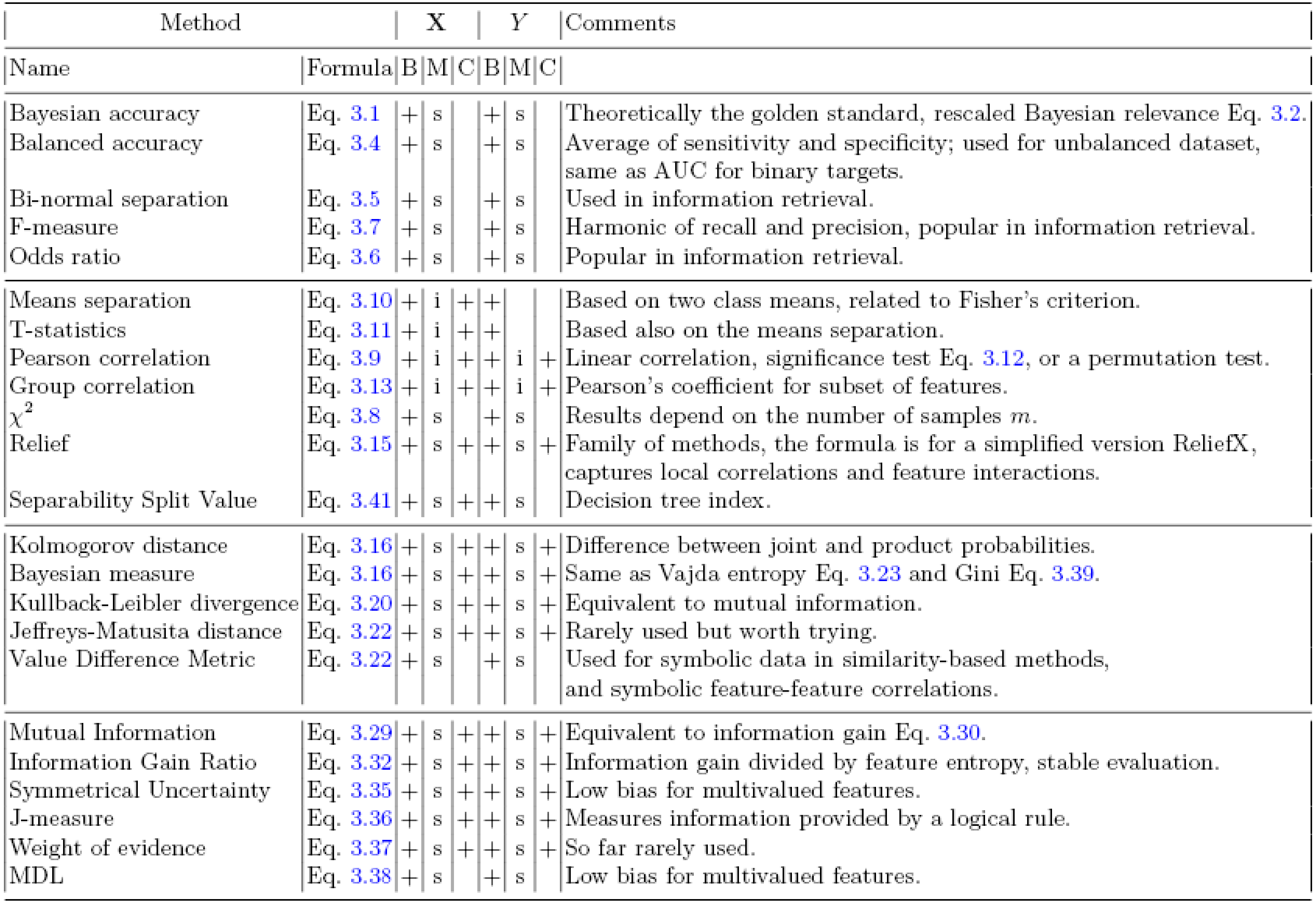
Gaussian Distribution



$MI(X,Y) = -(1/2)\ \log(1-R^2)$

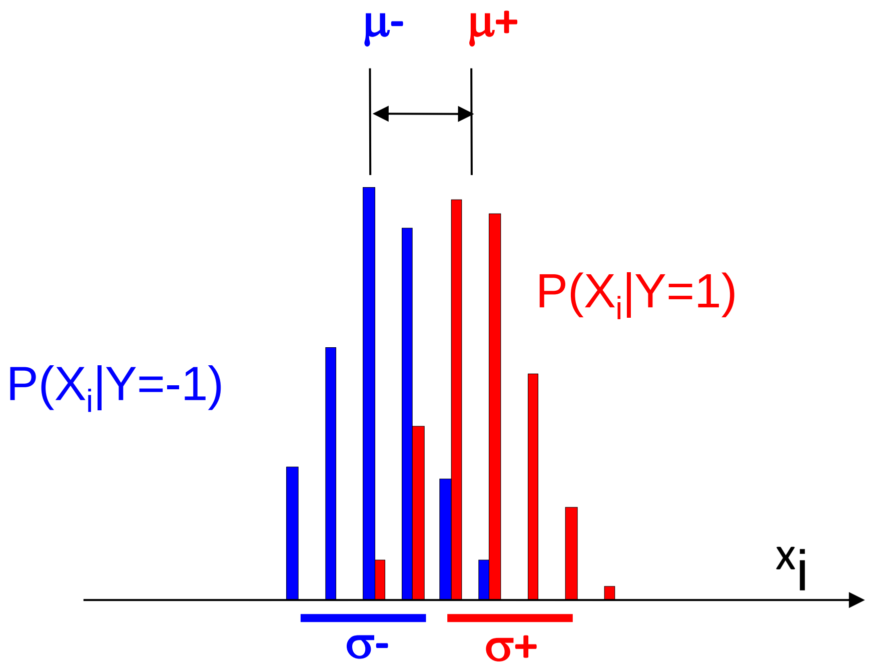
Other Criteria

(Guyon book, Chap 3)



Method		**X**			*Y*			Comments
Name	Formula	B	M	C	B	M	C	
Bayesian accuracy	Eq. 3.1	+	s		+	s		Theoretically the golden standard, rescaled Bayesian relevance Eq. 3.2.
Balanced accuracy	Eq. 3.4	+	s		+	s		Average of sensitivity and specificity; used for unbalanced dataset, same as AUC for binary targets.
Bi-normal separation	Eq. 3.5	+	s		+	s		Used in information retrieval.
F-measure	Eq. 3.7	+	s		+	s		Harmonic of recall and precision, popular in information retrieval.
Odds ratio	Eq. 3.6	+	s		+	s		Popular in information retrieval.
Means separation	Eq. 3.10	+	i	+	+			Based on two class means, related to Fisher's criterion.
T-statistics	Eq. 3.11	+	i	+	+			Based also on the means separation.
Pearson correlation	Eq. 3.9	+	i	+	+	i	+	Linear correlation, significance test Eq. 3.12, or a permutation test.
Group correlation	Eq. 3.13	+	i	+	+	i	+	Pearson's coefficient for subset of features.
χ^2	Eq. 3.8	+	s		+	s		Results depend on the number of samples m.
Relief	Eq. 3.15	+	s	+	+	s	+	Family of methods, the formula is for a simplified version ReliefX, captures local correlations and feature interactions.
Separability Split Value	Eq. 3.41	+	s	+	+	s		Decision tree index.
Kolmogorov distance	Eq. 3.16	+	s	+	+	s	+	Difference between joint and product probabilities.
Bayesian measure	Eq. 3.16	+	s	+	+	s	+	Same as Vajda entropy Eq. 3.23 and Gini Eq. 3.39.
Kullback-Leibler divergence	Eq. 3.20	+	s	+	+	s	+	Equivalent to mutual information.
Jeffreys-Matusita distance	Eq. 3.22	+	s	+	+	s	+	Rarely used but worth trying.
Value Difference Metric	Eq. 3.22	+	s		+	s		Used for symbolic data in similarity-based methods, and symbolic feature-feature correlations.
Mutual Information	Eq. 3.29	+	s	+	+	s	+	Equivalent to information gain Eq. 3.30.
Information Gain Ratio	Eq. 3.32	+	s	+	+	s	+	Information gain divided by feature entropy, stable evaluation.
Symmetrical Uncertainty	Eq. 3.35	+	s	+	+	s	+	Low bias for multivalued features.
J-measure	Eq. 3.36	+	s	+	+	s	+	Measures information provided by a logical rule.
Weight of evidence	Eq. 3.37	+	s	+	+	s	+	So far rarely used.
MDL	Eq. 3.38	+	s		+	s		Low bias for multivalued features.

