FEATURE SELECTION

Consider input data $x = [x^{(1)} \; \cdots \; x^{(d)}]^T$

Feature selection is the problem of selecting a subset of the variables $x^{(1)}, \ldots, x^{(d)}$ that are most relevant for a machine learning task, such as classification or regression.

There are three main reasons why we might want to perform feature selection

Ⓐ

-
-
-

Methods for feature selection fall into three categories:

- filter methods
- wrapper methods
- embedded methods

The Curse of Dimensionality

Feature selection can improve performance by eliminating <u>irrelevant</u> features.

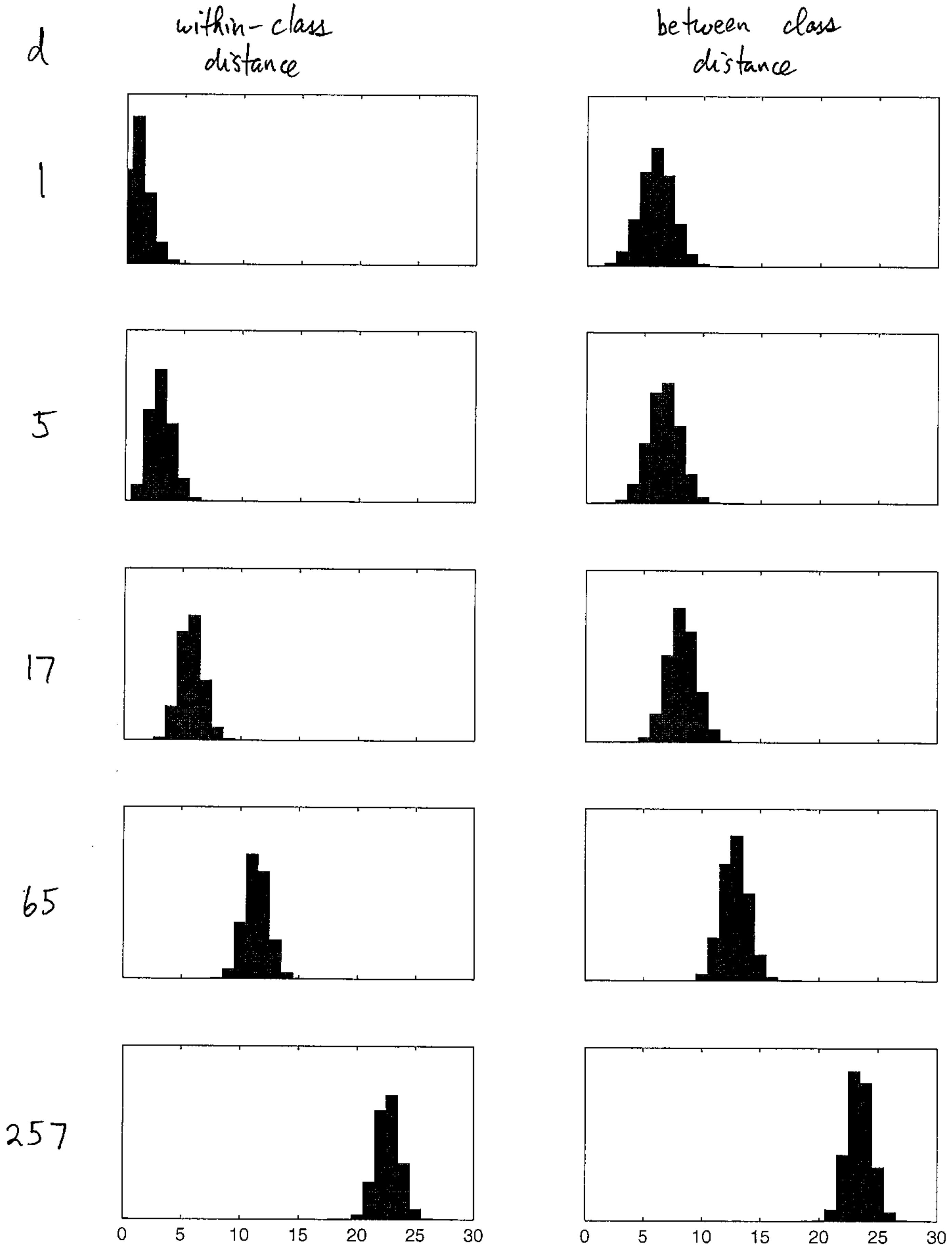
<u>Example</u> | Consider a classification problem where

