Nearest neighbor classification

Instructor: Taylor Berg-Kirkpatrick
Slides: Sanjoy Dasgupta
Office hours

Kishore Venkatswammy: Tues 10-11am (CSE 3rd floor, lobby near the Kitchenette) kvenkats@eng.ucsd.edu
Weiqi (Ricky) Peng: Thurs 1:30-2:30pm (CSE B270A) wep012@ucsd.edu
Rishikesh (Rish) Vaishnav: Mon 6-7pm (B215) rvaishna@eng.ucsd.edu
Harsh Jhamtani: Thurs 10-11am (CSE 3rd floor, lobby near the Kitchenette) jharsh@cs.cmu.edu
Yaobang (Cyrus) Deng: Tues 4-5pm (B270A) yad025@ucsd.edu
The problem we’ll solve today

Given an image of a handwritten digit, say which digit it is.

3

⇒ 3
The problem we’ll solve today

Given an image of a handwritten digit, say which digit it is.

3

⇒

3

Some more examples:

0 1 2 3 4

5 6 7 8 9
The machine learning approach

Assemble a data set:

The MNIST data set of handwritten digits:

- **Training set** of 60,000 images and their labels.
- **Test set** of 10,000 images and their labels.

And let the machine figure out the underlying patterns.
Nearest neighbor classification

Training images $x^{(1)}$, $x^{(2)}$, $x^{(3)}$, $x^{(60000)}$

Labels $y^{(1)}$, $y^{(2)}$, $y^{(3)}$, $y^{(60000)}$ are numbers in the range $0 – 9$
Nearest neighbor classification

Training images $x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(60000)}$
Labels $y^{(1)}, y^{(2)}, y^{(3)}, \ldots, y^{(60000)}$ are numbers in the range 0 – 9

How to classify a new image $x$?
Nearest neighbor classification

Training images $x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(60000)}$

Labels $y^{(1)}, y^{(2)}, y^{(3)}, \ldots, y^{(60000)}$ are numbers in the range $0 - 9$

How to classify a new image $x$?

- Find its nearest neighbor amongst the $x^{(i)}$
- Return $y^{(i)}$
The data space

How to measure the distance between images?

MNIST images:
- Size $28 \times 28$ (total: 784 pixels)
- Each pixel is grayscale: 0-255
The data space

How to measure the distance between images?

MNIST images:
- Size $28 \times 28$ (total: 784 pixels)
- Each pixel is grayscale: 0-255

Stretch each image into a vector with 784 coordinates:

- Data space $\mathcal{X} = \mathbb{R}^{784}$
- Label space $\mathcal{Y} = \{0, 1, \ldots, 9\}$
The distance function

Remember Euclidean distance in two dimensions?

\[ z = (3, 5) \]

\[ x = (1, 2) \]
Euclidean distance in higher dimension

Euclidean distance between 784-dimensional vectors $x, z$ is

$$\|x - z\| = \sqrt{\sum_{i=1}^{784} (x_i - z_i)^2}$$

Here $x_i$ is the $i$th coordinate of $x$. 
Nearest neighbor classification

Training images $x^{(1)}, \ldots, x^{(60000)}$, labels $y^{(1)}, \ldots, y^{(60000)}$

To classify a new image $x$:

- Find its nearest neighbor amongst the $x^{(i)}$
- **using Euclidean distance in $\mathbb{R}^{784}$**
- Return $y^{(i)}$
Nearest neighbor classification

Training images $x^{(1)}, \ldots, x^{(60000)}$, labels $y^{(1)}, \ldots, y^{(60000)}$

To classify a new image $x$:

- Find its nearest neighbor amongst the $x^{(i)}$ using Euclidean distance in $\mathbb{R}^{784}$
- Return $y^{(i)}$

How accurate is this classifier?
Accuracy of nearest neighbor on MNIST

Training set of 60,000 points.

- What is the error rate on training points?
  
  Zero.

In general, training error is an overly optimistic predictor of future performance.

A better gauge: separate test set of 10,000 points.

Test error = fraction of test points incorrectly classified.

- What test error would we expect for a random classifier?
  
  (One that picks a label 0−9 at random?)
  
  90%.

Test error of nearest neighbor: 3.09%.
Accuracy of nearest neighbor on MNIST

Training set of 60,000 points.

- What is the error rate on training points? Zero.
  In general, training error is an overly optimistic predictor of future performance.