T-test



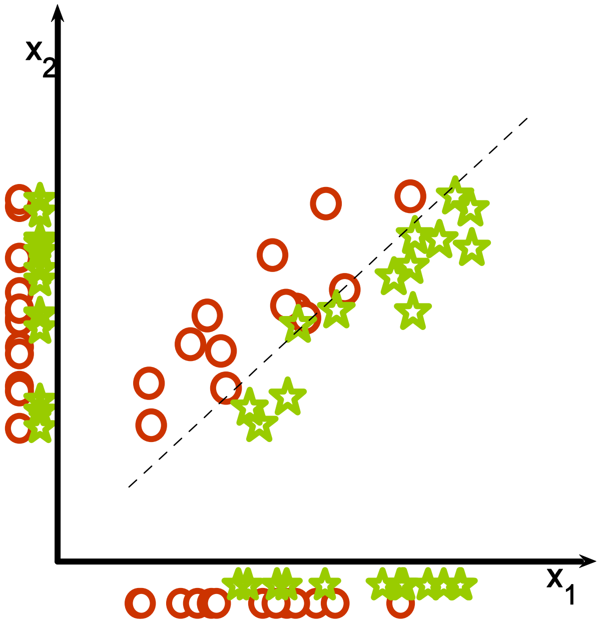
- Normally distributed classes, equal variance s^2 unknown; estimated from data as s^2_{within}.
- Null hypothesis H_0: m+ = m-
- T statistic: If H_0 is true:

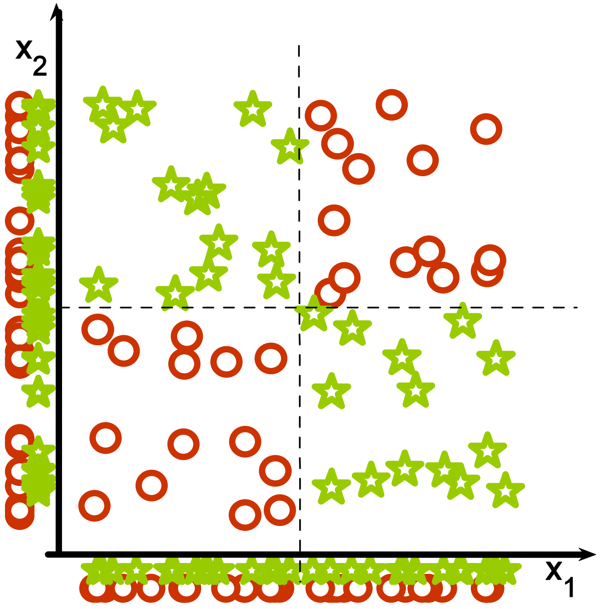
$$t = (m+ - m-)/(\sigma_{within}\sqrt{1/m^{+}+1/m^{-}})$$
$$\sim \text{Student}(m^{+}+m^{-}-2 \text{ d.f.})$$

For more, please see Chapter 2 of Guyon's book

Multivariate Selection

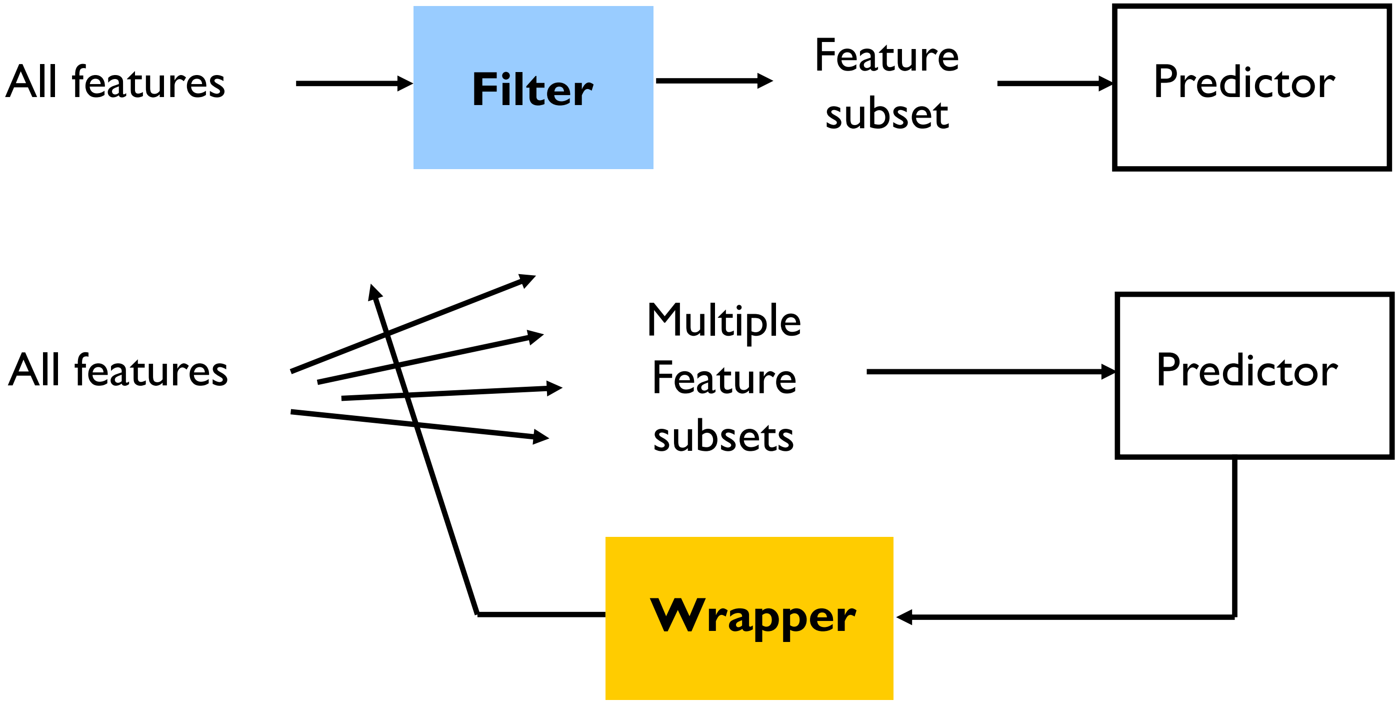
- What's the issue with univariate selection?





Filter vs Wrapper Methods

- **Main goal:** rank subsets of useful features.

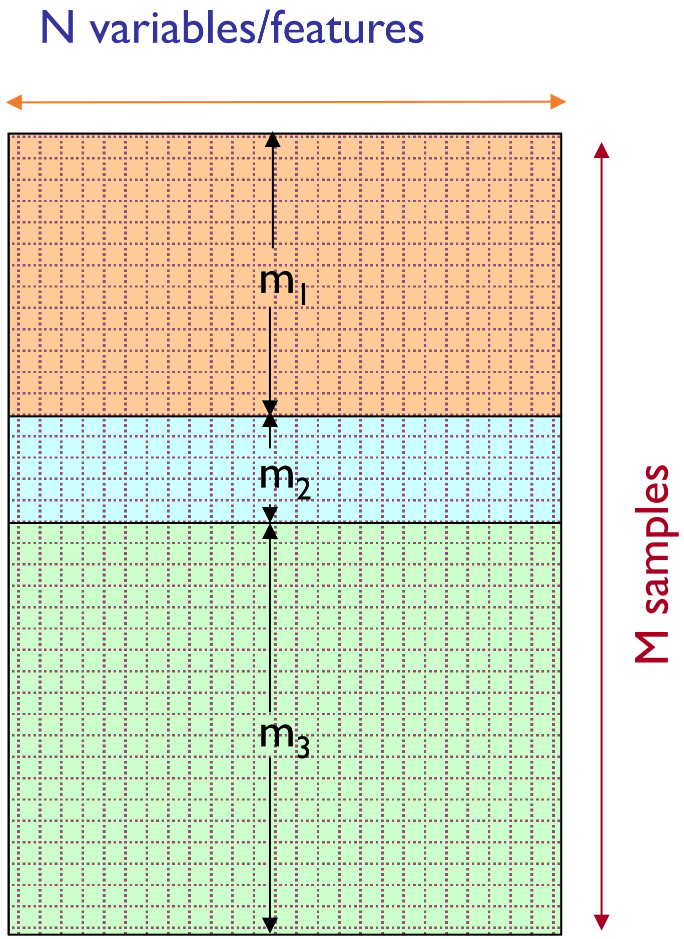


- **Danger of over-fitting** with intensive search!

Search Strategies

- **Forward selection** or **backward elimination.**
- **Beam search:** keep k best path at each step.
- **GSFS:** generalized sequential forward selection – when (n-k) features are left try all subsets of g features i.e. ${}^{n-k}C_g$ trainings. More trainings at each step, but fewer steps.
- **PTA(l,r):** plus l, take away r – at each step, run SFS l times then SBS r times.
- **Floating search** (SFFS and SBFS): One step of SFS (resp. SBS), then SBS (resp. SFS) as long as we find better subsets than those of the same size obtained so far. Any time, if a better subset of the same size was already found, switch abruptly.