$$x \mid y=0 \sim N(\mu_0, I) \qquad \mu_0 = [-1, 0, \ldots, 0]^T$$
$$x \mid y=1 \sim N(\mu_1, I), \qquad \mu_1 = \underbrace{[1, 0, \ldots, 0]}_{d}^T$$

Only the first feature is relevant.

However, as $d \to \infty$, the distance between two points with the same label is the same as the distance between two points with the opposite labels.

This is one manifestation of the "curse of dimensionality," which broadly refers to the computational/statistical challenges to learning as dimension increases.


d
within-class distance
between class distance
1
5
17
65
257
0
5
10
15
20
25
30

$\boxed{\text{Filter Methods}}$

Ⓑ Filter methods attempt to ______ features in order of importance. In supervised learning, "importance" is usually related to the ability of a feature to ______ the label or response variable.

Ⓒ Advantage:

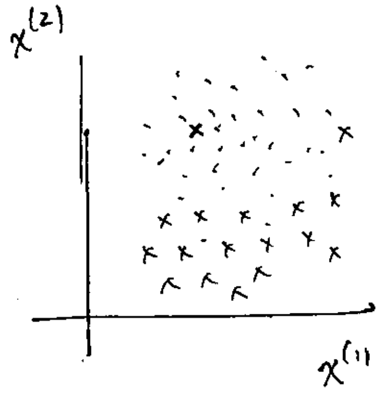
Disadvantage:

Classification

Consider training data $(x_1, y_1), \ldots, (x_n, y_n)$, where

$$x_i = \begin{bmatrix} x_i^{(1)} \\ \vdots \\ x_i^{(d)} \end{bmatrix} \in \mathbb{R}^d, \quad y_i \in \{-1, +1\}$$

How should we rank the features?



(D)

- misclassifation rate

$$m^{(j)} =$$

- two sample t-test statistic

$$t^{(j)} =$$

- margin

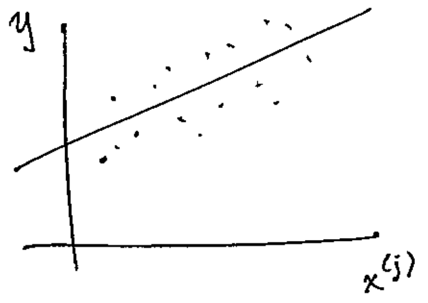
$$\gamma^{(j)} =$$

Linear Regression

In linear regression, we have training data $(x_1, y_1), \dots, (x_n, y_n)$, where $y_i \in \mathbb{R}$, and

(E) we expect y to change ______________ in response to changes in $x^{(j)}$.

How should we rank the features?



Ⓔ • correlation coefficient : $|\rho^{(j)}|$ where

$$\rho^{(j)} =$$

$$=$$

Mutual Information

The MI between X and Y is

$$I(X;Y) := \sum_x \sum_y p(x,y) \log\left[\frac{p(x,y)}{p(x)\,p(y)}\right]$$

This is the Kullback-Leibler divergence between $p(x,y)$ and $p(x)p(y)$. Note $I=0$ if X,Y independent.

