FEATURE SELECTION

Feature Selection

Feature selection is the problem of selecting a subset of features of a feature vector $x = [x^{(1)} \dots x^{(d)}]^T$ that are most relevant for a machine learning task, such as classification or regression.

Reasons to perform feature selection:

1. understanding / interpreting data
2. computational efficiency
3. improve performance

There are three primary categories of feature selection methods:

1. Filter methods
2. Wrapper methods
3. Embedded methods

The Curse of Dimensionality

This is the idea that data analysis becomes more

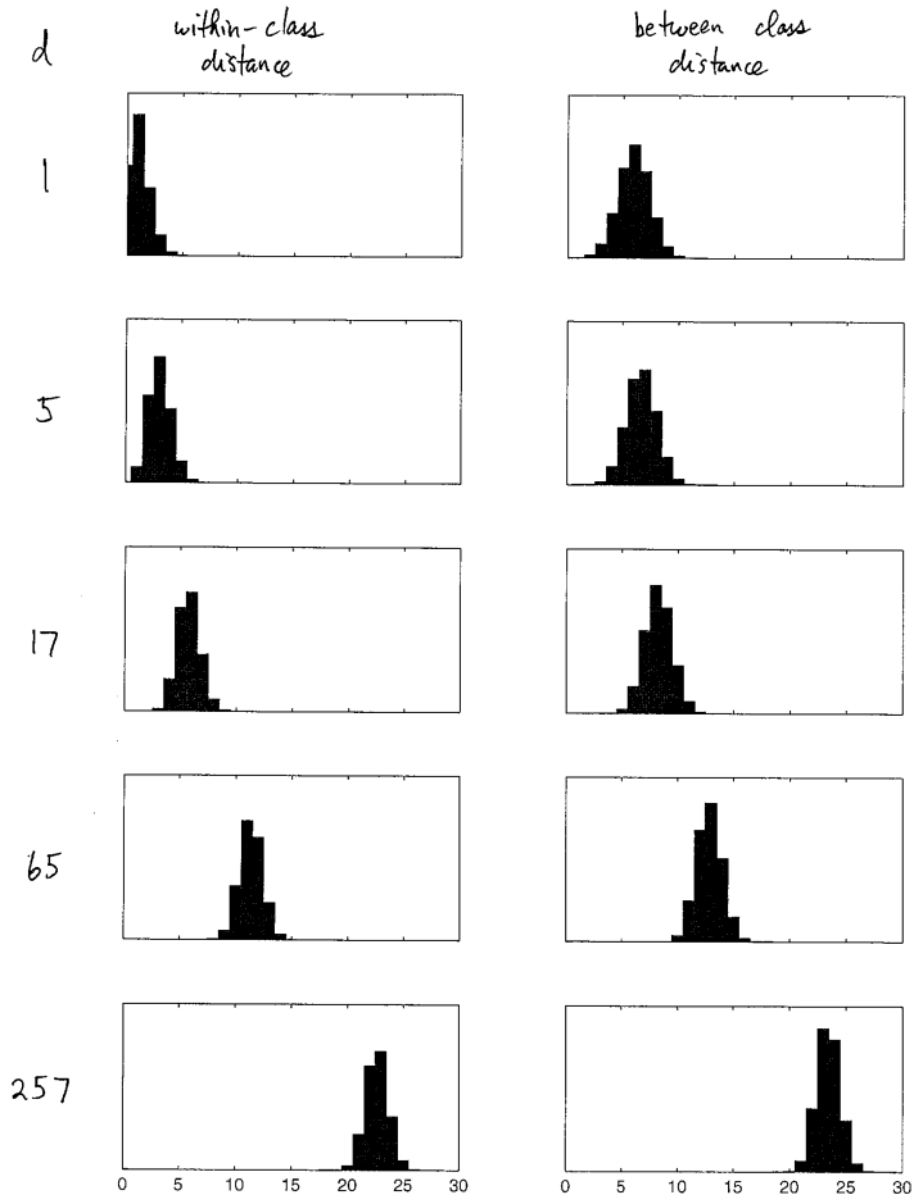
difficult, both statistically and computationally, as the dimension increases. It can be quantified in different ways. Here is on example that relates to feature selection.

Example Consider a classification problem where

$$X|Y=1 \sim N(\mu_1, I), \quad \mu_1 = [1, 0, \ldots, 0]^T$$

$$X|Y=-1 \sim N(\mu_{-1}, I), \quad \mu_{-1} = [-1, 0, \ldots, 0]$$

Only the first feature is relevant for classification. However, as $d \to \infty$, the distance between to random points in the same class has the same distribution as the distance between two random points in opposite classes.