Feature Subset Assessment

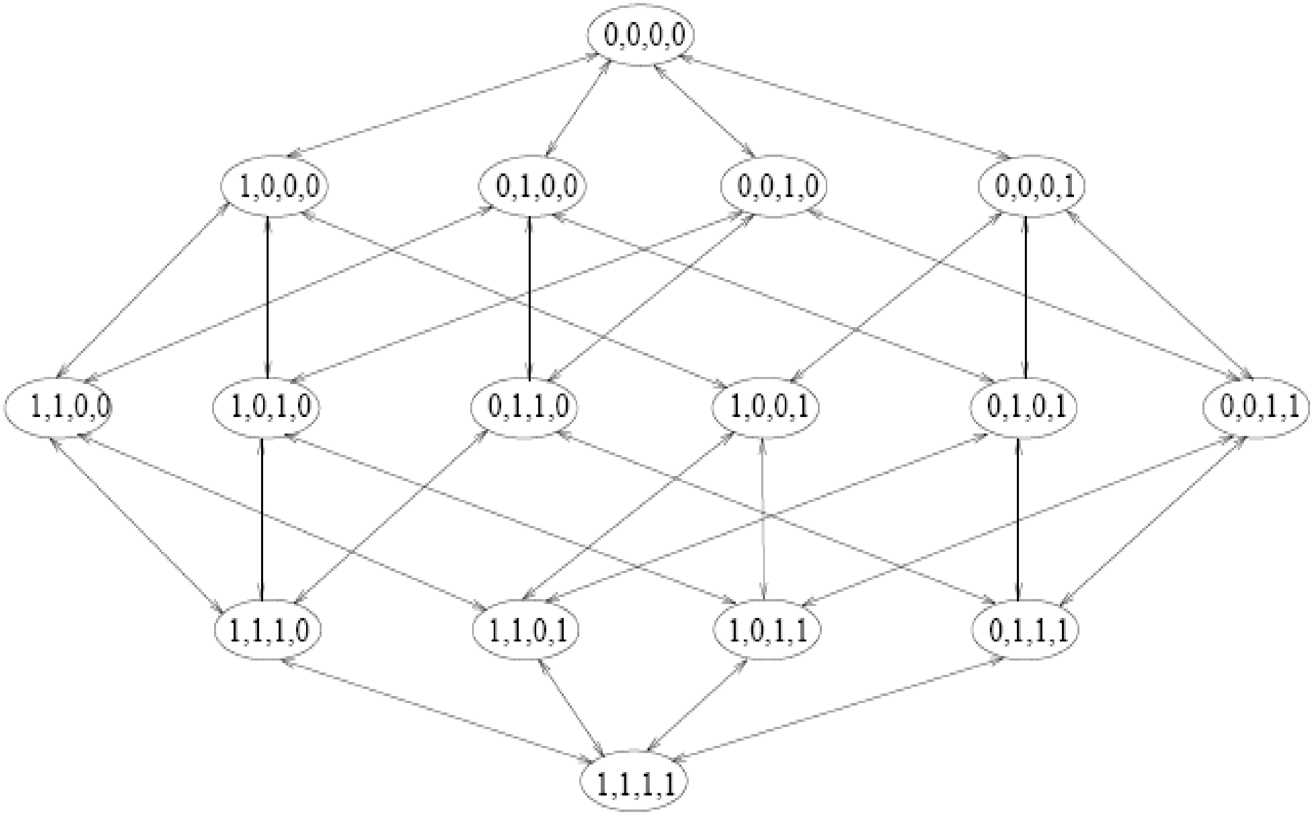


Split data into 3 sets:

training, validation, and test set.

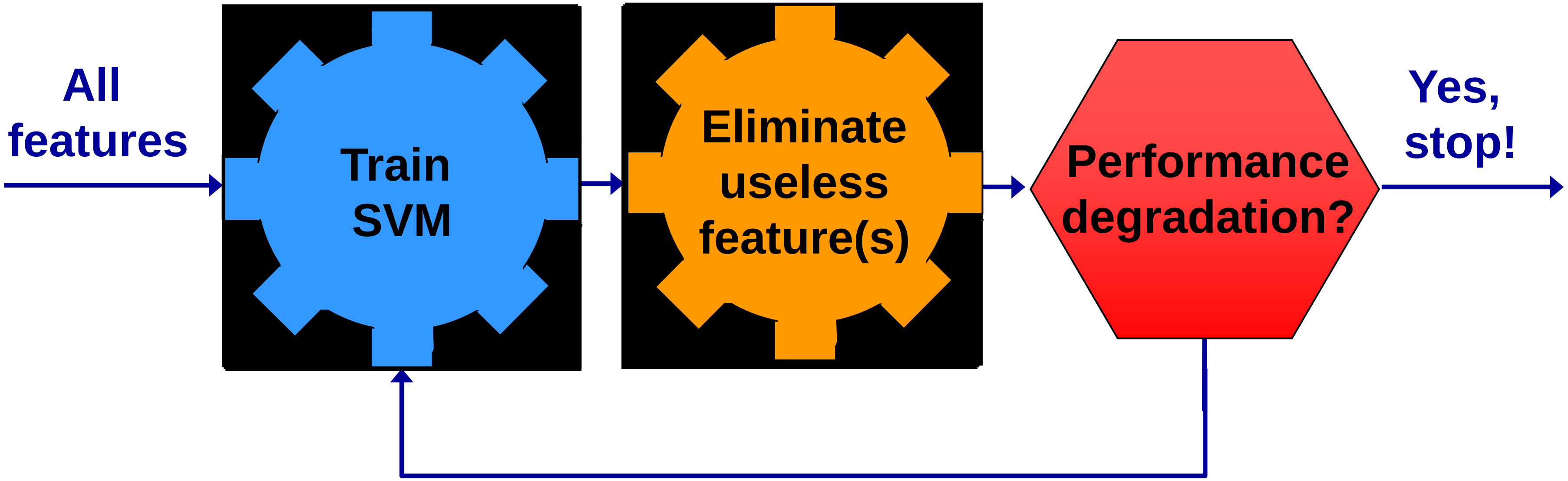
1. For each feature subset, train predictor on training data.
2. Select the feature subset, which performs best on validation data.
   1. Repeat and average if you want to reduce variance (cross-validation).
3. Test on test data.

Multivariate FS is complex!



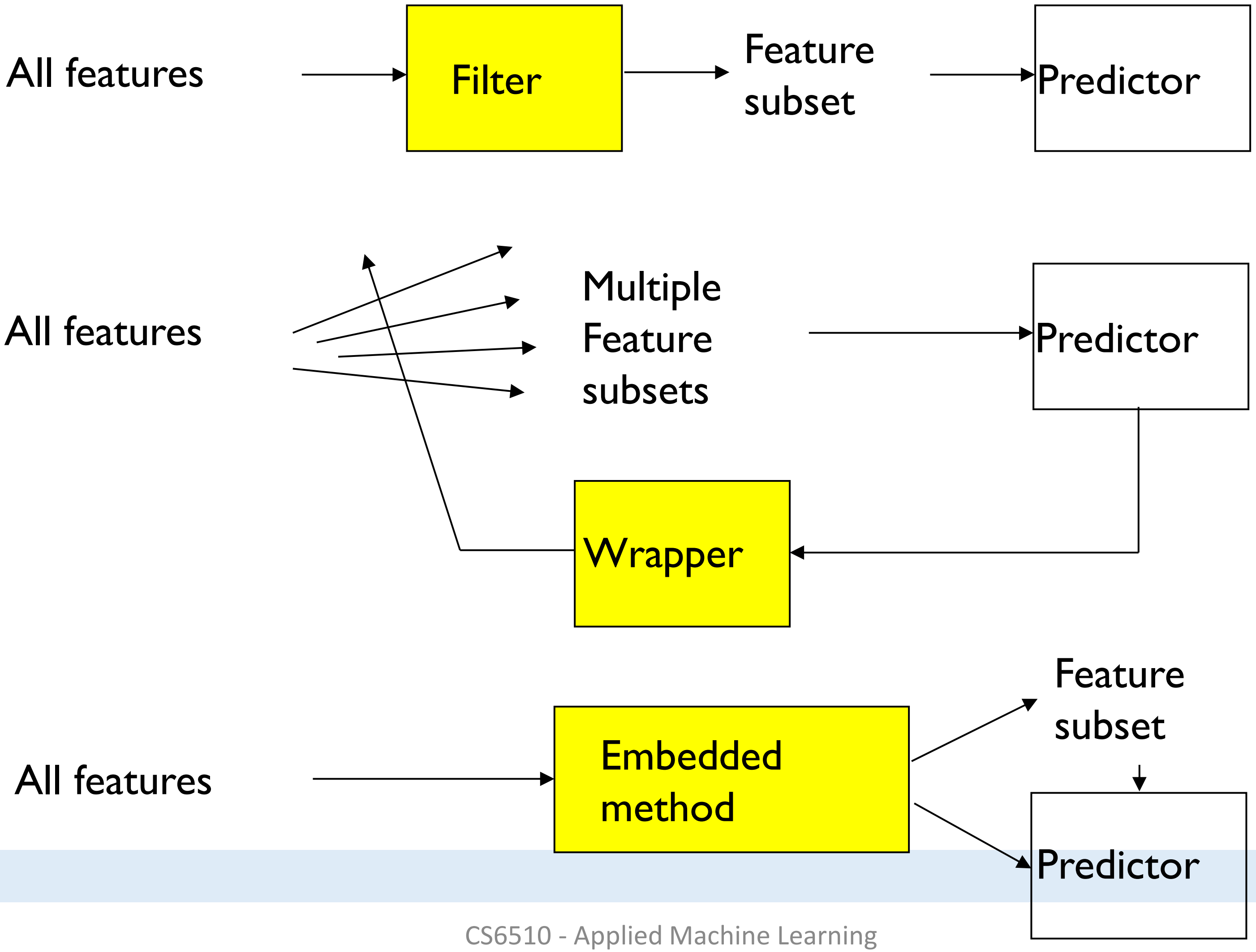
N features, 2^N possible feature subsets!

