Unstructured Data Analysis

Lecture 4: Visualizing high-dimensional data
Imagine we had hundreds of these.

How to visualize these for comparison?

Using our earlier analysis:
Compare pairs of food items across locations (e.g., scatter plot of cheese vs cereals consumption)

But unclear how to compare the locations (England, Wales, Scotland, N. Ireland)!

Source: http://setosa.io/ev/principal-component-analysis/
The issue is that as humans we can only really visualize up to 3 dimensions easily.

Goal: Somehow reduce the dimensionality of the data preferably to 1, 2, or 3
Principal Component Analysis (PCA)

How to project 2D data down to 1D?

Principal Component Analysis (PCA)

How to project 2D data down to 1D?

Simplest thing to try: flatten to one of the red axes
Principal Component Analysis (PCA)

How to project 2D data down to 1D?

Simplest thing to try: flatten to one of the red axes
(We could of course flatten to the other red axis)
Principal Component Analysis (PCA)

How to project 2D data down to 1D?
Principal Component Analysis (PCA)

How to project 2D data down to 1D?
Principal Component Analysis (PCA)

How to project 2D data down to 1D?

But notice that most of the variability in the data is not aligned with the red axes!

Most variability is along this green direction

Rotate!
Principal Component Analysis (PCA)

How to project 2D data down to 1D?

Most variability is along this green direction.
Principal Component Analysis (PCA)

How to project 2D data down to 1D?

Most variability is along this green direction.

The idea of PCA actually works for 2D → 2D as well (and just involves rotating, and not “flattening” the data).
Principal Component Analysis (PCA)

How to project 2D data down to 1D?
How to rotate 2D data so 1st axis has most variance

The idea of PCA actually works for 2D → 2D as well (and just involves rotating, and not “flattening” the data)

2nd green axis chosen to be 90° (“orthogonal”) from first green axis
Principal Component Analysis (PCA)

- Finds top $k$ orthogonal directions that explain the most variance in the data
  - 1st component: explains most variance along 1 dimension
  - 2nd component: explains most of remaining variance along next dimension that is orthogonal to 1st dimension
  - ...

- “Flatten” data to the top $k$ dimensions to get lower dimensional representation (if $k <$ original dimension)
Principal Component Analysis (PCA)

3D example from:
http://setosa.io/ev/principal-component-analysis/
Principal Component Analysis (PCA)

Demo
PCA reorients data so axes explain variance in “decreasing order”
→ can “flatten” *(project)* data onto a few axes that captures most variance
2D Swiss Roll

PCA would just flatten this thing and lose the information that the data actually lives on a 1D line that has been curved!
PCA would squash down this Swiss roll (like stepping on it from the top) mixing the red & white parts.
2D Swiss Roll
2D Swiss Roll
2D Swiss Roll
2D Swiss Roll
2D Swiss Roll
2D Swiss Roll

This is the desired result
Manifold Learning

- Nonlinear dimensionality reduction (in contrast to PCA which is linear)
- Find low-dimensional “manifold” that the data live on

Basic idea of a manifold:

1. Zoom in on any point (say, $x$)
2. The points near $x$ look like they’re in a lower-dimensional Euclidean space (e.g., a 2D plane in Swiss roll)
Do Data Actually Live on Manifolds?

Do Data Actually Live on Manifolds?

Do Data Actually Live on Manifolds?

Manifold Learning with Isomap

Step 1: For each point, find its nearest neighbors, and build a road ("edge") between them

(e.g., find closest 2 neighbors per point and add edges to them)

Step 2: Compute shortest distance from each point to every other point where you’re only allowed to travel on the roads

Step 3: It turns out that given all the distances between pairs of points, we can compute what the low-dimensional points should be (the algorithm for this is called multidimensional scaling)
**Isomap Calculation Example**

In **orange**: road lengths

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Shortest distances between every point to every other point where we are only allowed to travel along the roads
Isomap Calculation Example

In orange: road lengths

- 2 nearest neighbors of A: B, C
- 2 nearest neighbors of B: A, C
- 2 nearest neighbors of C: B, D
- 2 nearest neighbors of D: C, E
- 2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Isomap Calculation Example

In orange: road lengths

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td>0</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>
Isomap Calculation Example

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
**Isomap Calculation Example**

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors).

Shortest distances between every point to every other point where we are only allowed to travel along the roads:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td>0</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Isomap Calculation Example

In orange: road lengths

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Isomap Calculation Example

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>5</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Isomap Calculation Example

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Isomap Calculation Example

2 nearest neighbors of A: B, C
2 nearest neighbors of B: A, C
2 nearest neighbors of C: B, D
2 nearest neighbors of D: C, E
2 nearest neighbors of E: C, D

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>5</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Isomap Calculation Example

In **orange**: road lengths

Build "symmetric 2-NN" graph (add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point *where we are only allowed to travel along the roads*

## Shortest Distances

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>13</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>16</td>
<td>13</td>
<td>8</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>
Isomap Calculation Example

2 nearest neighbors of A: B, C
2 nearest neighbors of B: C, A
2 nearest neighbors of C: D, B
2 nearest neighbors of D: E, C
2 nearest neighbors of E: D, C

Build "symmetric 2-NN" graph  
(add edges for each point to its 2 nearest neighbors)

Shortest distances between every point to every other point where we are only allowed to travel along the roads

This matrix gets fed into multidimensional scaling to get 1D version of A, B, C, D, E

The solution is not unique!

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>13</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>16</td>
<td>13</td>
<td>8</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>
Isomap

Original high-dim. data

Build k-NN graph, computed shortest distances

Distance table (for high-dim. points)

Distance table (for low-dim. points)

Make these two as close as possible (Euclidean dist)

Low-dim. data

Compute Euclidean distances between all pairs of low-dimensional points