Tufts

Class #09: Uses of Nearest-Neighbors

Machine Learning (COMP 135): M. Allen, 02 Oct. 19

Uses of Nearest Neighbors

- Once we have found the k-nearest neighbors of a point, we can use this information:

1. ***In and of itself***: sometimes we just want to know what those nearest neighbors actually are (items that are similar to a given piece of data)
2. ***For additional classification purposes***: we want to find the nearest neighbors in a set of *already-classified* data, and then use those neighbors to classify new data
3. ***For regression purposes***: we want to find the nearest neighbors in a set of points for which we *already know* a functional (scalar) output, and then use those outputs to generate the output for some new data

Measuring Distances for Document Clustering & Retrieval

- Suppose we want to rank documents in a data-base or on the web based on how similar they are
  - We want a distance measurement that relates them
  - We can do a nearest-neighbor query for any article to get a set of those that are the closest (and most similar)
  - Searching for additional information based on a given document is equivalent to finding its nearest neighbors in the set of all document

The "Bag of Words" Document Model

- Suppose we have a set of documents $X = \{x_1, x_2, \ldots, x_n\}$
- Let $W = \{w \mid w \text{ is a word in some document } x_i\}$
- We can then treat each document x_i as a vector of word-counts (how many times each word occurs in the document):

$$C_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,|W|}\}$$

  - Assuming some fixed order of the set of words W
  - Not every word occurs in every document, so that some count values may be set to 0
- As previously noted, values tend to work better for purposes of classification if they are ***normalized***, so we set each value to be between 0 and 1 by dividing on largest count seen for ***any*** word in ***any*** document:

$$c_{i,j} \leftarrow \frac{c_{i,j}}{\max_{k,m} c_{k,m}}$$

Distances between Words

- We can now compute the distance function between any two documents (here we use the Euclidean):

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{|W|} (c_{i,k} - c_{j,k})^2}$$

- We could then build a KD-Tree, using the vectors of words as our dimension values, and query for some set of most similar documents to any document we start with
- ***Problem***: word counts turn out to be a lousy metric!
  - Common every-day words dominate the counts, making most documents appear quite similar, and making retrieval poor

Better Measures of Document Similarity

- We want to emphasize ***rare words*** over common ones:

1. Define word frequency: $t(w,x)$ as the (normalized) count of occurrences of word w in document x

$$c_x(w) = \#\text{ times word } w \text{ occurs in document } x$$

$$c_x^\star = \max_{w \in W} c_x(w)$$

$$t(w,x) = \frac{c_x(w)}{c_x^\star}$$

2. Define inverse document frequency of word w:

$$id(w) = \log \frac{|X|}{1 + |\{x \in X \mid w \in x\}|}$$

Total # of documents

that contain word w

3. Use combined measure for each word and document:

$$tid(w,x) = t(w,x) \times id(w)$$

Inverse Document Frequency

- We want to emphasize ***rare words*** over common ones:

$$id(w) = \log \frac{|X|}{1 + |\{x \in X \mid w \in x\}|}$$

$$tid(w,x) = t(w,x) \times id(w)$$

- $id(w)$ goes to 0 as the word w becomes more common
- $tid(w,x)$ is highest when w occurs ***often*** in document x, but is ***rare overall*** in the full document set

An Example

- The inverse document frequency of word w:

$$id(w) = \log \frac{|X|}{1 + |\{x \in X \mid w \in x\}|}$$

- Suppose we have 1,000 documents ($|X| = 1000$), and the word *the* occurs in every single one of them:

$$id(the) = \log \frac{1000}{1001} \approx -0.001442$$

- Conversely, if the word *banana* only appears in 10 of them:

$$id(banana) = \log \frac{1000}{10} \approx 6.644$$

- Thus, when calculating normalized word-counts, *banana* gets treated as being about 4,600 times more important than *the*!
  - If we threshold $id(w)$ to a minimum of 0 (never negative) we then ***completely ignore*** words that are in every document