Embedded Methods



Example: Recursive Feature Elimination (RFE) SVM. *Guyon-Weston, 2000.*

Filters, Wrapper and Embedded Methods



Filters

- Methods
  - Criterion: Measure feature/feature subset "relevance"
  - Search: Usually order features (individual feature ranking or nested subsets of features)
  - Assessment: Use statistical tests
- Results
  - Are (relatively) robust against overfitting
  - May fail to select the most "useful" features

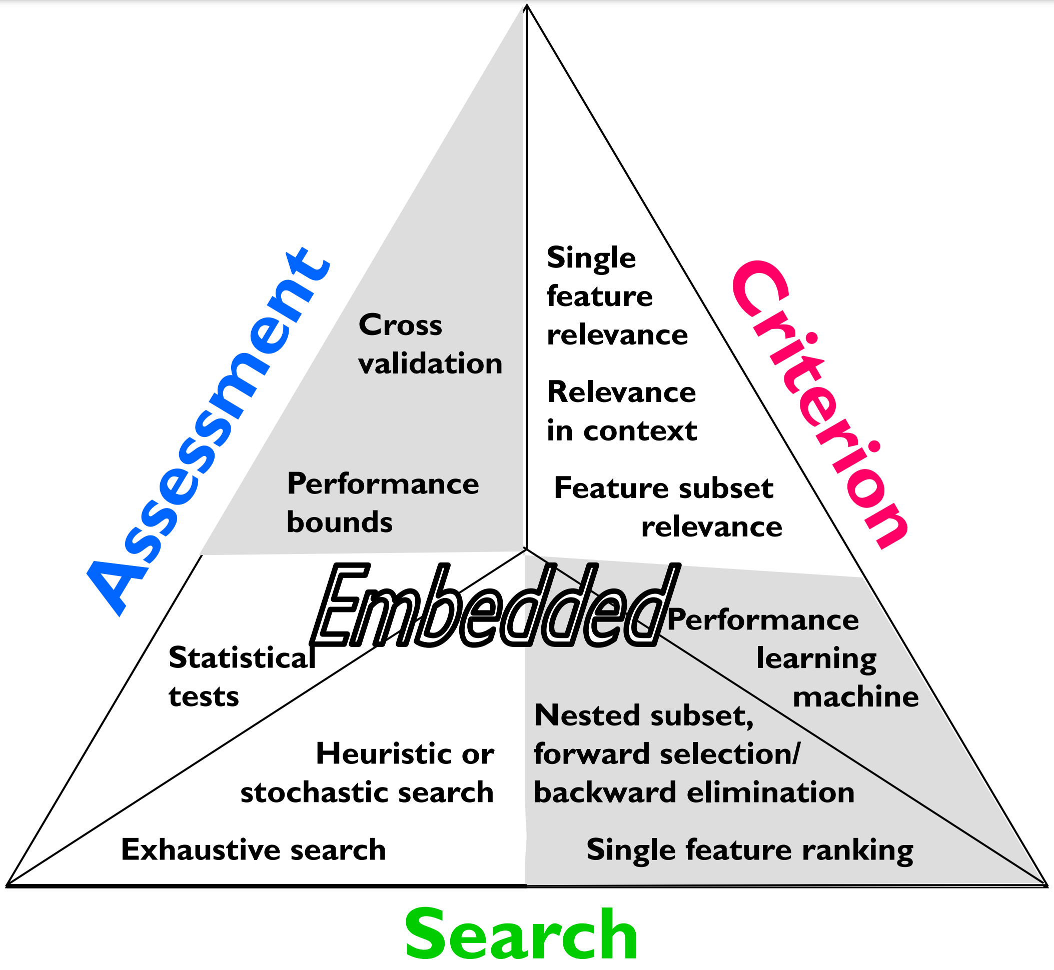
Wrappers

- Methods
  - Criterion: Measure feature subset "usefulness"
  - Search: Search the space of all feature subsets
  - Assessment: Use cross-validation
- Results
  - Can in principle find the most "useful" features, but
  - Are prone to overfitting

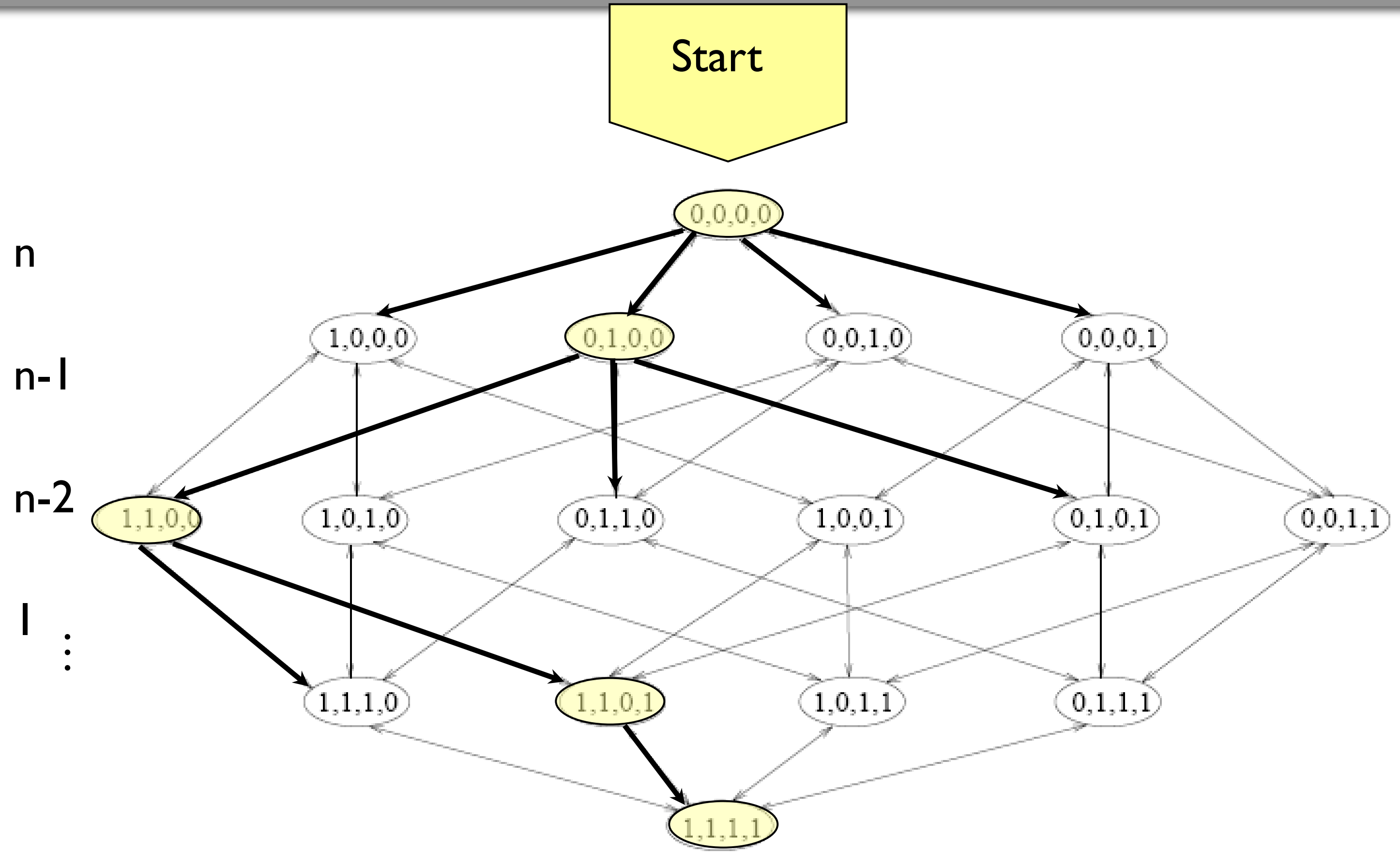
Embedded Methods

- Methods
  - Criterion: Measure feature subset "usefulness"
  - Search: **Search guided by the learning process**
  - Assessment: Use cross-validation
- Results
  - Similar to wrappers, but
  - Less computationally expensive
  - Less prone to overfitting

Three Ingredients



Forward Selection (Wrapper)