Feature selection can significantly improve performance when only a few features are relevant.

Filter Methods

Basic idea: sort the features by estimated relevance, and take the top k, where k is the desired # of features.

Consider a supervised learning problem with data $(x_1, y_1), \ldots, (x_n, y_n)$. In classification a common way to rank features is by $|t^{(j)}|$ where

$$t^{(j)} = \frac{\overline{x}_1^{(j)} - \overline{x}_{-1}^{(j)}}{s^{(j)}/\sqrt{n}},$$

this is the two-sample t-statistic

$\overline{x}_k^{(j)}$ = sample mean of $\{x_i^{(j)} \mid y_i = k\}$

$s^{(j)}$ = pooled sample standard dev. of $\{x_i^{(j)}\}$

For regression, one can rank by $|\rho^{(j)}|$ where

$$\rho^{(j)} = \frac{\text{cov}(X^{(j)}, Y)}{\sqrt{\text{Var}(X^{(j)})\,\text{Var}(Y)}}$$

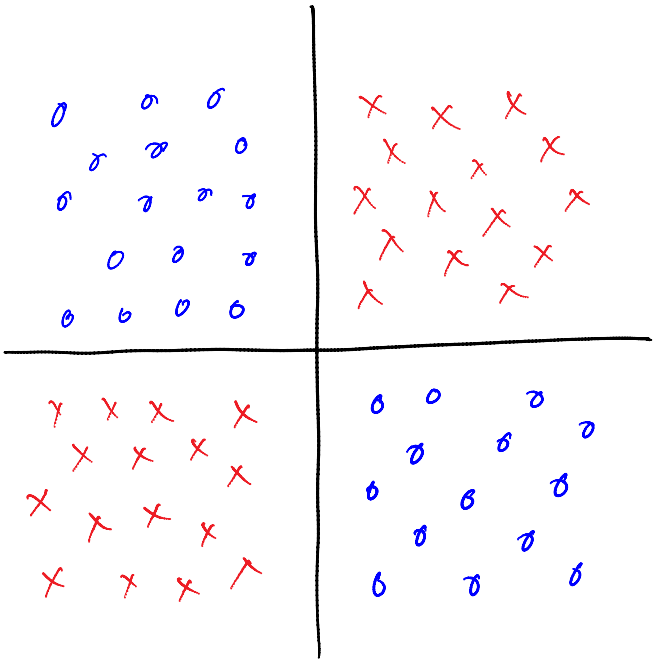
correlation coefficient

$$= \frac{\sum_{i=1}^{n} (x_i^{(j)} - \overline{x}^{(j)})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i^{(j)} - \overline{x}^{(j)})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

Advantage: Fast

Disadvantage: The k best features are generally not the best k

Example / XOR data



Each feature is useless by itself, but taken together they can be used to perfectly classify the data.

Another common ranking statistic that is used for both classification and regression is mutual information.

Wrapper Methods

Wrapper methods have three basic ingredients:

1. A machine learning algorithm
2. A method for evaluating the performance of the algorithm when trained on a subset of features

3. A strategy for searching through subsets of features

Examples!

1. LDA, logistic regression, SVM, kernel ridge regression
2. holdout, cross-validation
3. Forward selection:
   - Start with $S = \{\}$
   - Given a subset S, increase the subset to $S \cup \{j\}$ where j gives the biggest increase in performance

   Backward elimination
   - Start with $S = \{1, \ldots, d\}$
   - Given a subset S, decrease the subset to $S \setminus \{j\}$ where $j \in S$ gives the smallest decrease in performance.

Advantage: Captures feature interactions

Disadvantage: Slow

Wrapper methods are so-called because they wrap around the basic ML algorithm, running it many times on different subsets of features.

Embedded Methods

Embedded methods perform feature selection and function estimation simultaneously — feature selection is embedded within the ML algorithm.

We will focus on one example in particular, the Lasso. Lasso is an acronym for "least absolute shrinkage and selection operator."

Lasso is a method for linear regression that solves

$$\min_{w,b} \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i - b)^2 + \lambda\|w\|_1$$

where

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

is called the l_1 norm. More generally, for $0 < p < \infty$,

define

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

This is a norm for $p \geq 1$ (for $0 < p < 1$ the triangle inequality fails).

