Teaching Demonstration

The curse of dimensionality and clustering algorithms

David Varodayyan
varodayyan@stanford.edu
About this lecture

• Adapted from my lecture about $\frac{3}{4}$ way through my UIUC course
  
  *CS 361: Probability and Statistics for Computer Science*

• Topics covered so far include
  
  • Visualizing data
  • Probability
  • Principal components analysis
  • Classification algorithms

• Today’s slides and python notebook:
  
Today’s plan

• Recap
  • Visualizing data
  • Dimensionality reduction with PCA

• The curse of dimensionality

• The clustering problem

• $k$-means algorithm
The “Moneyball” baseball statistics dataset

<table>
<thead>
<tr>
<th>Team</th>
<th>League</th>
<th>Year</th>
<th>Runs Scored</th>
<th>Runs Against</th>
<th>Wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ARI</td>
<td>NL</td>
<td>2012</td>
<td>734</td>
<td>688</td>
</tr>
<tr>
<td>1</td>
<td>ATL</td>
<td>NL</td>
<td>2012</td>
<td>700</td>
<td>600</td>
</tr>
<tr>
<td>2</td>
<td>BAL</td>
<td>AL</td>
<td>2012</td>
<td>712</td>
<td>705</td>
</tr>
<tr>
<td>3</td>
<td>BOS</td>
<td>AL</td>
<td>2012</td>
<td>734</td>
<td>806</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1228</td>
<td>PIT</td>
<td>NL</td>
<td>1962</td>
<td>706</td>
<td>626</td>
</tr>
<tr>
<td>1229</td>
<td>SFG</td>
<td>NL</td>
<td>1962</td>
<td>878</td>
<td>690</td>
</tr>
<tr>
<td>1230</td>
<td>STL</td>
<td>NL</td>
<td>1962</td>
<td>774</td>
<td>664</td>
</tr>
<tr>
<td>1231</td>
<td>WSA</td>
<td>AL</td>
<td>1962</td>
<td>599</td>
<td>716</td>
</tr>
</tbody>
</table>
Visualizing where the data is with histograms
Visualizing where the data is with scatter plots
The Japanese facial expression dataset

• The dataset consists of \( n = 213 \) images of Japanese women

• Each image is grayscale and has \( 64 \times 64 \) resolution

• We can treat each image as a vector of dimension \( d = 4096 \)
Dimensionality reduction with PCA

Original image

Number of principal components

mean 1 5 10 20 50 100

Reconstructions

Error
The curse of dimensionality

The data isn’t where you think it is
How much volume of a cubic orange is peel?
What about a $d$-dimensional “cubic” orange?

- Total amount of orange: $2^d$
- Amount of fruity part: $2^d (1 - \epsilon)^d$
- Fraction that is peel: $1 - (1 - \epsilon)^d$

A high-dimensional orange is virtually all peel!
The curse of dimensionality

• If a dataset is uniformly distributed in a high-dimensional cube (or some other shape), the vast majority of data is far from the mean.

• We can also prove that the average distance between items grows with increasing dimensions.

• A $d$-dimensional histogram of the dataset is not very useful because
  • Most bins will be empty
  • Some bins will contain a single data point
  • Very few bins will contain more than one point

Dealing with data in high dimensions

• Collect as much data as possible

• Cluster data points together into one or more blobs

• Do PCA and/or fit a simple probability model to each blob
The clustering problem

• Given a dataset, separate the data items into clusters so that
  • Items within a cluster are close to each other
  • Items in different clusters are far from each other

• There are two problems to solve
  • Determine the number of clusters
  • Assign each item to a cluster

• Note that we are taking unlabeled data and assigning a class label to each item
Clustering approaches

• Divisive clustering
  • Treat the whole dataset as a single cluster
  • Then split the dataset recursively until you get a satisfactory clustering

• Agglomerative clustering
  • Treat each data item as its own cluster
  • Then merge clusters until you get a satisfactory clustering

• Iterative clustering (such as $k$-means)
\( k \)-means clustering algorithm

• Pick a value for \( k \), which is the number of clusters

• Select \( k \) random cluster centers

• Iterate the following two steps until convergence
  • Assign each data item to the nearest cluster center
  • Update each cluster center as the mean of the items assigned to its cluster
**k-means clustering example**

1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).

2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the **Voronoi diagram** generated by the means.