Start

n

n-1

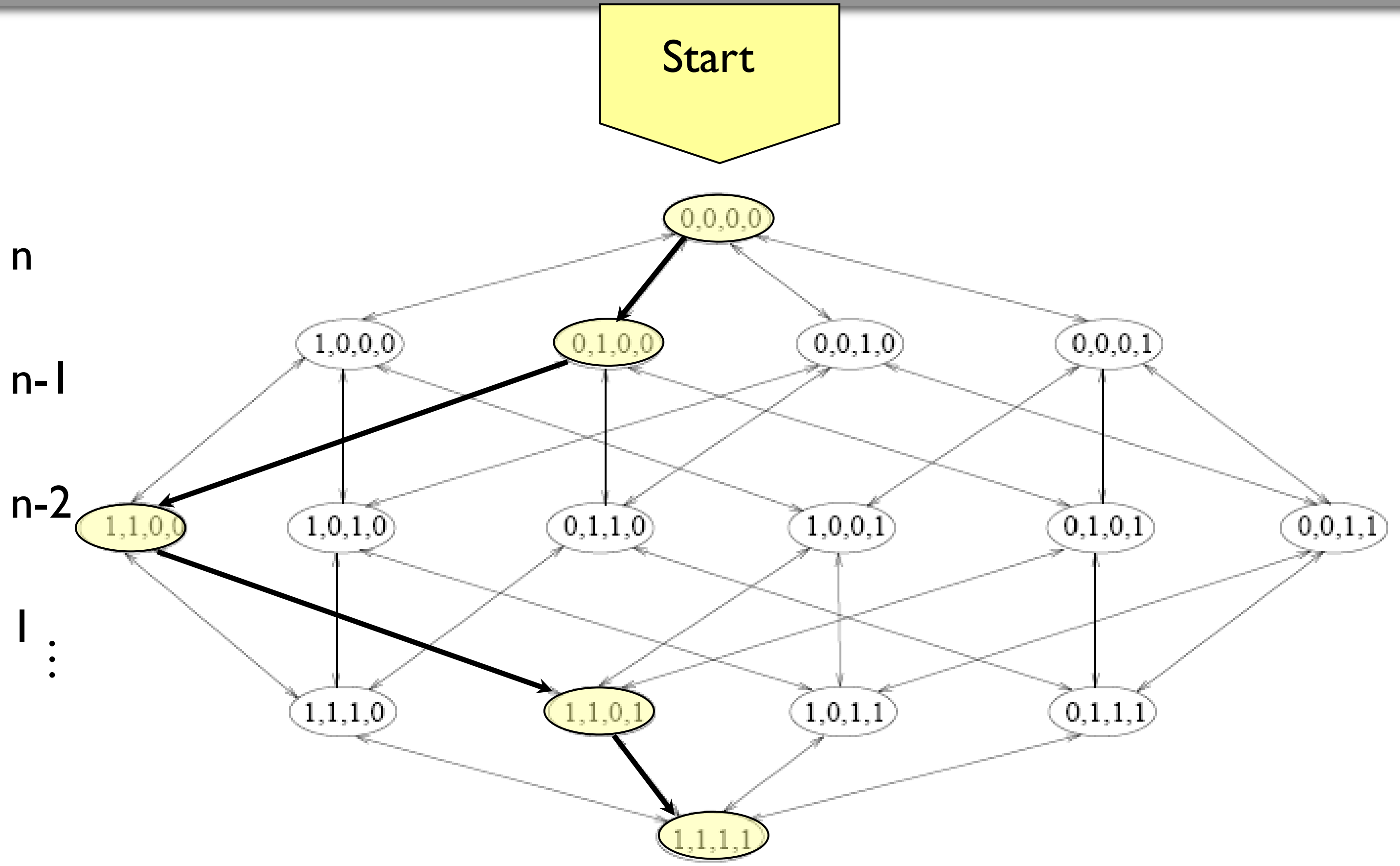
n-2

1

⋮

Also referred to as SFS: Sequential Forward Selection

Forward Selection (Embedded)



Guided search: we do not consider alternative paths.

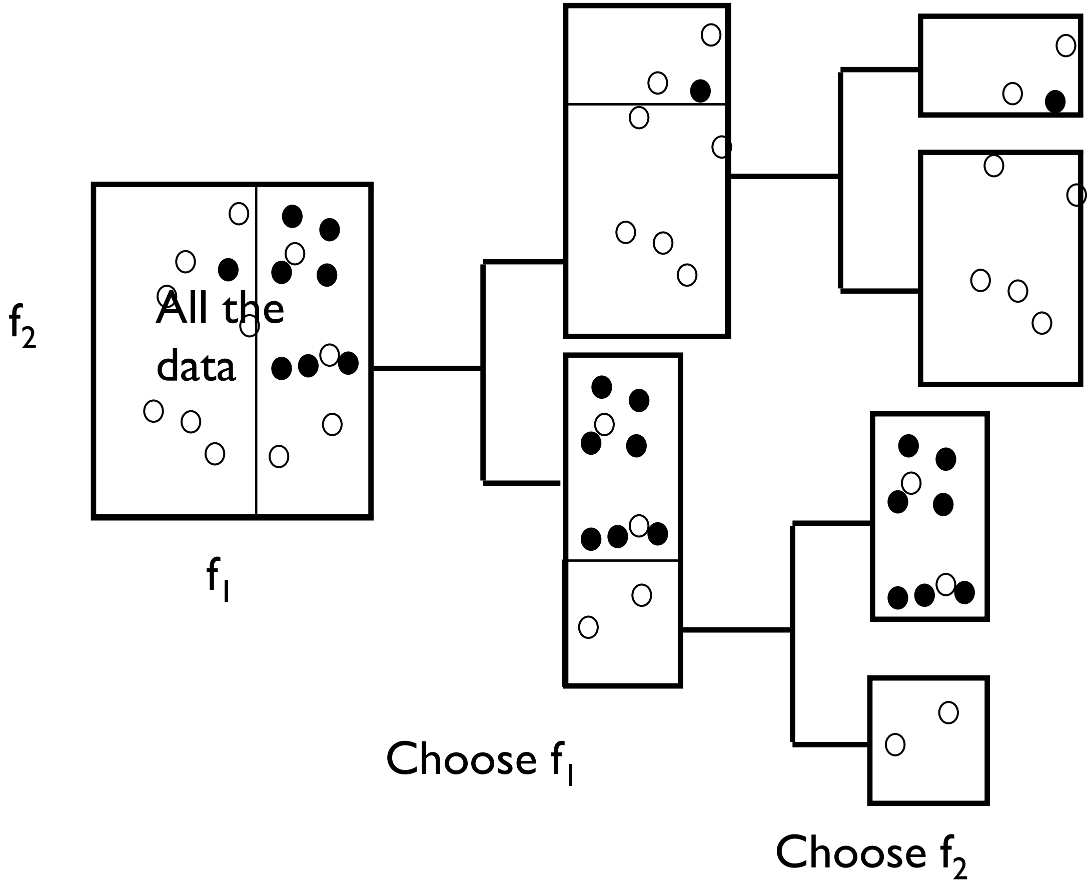
Forward Selection with GS

Stoppiglia, 2002. Gram-Schmidt orthogonalization.

- Select a first feature $X_{\nu(1)}$ with maximum cosine with the target $\cos(\mathbf{x}_i, \mathbf{y}) = \mathbf{x}.\mathbf{y}/||\mathbf{x}||\ ||\mathbf{y}||$
- For each remaining feature X_i
  - Project X_i and the target Y on the null space of the features already selected
  - Compute the cosine of X_i with the target in the projection
  - Select the feature $X_{\nu(k)}$ with maximum cosine with the target in the projection.

Embedded method for the linear least square predictor

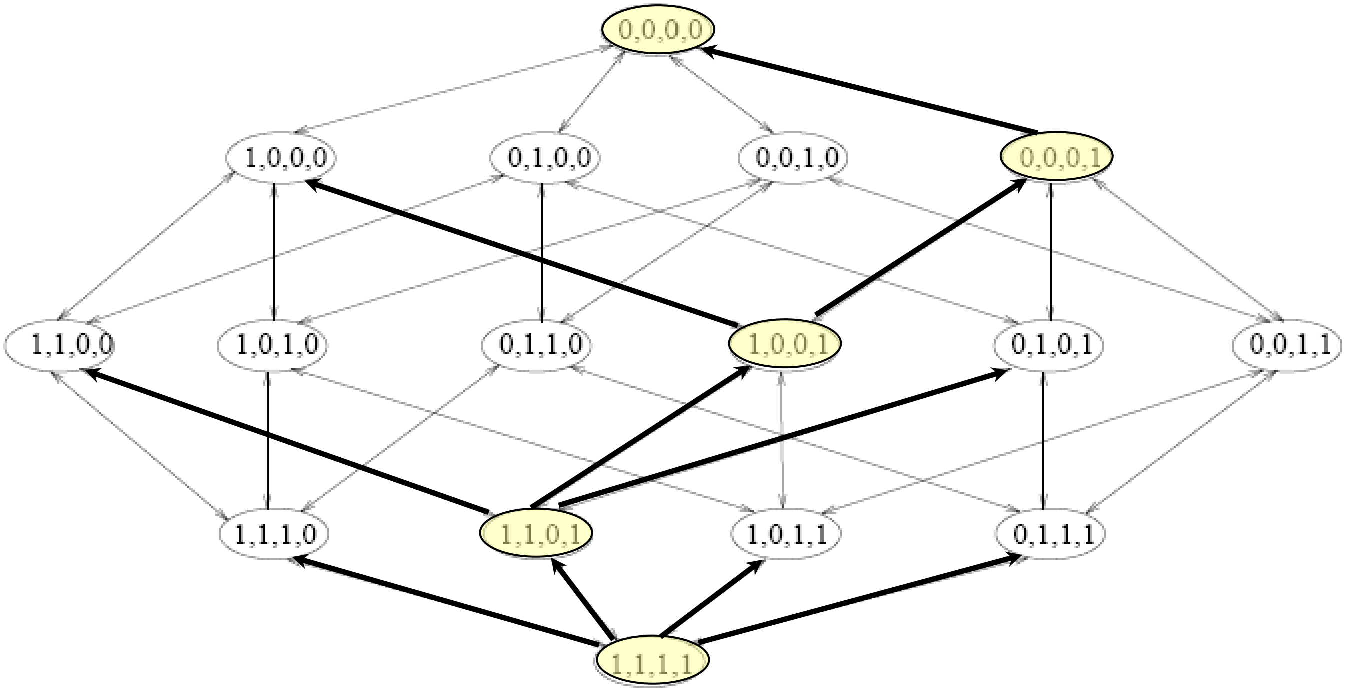
Forward Selection with Trees



At each step, choose the feature that "reduces entropy" most. Work towards "node purity".