If X^S denotes a subset of features corresponding to $S \subseteq \{1,2,\dots,d\}$, ideally we'd like to maximize

$$I(X^S;Y)$$

w.r.t S. Unfortunately this is typically ~~intractable~~ table

We could rank the features according to

$$I(X^{(j)};Y)$$

With MI, instead of simply ranking, we can choose features that are not <u>redundant</u>. For example, we can select features incrementally, say $X^{(j_1)}, X^{(j_2)}, \ldots,$ so that $X^{(j_k)}$ maximize

$$I(X^{(j_k)}, Y) - \beta \sum_{i=1}^{k-1} I(X^{(j_k)}, X^{(j_i)}).$$

MI can be extended to continuous X by replacing Σ with $\int$ and estimating densities, e.g., with KDEs.

Wrapper Methods

Wrapper methods have 3 basic ingredients:

1) a machine learning algorithm
2) a way to assess the performance of a subset of features
3) a strategy for searching through subsets of features

Ⓖ Advantage:

Disadvantage:

Wrapper methods are so called because they "wrap around" the basic learning algorithm, calling it many times on different feature subsets.

Examples

1)

2)

3) Forward selection

Backward elimination

Others?

$\boxed{\text{Embedded Methods}}$

Embedded methods perform feature selection jointly together with ~~model~~ fitting. Let's look at an example:

The Lasso

The Lasso is a method for linear regression. It is like ridge regression, except we use the l_1 penalty:

$$\hat{\beta} \leftarrow \underset{\beta}{\arg\min} \sum_{i=1}^{n} \left(y_i - \beta^T x_i\right)^2 + \lambda \|\beta\|_1$$

where

$$\|\beta\|_1 = \sum_{j=1}^{d} |\beta^{(j)}|$$

More generally,

$$\|\beta\|_p = \left(\sum_{j=1}^{d} |\beta^{(j)}|^p\right)^{\frac{1}{p}}$$

Denoteng

$$\underline{y} = \begin{bmatrix} y_1 \\ \\ \\ y_n \end{bmatrix}, \quad \underline{X} = \begin{bmatrix} x_1^{(1)} & \cdots & x_1^{(d)} \\ \vdots & & \\ \vdots & & \\ x_n^{(1)} & \cdots & x_n^{(d)} \end{bmatrix}$$

we may write the problem as

④ $$\hat{\beta} = \arg\min_{\beta}$$

s.t.

⑤ This optimization problem can be solved via

__________ __________.

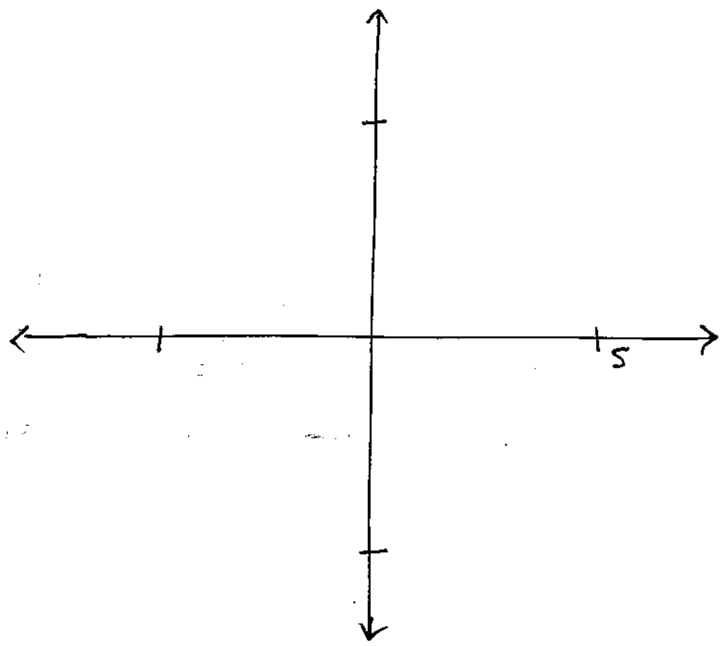
<u>Claim</u> The ℓ_1-penalized LS solution is __________ $\Longrightarrow$ simultaneous regression and feature selection