- A better gauge: separate test set of 10,000 points. Test error = fraction of test points incorrectly classified.
  What test error would we expect for a random classifier (One that picks a label 0−9 at random?) 90%.
  Test error of nearest neighbor: 3.09%.
Accuracy of nearest neighbor on MNIST

Training set of 60,000 points.

- What is the error rate on training points? **Zero.**
  In general, **training error** is an overly optimistic predictor of future performance.

- A better gauge: separate test set of 10,000 points.
  **Test error** = fraction of test points incorrectly classified.
Training set of 60,000 points.

- What is the error rate on training points? **Zero.**
  In general, **training error** is an overly optimistic predictor of future performance.

- A better gauge: separate test set of 10,000 points.
  **Test error** = fraction of test points incorrectly classified.

- What test error would we expect for a *random classifier*?
  (One that picks a label 0 – 9 at random?)
Accuracy of nearest neighbor on MNIST

Training set of 60,000 points.

- What is the error rate on training points? **Zero.**
  In general, **training error** is an overly optimistic predictor of future performance.

- A better gauge: separate test set of 10,000 points.
  **Test error** = fraction of test points incorrectly classified.

- What test error would we expect for a **random classifier**?
  (One that picks a label 0 – 9 at random?) **90%**.
Accuracy of nearest neighbor on MNIST

Training set of 60,000 points.

- What is the error rate on training points? **Zero.**
  In general, **training error** is an overly optimistic predictor of future performance.

- A better gauge: separate test set of 10,000 points.
  **Test error** = fraction of test points incorrectly classified.

- What test error would we expect for a *random classifier*?
  (One that picks a label 0 – 9 at random?) **90%**.

- Test error of nearest neighbor: **3.09%**.
Examples of errors

Test set of 10,000 points:
- 309 are misclassified
- Error rate 3.09%

Examples of errors:

<table>
<thead>
<tr>
<th>Query</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>
Examples of errors

Test set of 10,000 points:
- 309 are misclassified
- Error rate 3.09%

Examples of errors:

<table>
<thead>
<tr>
<th>Query</th>
<th>0</th>
<th>2</th>
<th>5</th>
<th>8</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN</td>
<td>4</td>
<td>0</td>
<td>8</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

Ideas for improvement: (1) $k$-NN (2) better distance function.
**K-nearest neighbor classification**

To classify a new point:

- Find the $k$ nearest neighbors in the training set.
- Return the most common label amongst them.

---

**MNIST:**

<table>
<thead>
<tr>
<th>$k$</th>
<th>Test error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.09</td>
</tr>
<tr>
<td>3</td>
<td>2.94</td>
</tr>
<tr>
<td>5</td>
<td>3.13</td>
</tr>
<tr>
<td>7</td>
<td>3.10</td>
</tr>
<tr>
<td>9</td>
<td>3.43</td>
</tr>
<tr>
<td>11</td>
<td>3.34</td>
</tr>
</tbody>
</table>

In real life, there's no test set. How to decide which $k$ is best?
To classify a new point:

- Find the $k$ nearest neighbors in the training set.
- Return the most common label amongst them.

**MNIST:**

<table>
<thead>
<tr>
<th>$k$</th>
<th>1</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test error (%)</td>
<td>3.09</td>
<td>2.94</td>
<td>3.13</td>
<td>3.10</td>
<td>3.43</td>
<td>3.34</td>
</tr>
</tbody>
</table>
**K-nearest neighbor classification**

To classify a new point:

- Find the *k* nearest neighbors in the training set.
- Return the most common label amongst them.

<table>
<thead>
<tr>
<th>MNIST:</th>
<th>(k)</th>
<th>1</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test error (%)</td>
<td>3.09</td>
<td>2.94</td>
<td>3.13</td>
<td>3.10</td>
<td>3.43</td>
<td>3.34</td>
<td></td>
</tr>
</tbody>
</table>

In real life, there’s no test set. How to decide which *k* is best?
Cross-validation

How to estimate the error of $k$-NN for a particular $k$?

10-fold cross-validation

• Divide the training set into 10 equal pieces.

Training set (call it $S$): 60,000 points

Call the pieces $S_1, S_2, \ldots, S_{10}$: 6,000 points each.

• For each piece $S_i$:

• Classify each point in $S_i$ using $k$-NN with training set $S - S_i$.

• Let $\epsilon_i$ = fraction of $S_i$ that is incorrectly classified.

• Take the average of these 10 numbers:

[estimated error with $k$-NN] = $\epsilon_1 + \cdots + \epsilon_{10}$
Cross-validation

How to estimate the error of $k$-NN for a particular $k$?