Backward Elimination (Wrapper)

Also referred to as SBS: Sequential Backward Selection



1

:

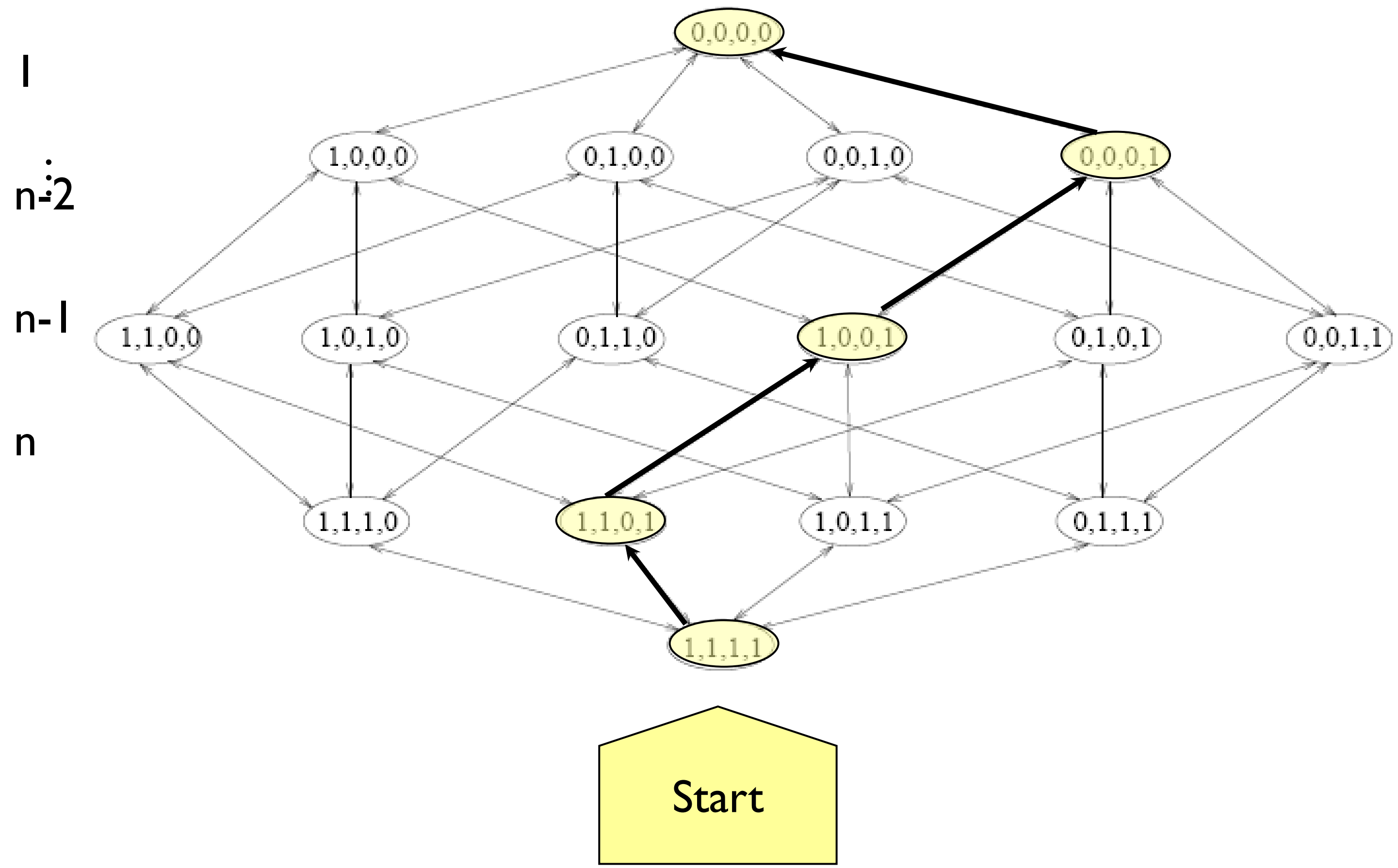
n-2

n-1

n

Start

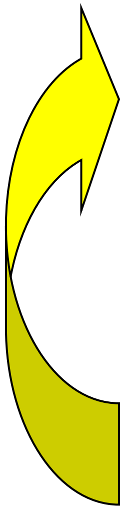
Backward Elimination (Embedded)



Backward Elimination: RFE

RFE-SVM, Guyon, Weston, et al, 2002

Start with all the features.

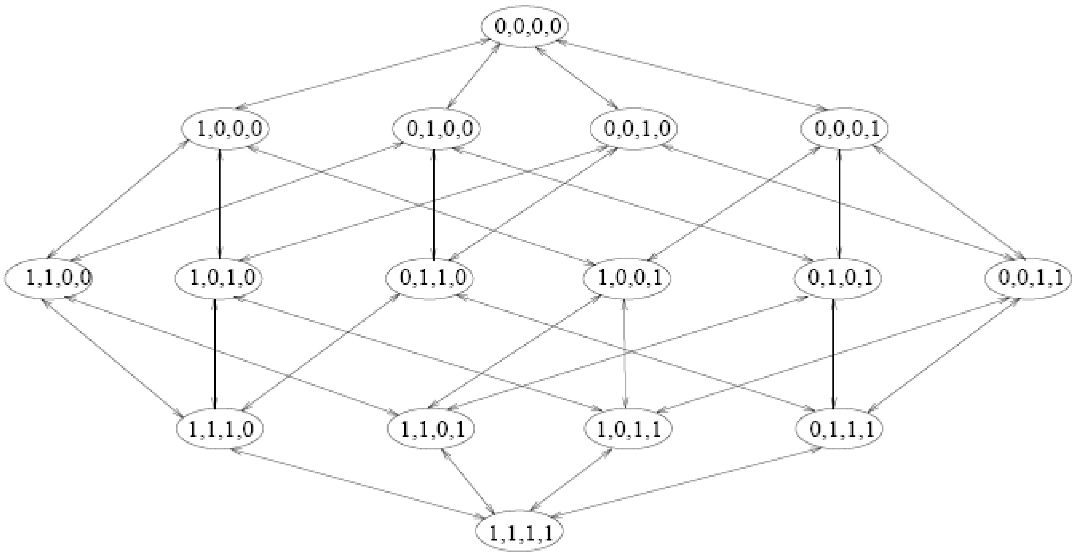


- Train a learning machine f on the current subset of features by minimizing a risk functional J[f].
- For each (remaining) feature X_i, estimate, without retraining f, the change in J[f] resulting from the removal of X_i.
- Remove the feature $X_{v(k)}$ that results in improving or least degrading J.

Embedded method for SVM, kernel methods, neural nets.

Scaling Factors

Idea: Transform a discrete space into a continuous space.



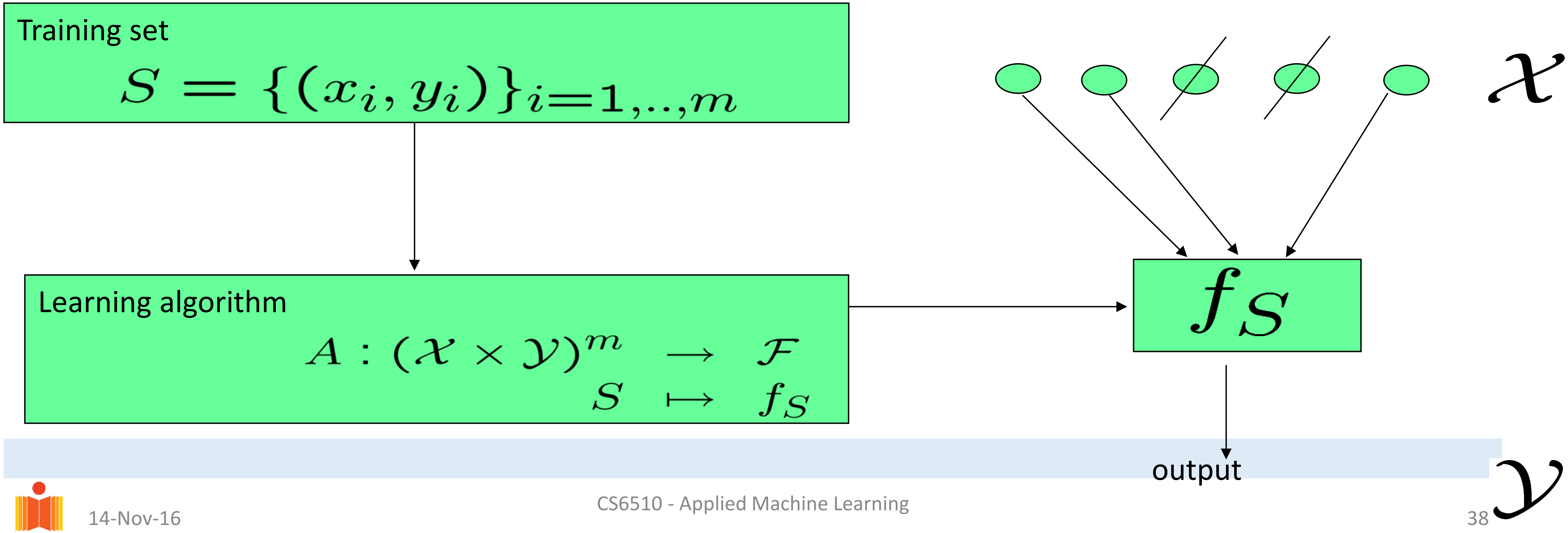
$\boldsymbol{\sigma}=[\sigma_1, \sigma_2, \sigma_3, \sigma_4]$

- Discrete indicators of feature presence: $\sigma_i \in \{0, 1\}$
- Continuous scaling factors: $\sigma_i \in R$

Now we can do gradient descent!