The reason has to do with the shape of the "ball"

$$\{\beta : \|\beta\|_p = s\} := B_p(s)$$

when $p = 1$

Ⓛ

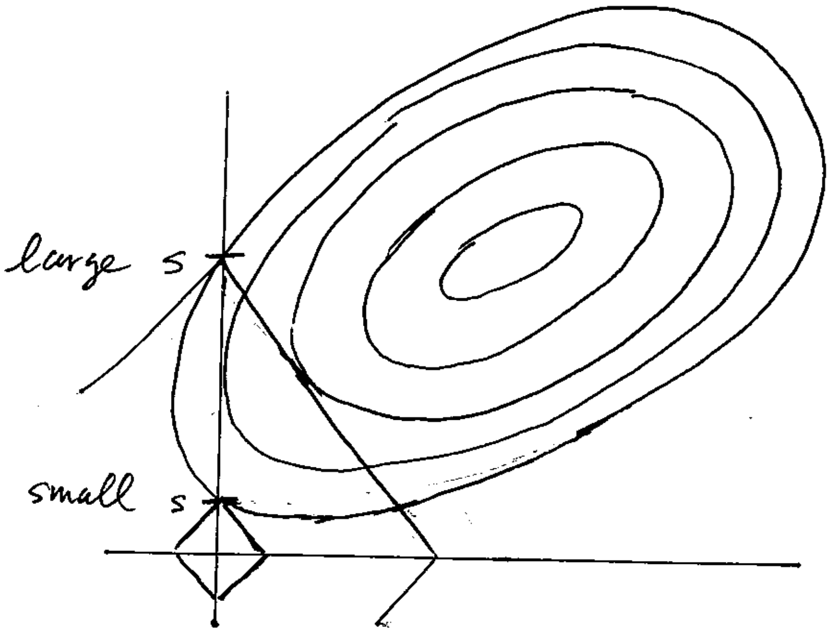


Now consider the set of all β with a given squared error:

$$\{\beta : \|\underline{y} - \underline{X}\beta\|^2 = c\}$$

(M) This is an ____________.

Therefore we have the following picture:



The ellipse tends to intersect the ℓ_1-ball

(N) $B_1(s)$ at a corner $\Longrightarrow$

In fact, this argument applies for any $p \le 1$, but if $p < 1$ the optimization

(O) problem is ____________.

Decreasing s / increasing λ $\Longrightarrow$

(P) •

•

References

- Guyon + Elisseeff, "An Introduction to Variable and Feature Selection," JMLR 2003.
- Hastie, Tibshirani + Friedman

Key A: understanding / interpretation, computational efficiency, performance

B. rank, predict

C. Advantage: simple, fast

Disadvantage: best k features $\neq$ k best features

D. $$m^{(j)} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{ y_i \neq \theta(x_i^{(j)}) \}, \theta = \text{threshold classifier}$$

$$t^{(j)} = \left| \frac{\overline{x_+^{(j)}} - \overline{x_-^{(j)}}}{s/\sqrt{n}} \right|, \quad s = \text{pooled sample standard dev.}$$

$$\gamma^{(j)} = \min_{\substack{k: y_k = 1 \\ l: y_l = -1}} \left| x_k^{(j)} - x_l^{(j)} \right| \quad \text{in separable case}$$

E. linearly

F. $$\rho^{(j)} = \frac{\text{cov}(x^{(j)}, y)}{\sqrt{\text{Var}(x^{(j)}) \cdot \text{Var}(y)}} = \frac{\sum_{i=1}^{n} (x_i^{(j)} - \overline{x^{(j)}})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i^{(j)} - \overline{x^{(j)}})^2 \cdot \sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

G. Advantage: captures feature interactions

Disadvantage: slow

H. 1) K-NN, LDA, LR, SVM, ...

2) holdout, cross-validation, ...

