

## Feature Selection

Why feature selection?

1. Some algorithms scale (computationally) poorly with increased dimension.
2. Irrelevant features can confuse some algorithms.
3. Redundant features adversely affect regularization.
4. Removal of features can increase (relative) margin.
5. Reduces data set/model file size.

Varieties of feature selection algorithms:

1. Filtration: ignores labels
2. Preprocessing: agnostic to subsequence learning algorithm
3. In-the-loop: tries different combinations for a particular learner

These are in increasing order of usefulness and decreasing order of ease.

---

### Filtration

Just remove any (binary) feature that occurs fewer than  $k$  times ( $k$  is typically one of  $\{2, 3, 4, 5, 10\}$ , depending on data set size).

Extension to non-binary features unclear.

---

### Preprocessing

Sort features by some “usefulness” score and take the top  $k$  (hundred, thousand, ten-thousand, depending on feature set size). Standard metric: information gain.

Compute IG for each feature on the training data *independently*. This will remove irrelevant features, but not redundant features.

Redundancy extension: take first  $k$  features and select  $k+1$ st feature to have maximal IG over  $y$  and minimal IG over previous  $k$  features. Computationally expensive for large feature sets.

Other metrics than IG possible: classification rate, BNS, correlation-coefficient, etc... basically the same deal as for decision trees.

Largely independent of subsequent learning algorithm (aside from choosing cutoff).

---

### In-the-loop

Train  $D$  many classifiers, each using only one feature. Choose the best. Then train  $D - 1$  classifiers, each using the first feature plus one additional. Choose the best. Then train  $D - 2$  classifiers, . . .

Features are selected (greedily) according to classification accuracy.

Hugely inefficient, unless the learning algorithm is really simple (eg., naive Bayes).