Formalism

- Definition: an embedded feature selection method is a *machine learning algorithm* that returns a model using a limited number of features.



$$S = \{(x_i, y_i)\}_{i=1,..,m}$$

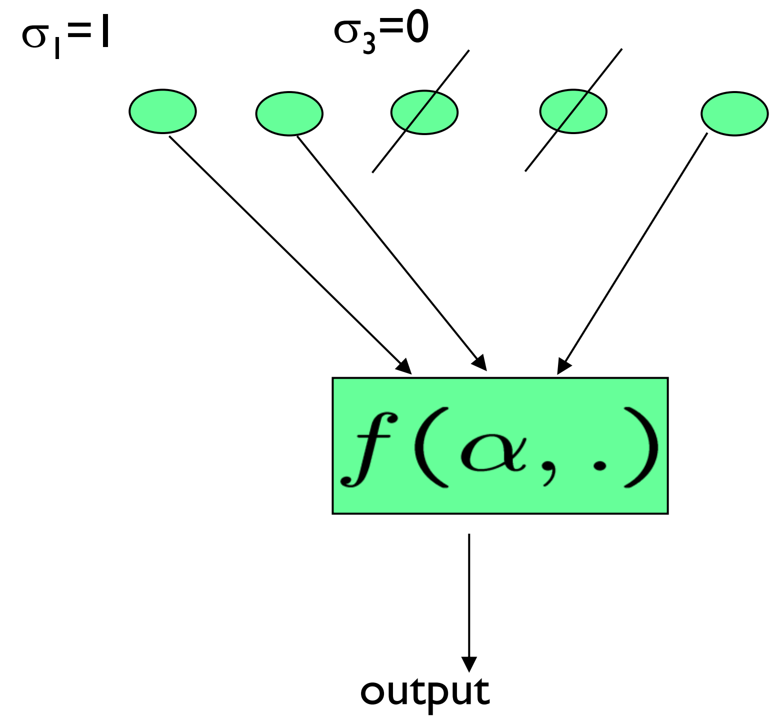
$$\begin{array}{rcl} A : (\mathcal{X} \times \mathcal{Y})^m & \to & \mathcal{F} \\ S & \mapsto & f_S \end{array}$$

Feature Selection as Model Selection

- Let us consider the following set of functions parameterized by α and where $\sigma \in \{0,1\}^n$ represents the use (σ_i=1) or rejection of feature i.

$$\sigma \circ x = (\sigma_1 x_1, .., \sigma_n x_n)$$

$$\begin{array}{rcl} f : \Lambda \times \mathbb{R}^n & \longrightarrow & \mathbb{R} \\ (\alpha, \sigma \circ x) & \longmapsto & f(\alpha, \sigma \circ x) \end{array}$$



Example (linear systems, α=w): $w.x + b = \sum_{i=1}^{n} w_i x_i \sigma_i = \sum_{\sigma_i \neq 0} w_i x_i$

Feature Selection as Model Selection

- We are interested in finding α and σ such that the generalization error is minimized:

$$\min_{\sigma,\alpha} R(\alpha,\sigma)$$

where $$R(\alpha,\sigma) = \int L(f(\alpha,\sigma \circ x), y)\,dP(x,y)$$

Sometimes we add a constraint: # non zero σ_i' s · s_0
Problem: the generalization error is not known…

Feature Selection as Model Selection

- The generalization error is not known directly but bounds can be used.
- Most embedded methods minimize those bounds using different optimization strategies:
  - Add and remove features
  - Relaxation methods and gradient descent
  - Relaxation methods and regularization

Example of bounds (linear systems):

Non separable

$$R(w,\sigma) \leq \frac{1}{m}\sum_{k=1}^{m} L(w.x_k + b, y_k) + O(\frac{r\|w\|}{\sqrt{m}})$$

Linearly separable

$$R(w,\sigma) \leq O(\frac{r^2\|w\|^2}{m})$$

Feature Selection as Model Selection

- How to minimize $\min_{\sigma,\alpha} R(\alpha,\sigma)$

Most approaches use the following method:

1. Set $\sigma = (1, ., 1)$
2. Compute $\alpha^* = \arg\min_\alpha \hat{R}(\alpha, \sigma)$
3. Compute $\sigma^* = \arg\min_\sigma \hat{R}(\alpha^*, \sigma)$
4. Set $\sigma \leftarrow \sigma^*$ and go back to 2.

This optimization is often done by relaxing the constraint
σ in $\{0,1\}^n$
as σ in $[0,1]^n$

Add/Remove Features

- Many learning algorithms are cast into a minimization of some regularized functional:

$$\underbrace{\min_{\alpha} \hat{R}(\alpha, \sigma)}_{G(\sigma)} = \min_{\alpha} \sum_{k=1}^{m} L(f(\alpha, \sigma \circ x_k), y_k) + \Omega(\alpha)$$

Empirical error

Regularization capacity control

- What does G(σ) become if one feature is removed?
- Sometimes, G can only increase… (e.g. SVM)

Add/Remove Features

- It can be shown (under some conditions) that the removal of one feature will induce a change in G proportional to:

$$\sum_{k=1}^{m} \left(\frac{\partial f}{\partial x^i}\right)^2 (\alpha, x_k)$$

Gradient of *f* wrt. ith feature at point x_k

- Examples: SVMs

! RFE ($\Omega(\alpha) = \Omega(w) = \sum_i w_i^2$) $\longrightarrow$ $\frac{\partial f}{\partial x^i} \propto w_i$

Add/Remove features - RFE

- Recursive Feature Elimination

1. Set $F = \{1, .., n\}$

2. Get w^* as the solution on a SVM on the data set restricted to features in F

3. Select top features as ranked by the $|w_i^*|$'s

4. Back to 2.

Minimize estimate of R(α,σ) wrt. α

Minimize the estimate R(α,σ) wrt. σ and under a constraint that only limited number of features must be selected

Add/Remove Features: Summary

Many algorithms can be turned into embedded methods for feature selections by using the following approach:

1. Choose an objective function that measure how well the model returned by the algorithm performs

1. "Differentiate" (or sensitivity analysis) this objective function according to the σ parameter (i.e. how does the value of this function change when one feature is removed and the algorithm is rerun)

1. Select the features whose removal (resp. addition) induces the desired change in the objective function (i.e. minimize error estimate, maximize alignment with target, etc.)

What makes this method an 'embedded method' is the use of the structure of the learning algorithm to compute the gradient and to search/weight relevant features.

Gradient Descent

- How to minimize $\min_{\sigma,\alpha} \dot{R}(\alpha,\sigma)$

Most approaches use the following method:

1. Set $\sigma = (1,.,1)$
2. Compute $\alpha^* = \arg\min_\alpha R(\alpha,\sigma)$
3. Compute $\sigma^* = \sigma - \lambda \nabla_\sigma R(\alpha^*,\sigma)$
4. Set $\sigma \leftarrow \sigma^*$ and go back to 2.

Would it make sense to perform just a gradient step here too?

Gradient step in $[0,1]^n$.

Gradient Descent

Advantage of this approach:

- can be done for non-linear systems (e.g. SVM with Gaussian kernels)
- can mix the search for features with the search for an optimal regularization parameters and/or other kernel parameters.

Drawback:

- heavy computations
- back to gradient based machine algorithms (early stopping, initialization, etc.)

Gradient Descent: Summary

- Many algorithms can be turned into embedded methods for feature selections by using the following approach:

1. Choose an objective function that measure how well the model returned by the algorithm performs
2. Differentiate this objective function according to the σ parameter
3. Performs a gradient descent on σ. At each iteration, rerun the initial learning algorithm to compute its solution on the new scaled feature space.
4. Stop when no more changes (or early stopping, etc.)
5. Threshold values to get list of features and retrain algorithm on the subset of features.

Difference from add/remove approach is the search strategy. It still uses the inner structure of the learning model but it scales features rather than selecting them.

Design Strategies Revisited

- Model selection strategy: find the subset of features such that the model is the best.
- Alternative strategy: Directly minimize the number of features that an algorithm uses (focus on feature selection directly and forget generalization error).
- In the case of linear system, feature selection can be expressed as:

$$\min_{w} \sum_{i=1}^{n} 1_{w_i \neq 0}$$

subject to $y_k\left(w.x_k + b\right) \geq 0$

NP Hard

- Amaldi and Kann (1998) showed that the minimization problem related to feature selection for linear systems is NP hard.

- Is feature selection hopeless?

- How can we approximate this minimization?

Minimization of a Sparsity Function

- Replace $\sum_{i=1}^{n} 1_{w_i \neq 0}$ by another objective function:
  - l_1 norm: → $||w||_1 = \sum_{i=1}^{n} |w_i|$
  - Differentiable function: → $\sum_{i=1}^{n} (1 - \exp^{-\alpha |w_i|})$
- Do the optimization directly!