As we saw previously in our study of least squares/ridge regression, the optimal b is

$$\hat{b} = \bar{y} - \hat{w}^T \bar{x}$$

where

$$\hat{w} = \arg\min_w \frac{1}{n} \sum_{i=1}^{n} (\tilde{y}_i - w^T \tilde{x}_i)^2 + \lambda \|w\|_1$$

with $\tilde{y}_i = y_i - \bar{y}$, $\tilde{x}_i = x_i - \bar{x}$. In matrix-vector form, this is

$$\hat{w} = \arg\min_w \frac{1}{n} \|\tilde{y} - \tilde{X} w\|_2^2 + \lambda \|w\|_1$$

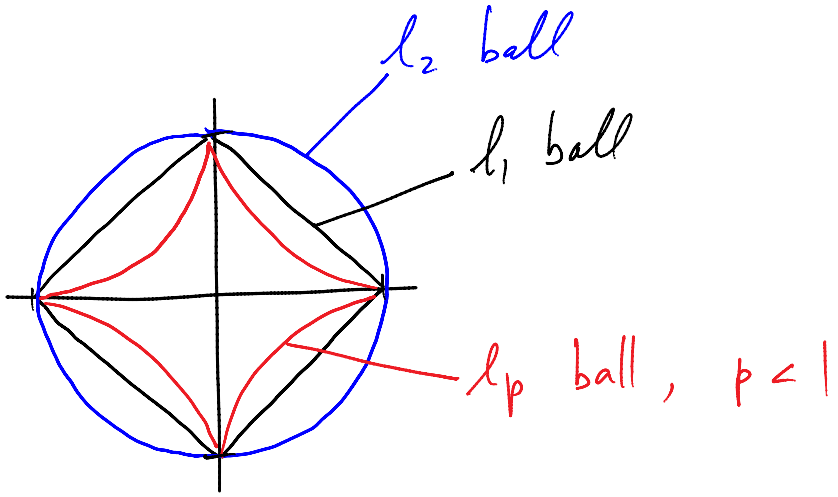
$$= \arg\min_w \|\tilde{y} - \tilde{X} w\|_2^2$$

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

by Lagrange multiplier theory

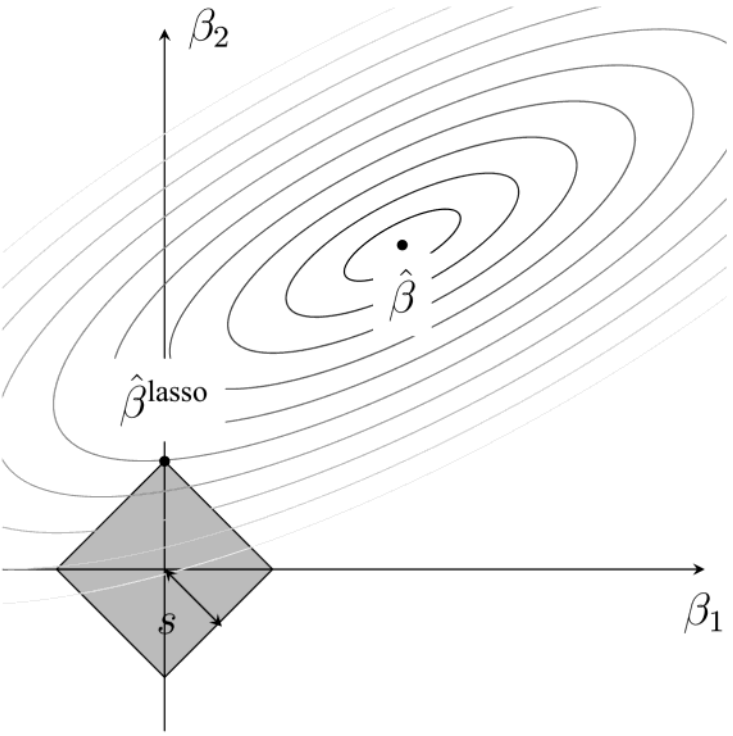
The key observation about the ℓ_1-penalized LS solution is that $\hat{w}$ is sparse, meaning that the method automatically selects relevant features.

One explanation of this property is based on the shape of the ℓ_1 ball $\{w : \|w\|_1 \le s\}$.



Meanwhile, $\{w \mid \|\tilde{y} - \tilde{X}w\|^2 = c\}$ is an ellipse.

Thus we have the following picture:



The ellipses tend to intersect the diamond-shaped l_1-ball where many $w^{(j)} = 0$. The smaller s (equivalently, the larger λ), the sparser $\hat{w}$.

In fact, the above argument applies to other l_p balls, $p < 1$, but $\|w\|_p$ is nonconvex in w for $p < 1$.

A significant disadvantage of the l_1 penalty relative to l_2 is that the problem cannot be kernelized.

References

- Guyon and Elisseeff, "An Introduction to Variable and Feature Selection," JMLR 2003
- Hastie, Tibshirani, & Friedman, "The Elements of Statistical Learning."