I. $$\hat{\beta} = \arg\min_{\beta} \|\underline{y} - \underline{X}\beta\|^2$$

$$\text{s.t.} \quad \|\beta\|_1 \leq s.$$

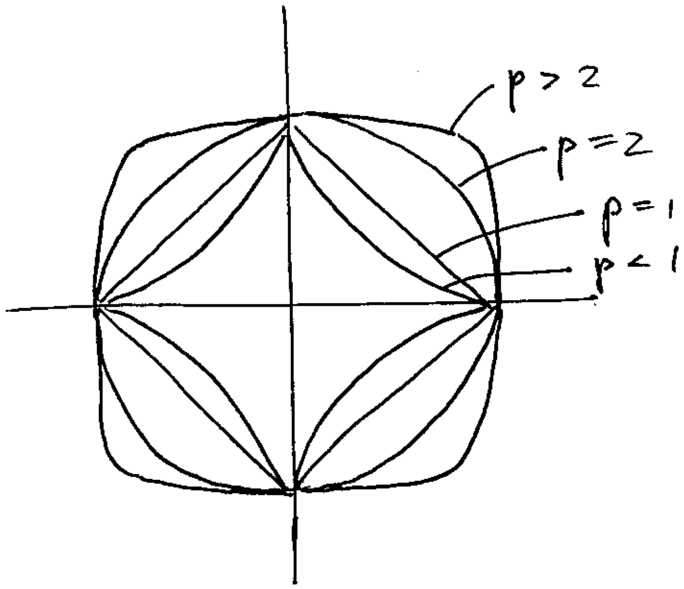
J. quadratic programming: $$\arg\min_{\beta} \|\underline{y} - \underline{X}\beta\|^2$$

$$\text{s.t.} \quad -t_j \leq \beta^{(j)} \leq t_j$$

$$\sum t_j \leq s$$

K.

L.



M. ellipse

N. sparsity

O. nonconvex

P. increasing sparsity, increasing squared error briefly review model selection methods and statistical tests used to that effect (Section 6). Finally, we conclude the paper with a discussion section in which we go over more advanced issues (Section 7). Because the organization of our paper does not follow the work flow of building a machine learning application, we summarize the steps that may be taken to solve a feature selection problem in a check list[2]:

1. **Do you have domain knowledge?** If yes, construct a better set of "ad hoc" features.
2. **Are your features commensurate?** If no, consider normalizing them.
3. **Do you suspect interdependence of features?** If yes, expand your feature set by constructing conjunctive features or products of features, as much as your computer resources allow you (see example of use in Section 4.4).
4. **Do you need to prune the input variables** (e.g. for cost, speed or data understanding reasons)? If no, construct disjunctive features or weighted sums of features (e.g. by clustering or matrix factorization, see Section 5).
5. **Do you need to assess features individually** (e.g. to understand their influence on the system or because their number is so large that you need to do a first filtering)? If yes, use a variable ranking method (Section 2 and Section 7.2); else, do it anyway to get baseline results.
6. **Do you need a predictor?** If no, stop.
7. **Do you suspect your data is "dirty"** (has a few meaningless input patterns and/or noisy outputs or wrong class labels)? If yes, detect the outlier examples using the top ranking variables obtained in step 5 as representation; check and/or discard them.
8. **Do you know what to try first?** If no, use a linear predictor.[3] Use a forward selection method (Section 4.2) with the "probe" method as a stopping criterion (Section 6) or use the ℓ_0-norm embedded method (Section 4.3). For comparison, following the ranking of step 5, construct a sequence of predictors of same nature using increasing subsets of features. Can you match or improve performance with a smaller subset? If yes, try a non-linear predictor with that subset.
9. **Do you have new ideas, time, computational resources, and enough examples?** If yes, compare several feature selection methods, including your new idea, correlation coefficients, backward selection and embedded methods (Section 4). Use linear and non-linear predictors. Select the best approach with model selection (Section 6).
10. **Do you want a stable solution** (to improve performance and/or understanding)? If yes, sub-sample your data and redo your analysis for several "bootstraps" (Section 7.1).

2. We caution the reader that this check list is heuristic. The only recommendation that is almost surely valid is to try the simplest things first.
3. By "linear predictor" we mean linear in the parameters. Feature construction may render the predictor non-linear in the input variables.