L_0-SVM

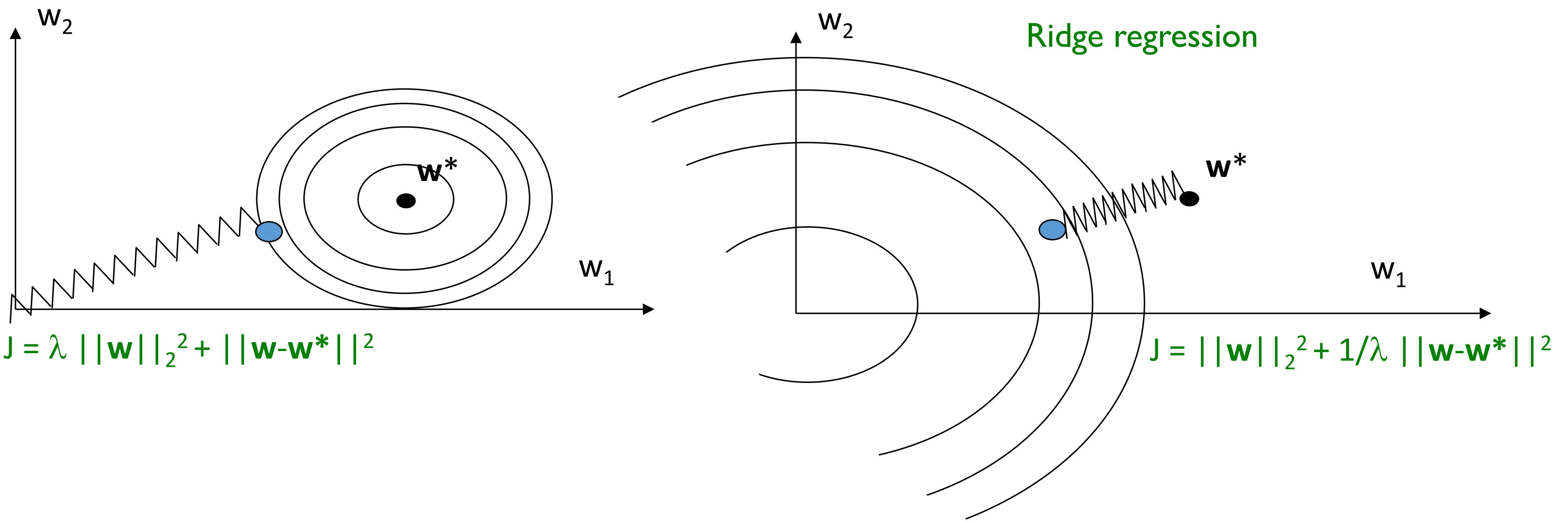
- Replace the regularizer $||w||^2$ by the l_0 norm $\sum_{i=1}^{n} 1_{w_i \neq 0}$
- Further replace $\sum_{i=1}^{n} 1_{w_i \neq 0}$ by $\sum_i \log(\varepsilon + |w_i|)$
- Boils down to the following multiplicative update algorithm:

1. Set $\sigma = (1, .., 1)$
2. Get w^* solution of an SVM on data set where each input is scaled by σ.
3. Set $\sigma = w^* \circ \sigma$
4. back to 2.

L_1-SVM

- The version of the SVM where $||w||^2$ is replace by the l_1 norm $\sum_i |w_i|$ can be considered as an embedded method:
  - Only a limited number of weights will be non zero (tend to remove redundant features)
  - Difference from the regular SVM where redundant features are all included (non zero weights)

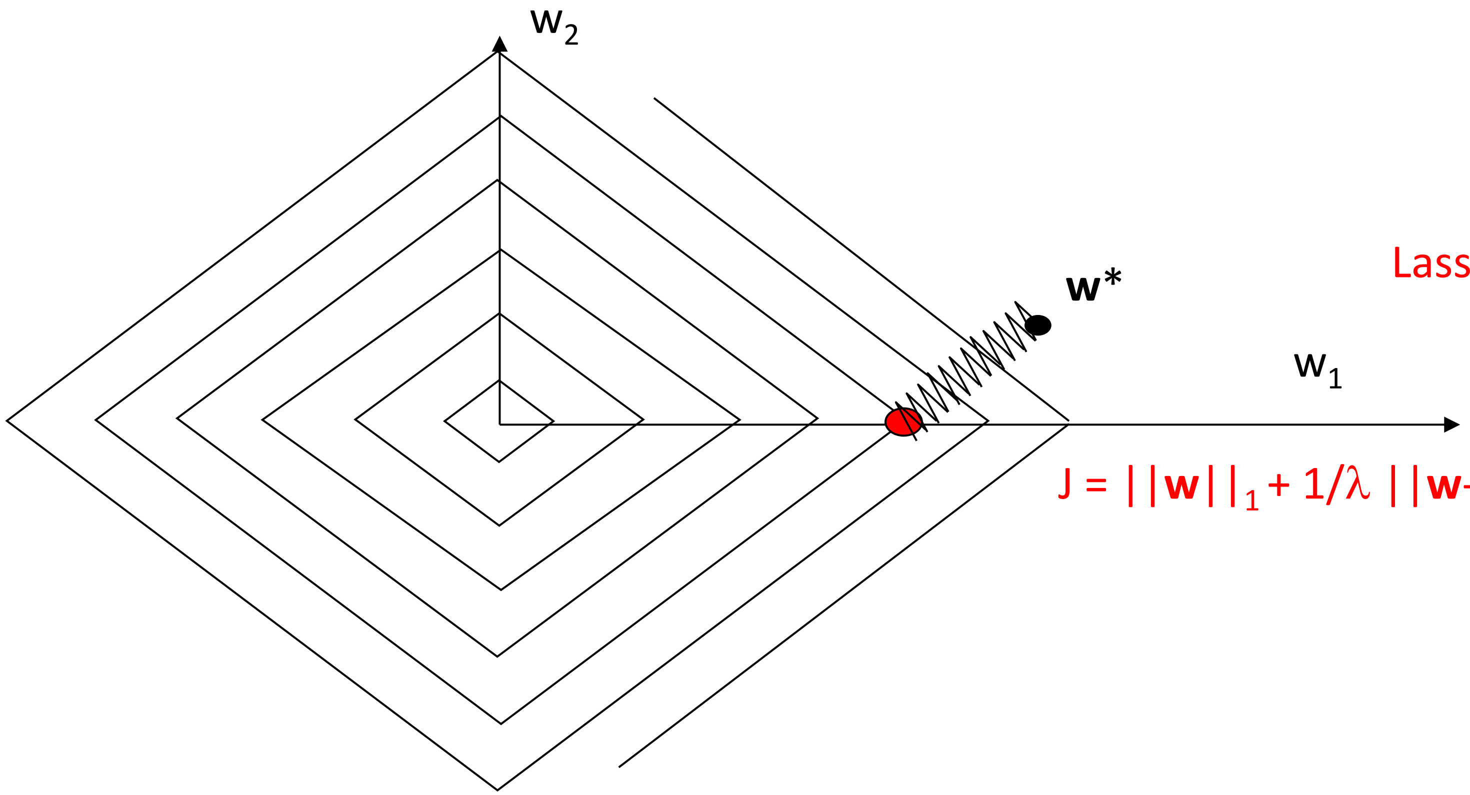
Effect of Regularizer



$J = \lambda \, ||\mathbf{w}||_2^2 + ||\mathbf{w}\text{-}\mathbf{w^*}||^2$

$J = ||\mathbf{w}||_2^2 + 1/\lambda \, ||\mathbf{w}\text{-}\mathbf{w^*}||^2$

Mechanical Interpretation



Lasso

$J = ||\mathbf{w}||_1 + 1/\lambda\ ||\mathbf{w\text{-}w^*}||^2$

Embedded Methods: Summary

- Embedded methods are a good inspiration to design new feature selection techniques for your own algorithms:
    - Find a functional that represents your prior knowledge about what a good model is.
    - Add the **s** weights into the functional and make sure it's either differentiable or you can perform a sensitivity analysis efficiently
    - Optimize alternatively according to **a** and **s**
    - Use early stopping (validation set) or your own stopping criterion to stop and select the subset of features

- Embedded methods are therefore not too far from wrapper techniques and can be extended to multiclass, regression, etc…

Complexity of Feature Selection

With high probability:

$$\text{Generalization\_error} \leq \text{Validation\_error} + \varepsilon(C/m_2)$$

m_2: number of *validation* examples
N: total number of features
n: feature subset size

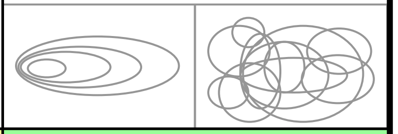
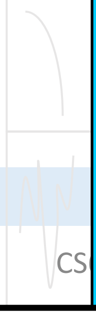
Method	**Number of subsets tried**	**Complexity C**
Exhaustive search wrapper	2^N	N
Nested subsets Feature ranking	N(N+1)/2 or N	log N

Try to keep C of the order of m_2.

Examples of FS Algorithms

keep C = O(m_2)

keep C = O(m_1)

	Univariate	Multivariate 
Linear	T-test, AUC, feature ranking	RFE with linear SVM or LDA
Non-linear 	Mutual information feature ranking	Nearest Neighbors Neural Nets Trees, SVM

In Practice…

- **No method is universally better:**
  - wide variety of types of variables, data distributions, learning machines, and objectives.
- **Match the method complexity to the ratio M/N:**
  - univariate feature selection may work better than multivariate feature selection; non-linear classifiers are not always better.
- **Feature selection is not always necessary to achieve good performance.**

Readings

- “Introduction to Machine Learning” by Ethem Alpaydin, Chapter 6
- http://machinelearningmastery.com/an-introduction-to-feature-selection/
- Introduction to Feature and Variable Selection by Guyon