3. The **centroid** of each of the *k* clusters becomes the new mean.

4. Steps 2 and 3 are repeated until convergence has been reached.

The Iris dataset

• Famous dataset collected by botanist Edgar Anderson and popularized by statistician Ronald Fisher in 1936

• There are 4 measurements per item
  • Sepal length (cm)
  • Sepal width (cm)
  • Petal length (cm)
  • Petal width (cm)

• See today’s notebook: http://web.stanford.edu/~divad/
The dataset actually contains 3 species
$k$-means clustering results: iris

true labels

$k = 3$ clusters
Choosing the number of clusters $k$

• Given a $k$-means clustering of $N$ data items $x_i$ to $k$ cluster centers $c_j$, define the sum of square distances from each $x_i$ to its cluster center as a cost function “within cluster sum of squares”

$$\sum_{i=1}^{N} \sum_{j=1}^{k} \delta_{i,j} \|x_i - c_j\|^2$$

where

$$\delta_{i,j} = \begin{cases} 
1 & \text{if } x_i \in \text{cluster } j \\
0 & \text{if } x_i \notin \text{cluster } j 
\end{cases}$$

• Perform $k$-means clustering for many values of $k$ and find the knee in the “within cluster sum of squares”
Choosing the number of clusters $k$

Find the knee in the curve
Summary

• In high-dimensional datasets, the data isn’t where you think it is

• Collect as much data as you can, and cluster it into blobs

• There are variety of clustering techniques; $k$-means is quite effective

• Use “within cluster sum of squares” to help choose number of clusters
Final thoughts

• Clustering is usually just one part of a data pipeline and it is often good enough to get the number of clusters approximately right.

• Next lecture: the course project!
  • You will build a classifier that reads in a Fitbit accelerometer signal and tells what activity is being performed.
  • Within the classification pipeline, you will create a “pattern vocabulary” using $k$-means clustering.
Extra slides
The project: activity from accelerometer data

• The dataset consists of Fitbit-like accelerometer signals, each of which
  • Can be of arbitrary length
  • Consists of 3 dimensions (x, y, z) of data sampled at 32 Hz
  • Is labeled with one of 14 activities, such as “brushing teeth”

https://archive.ics.uci.edu/ml/datasets/Dataset+for+ADL+Recognition+with+Wrist-worn+Accelerometer

• Your task is to train a classifier to take an accelerometer signal and map it to an activity
The project: looking at the raw data
The project: building a pattern vocabulary

- Slice each signal into non-overlapping pieces of 1 second duration, which gives you pieces of size $d = 32 \times 3 = 96$

- Cluster the 96-dimensional vectors to $k$ cluster centers using scikit-learn’s $k$-means clustering algorithm

Some cluster centers, x dimension only
The project: representation and classification

• Represent each signal as a $k$-dimensional feature vector

• Train a multiclass classifier such as scikit-learn’s random forest on the training vectors

• Evaluate the classifier using the test vectors

• Improve the classifier by tuning parameters
Some variants of $k$-means clustering

- Soft assignment allows some data items to belong to multiple clusters with weights associated with each cluster
- Hierarchical $k$-means speeds up clustering for very large datasets
  - Sample the dataset and apply $k$-means with a small value of $k$
  - Assign all the data to one of the clusters
  - Subcluster each individual cluster
  - Repeat until you have a tree of clusters of your desired depth
- $k$-medioids allows clustering of data that cannot be averaged
Multivariate normal distribution

- Extension of the normal distribution to multiple dimensions

- Example: bivariate (2-dimensional) normal distribution

Multivariate normal probability density

A multivariate normal random vector $\mathbf{X}$ of dimension $d$ has density

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp \left( -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$$

where

- $\boldsymbol{\mu} = E[\mathbf{X}]$ is a $d$-dimensional vector called the mean
- $\Sigma = E[(\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^T]$ is a $d \times d$ symmetric and positive semidefinite matrix called the covariance matrix
Multivariate MLE

Given a $d$-dimensional dataset $\{x\}$ consisting of $N$ items, we can fit a multivariate normal distribution using maximum likelihood estimation

$$\hat{\mu}_{MLE} = \text{mean}(\{x\}) = \frac{\sum_i x_i}{N}$$

$$\hat{\Sigma}_{MLE} = \text{Covmat}(\{x\}) = \frac{\sum_i (x_i - \text{mean}(\{x\}))(x_i - \text{mean}(\{x\}))^T}{N}$$