10-fold cross-validation

• Divide the training set into 10 equal pieces.
  Training set (call it $S$): 60,000 points
  Call the pieces $S_1, S_2, \ldots, S_{10}$: 6,000 points each.

• For each piece $S_i$:
  • Classify each point in $S_i$ using $k$-NN with training set $S - S_i$
  • Let $\epsilon_i =$ fraction of $S_i$ that is incorrectly classified

• Take the average of these 10 numbers:

  \[
  \text{estimated error with } k\text{-NN} = \frac{\epsilon_1 + \cdots + \epsilon_{10}}{10}
  \]
Another improvement: better distance functions

The Euclidean ($\ell_2$) distance between these two images is very high!
Another improvement: better distance functions

The Euclidean ($\ell_2$) distance between these two images is very high!

Much better idea: distance measures that are invariant under:

- Small translations and rotations. e.g. tangent distance.
- A broader family of natural deformations. e.g. shape context.
Another improvement: better distance functions

The Euclidean ($\ell_2$) distance between these two images is very high!

Much better idea: distance measures that are invariant under:

- Small translations and rotations. e.g. tangent distance.
- A broader family of natural deformations. e.g. shape context.

Test error rates: \[
\begin{array}{ccc}
\ell_2 & \text{tangent distance} & \text{shape context} \\
3.09 & 1.10 & 0.63 \\
\end{array}
\]
Related problem: feature selection

Feature selection/reweighting is part of picking a distance function. And, one noisy feature can wreak havoc with nearest neighbor!
Related problem: feature selection

Feature selection/reweighting is part of picking a distance function. And, one noisy feature can wreak havoc with nearest neighbor!
Algorithmic issue: speeding up NN search

Naive search takes time $O(n)$ for training set of size $n$: slow!

Locality sensitive hashing
Ball trees
K-d trees

These are part of standard Python libraries for NN, and help a lot.
Algorithmic issue: speeding up NN search

Naive search takes time $O(n)$ for training set of size $n$: slow!

Luckily there are data structures for speeding up nearest neighbor search, like:

1. Locality sensitive hashing
2. Ball trees
3. $K$-d trees

These are part of standard Python libraries for NN, and help a lot.
Postscript:
Useful distance functions for machine learning
Measuring distance in $\mathbb{R}^m$

Usual choice: **Euclidean distance**:

$$\|x - z\|_2 = \sqrt{\sum_{i=1}^{m} (x_i - z_i)^2}.$$
Measuring distance in $\mathbb{R}^m$

Usual choice: **Euclidean distance**:

$$\|x - z\|_2 = \sqrt{\sum_{i=1}^{m} (x_i - z_i)^2}.$$ 

For $p \geq 1$, here is $\ell_p$ **distance**:

$$\|x - z\|_p = \left(\sum_{i=1}^{m} |x_i - z_i|^p\right)^{1/p}$$

- $p = 2$: Euclidean distance
- $\ell_1$ distance: $\|x - z\|_1 = \sum_{i=1}^{m} |x_i - z_i|$ 
- $\ell_\infty$ distance: $\|x - z\|_\infty = \max_i |x_i - z_i|$
Example 1

Consider the all-ones vector \((1, 1, \ldots, 1)\) in \(\mathbb{R}^d\). What are its \(\ell_2\), \(\ell_1\), and \(\ell_\infty\) length?
Example 2

In $\mathbb{R}^2$, draw all points with:

1. $\ell_2$ length 1
2. $\ell_1$ length 1
3. $\ell_\infty$ length 1
Let $\mathcal{X}$ be the space in which data lie.

A distance function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a **metric** if it satisfies these properties:

- $d(x, y) \geq 0$ (nonnegativity)
- $d(x, y) = 0$ if and only if $x = y$
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)
Example 1

$\mathcal{X} = \mathbb{R}^m$ and $d(x, y) = \|x - y\|_p$

Check:

- $d(x, y) \geq 0$ (nonnegativity)
- $d(x, y) = 0$ if and only if $x = y \; \forall x, y \in \mathbb{R}$
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)
Example 2

$\mathcal{X} = \{\text{strings over some alphabet}\}$ and $d =$ edit distance

Check:

- $d(x, y) \geq 0$ (nonnegativity)
- $d(x, y) = 0$ if and only if $x = y$
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)
A non-metric distance function

Let $p, q$ be probability distributions on some set $\mathcal{X}$.

The Kullback-Leibler divergence or relative entropy between $p, q$ is:

$$d(p, q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}.$$