

k-Nearest Neighbors

Machine Learning - 10601

Geoff Gordon, Miroslav Dudík

[partly based on slides of Carlos Guestrin and Andrew Moore]

<http://www.cs.cmu.edu/~ggordon/10601/>

October 19, 2009

Task: Classification

Goal: learn a map $h: \mathbf{x} \mapsto \mathbf{y}$

Data: $(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2), \dots, (\mathbf{x}^N, \mathbf{y}^N)$

So far:

- logistic regression:

$$p(Y=1|X=\mathbf{x}) = \frac{1}{1 + \exp\{-w_0 - \sum_{j=1}^k w_j x_j\}}$$

- Naïve Bayes:

$$p(\mathbf{X}, \mathbf{Y}) = p(\mathbf{Y}) \prod_{j=1}^k p(X_j | \mathbf{Y})$$

$p(Y=1|X=\mathbf{x})$ the same functional form as logistic regression provided that:

Logistic regression, naïve Bayes & linear separation

$$p(Y=1|X=x) = \frac{1}{1 + \exp\{-w_0 - \sum_{j=1}^k w_j x_j\}}$$

classification rule: $h(x) = 1$ iff $\sum_{j=1}^k w_j x_j \geq w_0$

Good: if classes can be separated by a hyperplane

- positive examples:

$$w_0 + \sum_{j=1}^k w_j x_j > 0$$

- negative examples:

$$w_0 + \sum_{j=1}^k w_j x_j < 0$$

Dealing with non-linear decision boundary

Say $Y = X_1 \text{ XOR } X_2$

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Dealing with non-linear decision boundary

Approach I:

include “non-linear” features

$$X_3 = X_1 X_2$$

Problems:

- which additional features?
at what degree polynomial should we stop?

Dealing with non-linear decision boundary

Approach II:

consider **non-linear hypotheses**

- k-nearest neighbors,
decision trees, artificial neural nets, ...

Problems:

- optimization non-convex,
may find only a local optimum

Nearest Neighbor Classifier:

simplest classifier in your toolkit

- store training data
- classify each test point according to the nearest training point

k-Nearest Neighbor Classifier:

k-th **simplest classifier in your toolkit**

- store training data
- classify each test point according to the majority among its **k** nearest neighbors

k-Nearest Neighbors

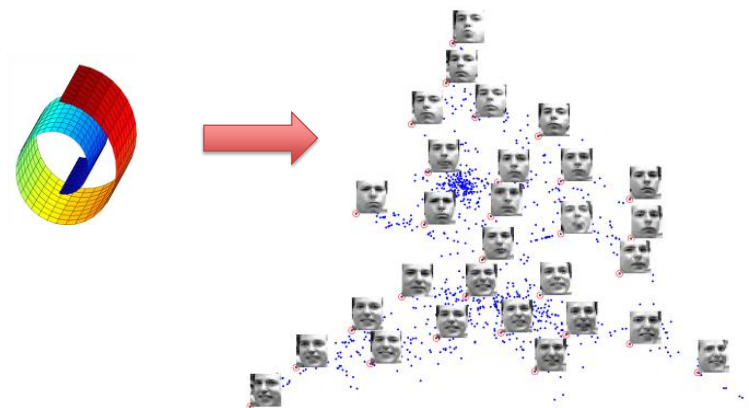
As **k** increases:

k-Nearest Neighbors

- **simple**—a useful baseline when developing new algorithms (for baseline often **k=1**)
- **general**—with enough data, arbitrary boundaries
- **downside:** need to store **all the data**

k-Nearest Neighbors

- works well if **intrinsic dimensionality low**
often in vision, e.g., **recognizing facial expressions**



k-Nearest Neighbors

For generalizations to regression:

Andrew Moore's tutorial on **Instance-based Learning**
<http://www.autonlab.org/tutorials/mbl.html>