Machine learning lecture slides

Nearest neighbor classification

Outline

- Optical character recognition (OCR) example
- Nearest neighbor rule
- Error rate, test error rate
- k-nearest neighbor rule
-
-
-

Example: OCR for digits

- Goal: Automatically label images of handwritten digits
- Possible labels are $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- Start with a large collection of already-labeled images
  - $D := \{(x_1, y_1), \ldots, (x_n, y_n)\}$

ng

Figure 1: Some images of handwritten digits from

Nearest neighbor (NN) classifier

- *Nearest neighbor (NN) classifier* NN_D:
  - Represented using collection of labeled examples $D := ((x_1, y_1), \ldots, (x_n, y_n))$, plus a snippet of code
- Input: x

r)

Naïve distance between images of handwritten digits (1)

- Treat (grayscale) images as vectors in Euclidean space $\mathbb{R}^d$
  - $d = 28^2 = 784$
  - Generalizes physical 3-dimensional space
- Each point $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$ is a vector of d real numbers

Figure 2: Grayscale pixel representation of an image of a handwritten “4”

Naïve distance between images of handwritten digits (2)

- Why use this for images?

- Why not use this for images?

Recap: OCR via NN

- What is the core prediction problem?

- What *features* (i.e., predictive variables) are available?

-

- Is there enough information (“training da relationship between the features and lab

- What are the modeling assumptions?

- Is high-accuracy prediction a useful goal for the application?

Error rate

- *Error rate* (on a collection of labeled examples S)
  - Fraction of labeled examples in S that have incorrect label prediction from $\hat{f}$
  - Written as $\mathrm{err}(\hat{f}, S)$
  - (Often, the word "rate" is omitted)
-

Test error rate (1)

- Better evaluation: *test error rate*
  - Train/test split, $S \cap T = \emptyset$
    - S is *training data*, T is *test data*
  - Classifier $\hat{f}$ only based on S

-
    - What is the test error rate of uniformly r
    - What is the test error rate of a constant p

Test error rate (2)

- Why is test error rate meaningful?

- What are the drawbacks of evaluation via te

Figu

Figu

Assignment Project Exam Help

Figu https://eduassistpro.github.i

Add WeChat edu_assist_pr

Assignment Project Exam Help

Figu https://eduassistpro.github.i

Add WeChat edu_assist_pr

More on the modeling assumptions

- Modeling assumption: Nearby images are more likely to have the same label than different labels.
  - This is an assumption about the choice of distance function
  - In our OCR example, this is an assumption about the choice of features

- What are the kinds of errors made by NN_S?

Figu

Figure 8: Three nearest neighbors of the test example (8,2,2)

- *k-nearest neighbor (k-NN) classifier* $\mathrm{NN}_{k,D}$
- Input: x
  - Find the k nearest neighbors of x in D
  - Return the plurality of the corresponding labels
-

- Smaller k: smaller training error rate
- Larger k: higher training error rate, but predictions more "stable" due to voting
- On OCR data: lowest test error rate achieved at $k = 3$

					9
					0.0341

- k is a *hyperparameter* of k-NN
- How to choose hyperparameters?
  - Bad idea: Choosing k that yields lowest training error rate (degenerate choice: $k = 1$)
  - Better idea: Simulate train/test split on the training data
- :

$$V_k := \mathrm{err}(\mathrm{N}$$

  - Let $\hat{k}$ be the value of k for which
  - Classifier to use is $\mathrm{NN}_{\hat{k},S}$

- Specialize to input types
  - Edit distance for strings
  - Shape distance for images
  - Time warping distance for audio waveforms

- Generic distances for vectors of real numbers
  - ℓ_p distances

$$\|x - z\|_p = \left(\sum_{j=1}^{d} |x_j - z_j|^p \right)^{1/p}.$$

- Distance functions for images of handwritten digits

distance	ℓ_2	ℓ_3	tangent	shape
test error rate	0.0309	0.0283	0.0110	0.0063

- When using numerical *features* (arranged in a vector from $\mathbb{R}^d$):
  - Scale of features matters
  - Noisy features can ruin NN
    - E.g., consider what happens in OCR example if you have another 10000 additional features that are pure "noise"

the

- "Curse of dimension"
  - Weird effects in $\mathbb{R}^d$ for large d
  - Can find $2^{\Omega(d)}$ points that are approximately equidistant

Computation for NN

- Brute force search: $\Theta(dn)$ time for each prediction (using Euclidean distance in $\mathbb{R}^d$)
- Clever data structures: "improve" to $2^d \log(n)$ time
- Approximate nearest neighbors: sub-linear time to get