Distances between Words

- Given the threshold on the inverse document frequency, the distance between two documents is now ***proportional*** to that measure:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{|W|} (tid(w_k, x_i) - tid(w_k, x_j))^2}$$

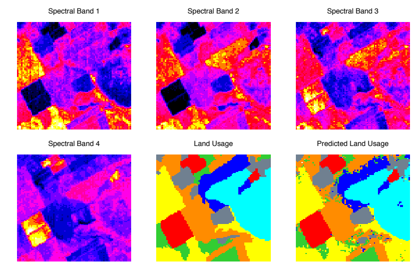
$$= \sqrt{\sum_{k=1}^{|W|} ([t(w_k, x_i) \times id(w_k)] - [t(w_k, x_j) \times id(w_k)])^2}$$

$$= \sqrt{\sum_{k=1}^{|W|} (id(w_k) \times [t(w_k, x_i) - t(w_k, x_j)])^2}$$

- Our KD-Tree can now efficiently find similar documents based upon this metric
- Mathematically, words for which frequency $id(w) = 0$ have no effect on the distance
  - Obviously, in implementing this we can simply ***remove*** those words from word-set W in the first place to skip useless clock-cycles…

Nearest-Neighbor Clustering for Image Classification

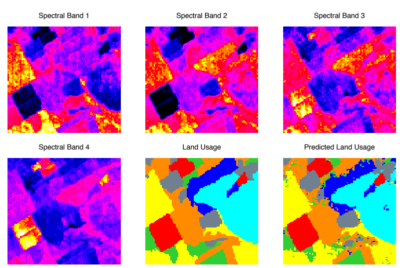
Image source: Hastie, et al., *Elements of Statistical Learning* (Springer, 2017)



- The STATLOG project (Michie et al., 1994): given satellite imagery of land, predict its agricultural use for mapping purposes
- Training set: sets of images in 4 spectral bands, with actual use of land (7 soil/crop categories) based upon manual survey

Nearest-Neighbor Clustering for Image Classification

Image source: Hastie, et al., *Elements of Statistical Learning* (Springer, 2017)



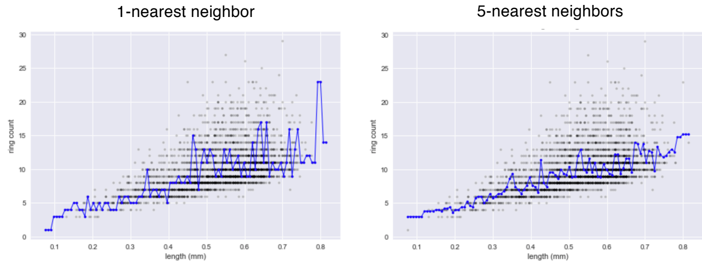
N	N	N
N	X	N
N	N	N

- To predict the usage for a given pixel in a new image:
  1. In each band, get value of a pixel and 8 adjacent, for (4 x 9) = 36 features
  2. Find the 5 nearest neighbors of that feature-vector in labeled training set
  3. Assign the land use class of the majority of those 5 neighbors
- Achieved test error of 9.5% with a very simple algorithm

Nearest-Neighbor Regression

- Given a data-set of various features of abalone (sex, size, weight, etc.), a regression classifier predicts shellfish age
- A training set of measurements, with real age determined by counting rings in the abalone shell, is analyzed and grouped into nearest neighbor units
- A predictor for new data is generated according to the average age value of neighbors

Nearest-Neighbor Regression



- Predictions for 100 points, given regression on shell length and age
- With one-nearest neighbor (left), the result has higher variability and predictions are noisier
- With five-nearest neighbors (right), results are smoothed out over multiple data-points

This Week & Next

- **Today**: Nearest Neighbors, Clustering and Regression
- **Next**: Support Vector Machines
- **Readings**: Linked from class website schedule page.
- **Homework 02**: due Thursday, 03 October, 9:00 AM
- **Homework 03**: due Wednesday, 16 October, 9:00 AM
- **Office Hours**: 237 Halligan, Tuesday, 11:00 AM – 1:00 PM
  - TA hours can be found on class website as well