Chapter 7 – K-Nearest-Neighbor

Data Mining for Business Intelligence

Shmueli, Patel & Bruce

Characteristics

Data-driven, not model-driven

Makes no assumptions about the data

Basic Idea

For a given record to be classified, identify nearby records

"Near" means records with similar predictor values X_1, X_2, ... X_p

Classify the record as whatever the predominant class is among the nearby records (the "neighbors")

How to measure “nearby”?

The most popular distance measure is **Euclidean distance**

$$\sqrt{(x_1 - u_1)^2 + (x_2 - u_2)^2 + \cdots + \left(x_p - u_p\right)^2}$$

Choosing k

K is the number of nearby neighbors to be used to classify the new record

- *K*=1 means use the single nearest record
- *K*=5 means use the 5 nearest records

Typically choose that value of *k* which has lowest error rate in validation data

Low *k* vs. High *k*

Low values of *k* (1, 3, …) capture local structure in data (but also noise)

High values of *k* provide more smoothing, less noise, but may miss local structure

Note: the extreme case of k = n (i.e., the entire data set) is the same as the "naïve rule" (classify all records according to majority class)

Example: Riding Mowers

Data: 24 households classified as owning or not owning riding mowers

Predictors: Income, Lot Size

Income	Lot_Size	Ownership
60.0	18.4	owner
85.5	16.8	owner
64.8	21.6	owner
61.5	20.8	owner
87.0	23.6	owner
110.1	19.2	owner
108.0	17.6	owner
82.8	22.4	owner
69.0	20.0	owner
93.0	20.8	owner
51.0	22.0	owner
81.0	20.0	owner
75.0	19.6	non-owner
52.8	20.8	non-owner
64.8	17.2	non-owner
43.2	20.4	non-owner
84.0	17.6	non-owner
49.2	17.6	non-owner
59.4	16.0	non-owner
66.0	18.4	non-owner
47.4	16.4	non-owner
33.0	18.8	non-owner
51.0	14.0	non-owner
63.0	14.8	non-owner

XLMiner Output

For each record in validation data (6 records) XLMiner finds neighbors amongst training data (18 records).

The record is scored for k=1, k=2, … k=18.

Best k appears to be k=8.

k = 9, k = 10, k=14 also share low error rate, but best to choose lowest k.

Value of k	% Error Training	% Error Validation	
1	0.00	33.33	
2	16.67	33.33	
3	11.11	33.33	
4	22.22	33.33	
5	11.11	33.33	
6	27.78	33.33	
7	22.22	33.33	
8	22.22	16.67	<--- Best k
9	22.22	16.67	
10	22.22	16.67	
11	16.67	33.33	
12	16.67	16.67	
13	11.11	33.33	
14	11.11	16.67	
15	5.56	33.33	
16	16.67	33.33	
17	11.11	33.33	
18	50.00	50.00	

Using K-NN for Prediction (for Numerical Outcome)

- Instead of “majority vote determines class” use average of response values
- May be a weighted average, weight decreasing with distance

Advantages

- Simple
- No assumptions required about Normal distribution, etc.
- Effective at capturing complex interactions among variables without having to define a statistical model

Shortcomings

- Required size of training set increases exponentially with # of predictors, *p*

  This is because expected distance to nearest neighbor increases with *p* (with large vector of predictors, all records end up "far away" from each other)

- In a large training set, it takes a long time to find distances to all the neighbors and then identify the nearest one(s)
- These constitute "curse of dimensionality"

Dealing with the Curse

- Reduce dimension of predictors (e.g., with PCA)
- Computational shortcuts that settle for “almost nearest neighbors”

Summary

- Find distance between record-to-be-classified and all other records
- Select k-nearest records
  - Classify it according to majority vote of nearest neighbors
  - Or, for prediction, take the as average of the nearest neighbors
- “Curse of dimensionality” – need to limit # of predictors