Programming Machine Learning Applications

Lecture Three: Proximity and the k-Nearest-Neighbors Algorithm

Dr. Aleksandar Velkoski

Types of Datasets

Instances & Features

Interactive Workshop

Review of Lecture Two

Proximity

k-Nearest-Neighbor Classifier

Interactive Workshop

Lecture Three

Proximity

Similarity - Dissimilarity - Proximity

Similarity

- Numerical measure of how alike two features are
- Value is higher when instances are more … ?

Dissimilarity

- Numerical measure of how different two features are
- Value is higher when instances are more … ?

Proximity can refer to similarity of dissimilarity

Why is it Important?

Data Mining Tasks

- Clustering
- k-Nearest-Neighbor search, classification, and prediction
- Characterization and Discrimination
- Automatic Categorization
- Correlation Analysis

Real-world Applications

- Personalization
- Recommender Systems
- Document Categorization
- Information Retrieval
- Target Marketing

Measuring Proximity

In order to group similar items, we need a way to measure proximity

It often requires the representation of objects as feature vectors

Employee DB

ID	Gender	Age	Salary
1	F	27	19,000
2	M	51	64,000
3	M	52	100,000
4	F	33	55,000
5	M	45	45,000

Term Frequencies

	T1	T2	T3	T4	T5	T6
Doc1	0	4	0	0	0	2
Doc2	3	1	4	3	1	2
Doc3	3	0	0	0	3	0
Doc4	0	1	0	3	0	0
Doc5	2	2	2	3	1	4

Properties

For all objects A and B, $dist(A, B) >= 0$, and $dist(A, B) = dist(B, A)$

For any object A, $dist(A, A) = 0$

$dist(A, C) <= dist(A, B) + dist(B, C)$

Representation

Each object can be viewed as an n-dimensional vector, where the dimensions are the features in the data

Example (employee DB): <M, 51, 64000>
Example (documents): <3, 1, 4, 3, 1, 2>

The vector representation allows us to compare proximity between pairs of items using standard vector operations

Data Matrix and Distance Matrix

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

Nominal Features

m: # of matches, p: total # of variables

$$d(i, j) = \frac{p - m}{p}$$

Binary Features

Contingency Table

	1	0	sum
1	q	r	$q+r$
0	s	t	$s+t$
sum	$q+s$	$r+t$	p

Symmetric

$$d(i, j) = \frac{r+s}{q+r+s+t}$$

Asymmetric

$$d(i, j) = \frac{r+s}{q+r+s}$$

Numeric Features

Consider two vectors: $X = \langle x_1, x_2, \cdots, x_n \rangle$ $Y = \langle y_1, y_2, \cdots, y_n \rangle$

Manhattan Distance

$$dist(X,Y) = |x_1 - y_1| + |x_2 - y_2| + \cdots + |x_n - y_n|$$

Euclidean Distance

$$dist(X,Y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2}$$

Numeric Example

Data Matrix

point	attribute1	attribute2
x1	1	2
x2	3	5
x3	2	0
x4	4	5

Manhattan

	x1	*x2*	*x3*	*x4*
x1	0			
x2	5	0		
x3	3	6	0	
x4	6	1	7	0

Euclidean

	x1	*x2*	*x3*	*x4*
x1	0			
x2	3.61	0		
x3	2.24	5.1	0	
x4	4.24	1	5.39	0

Minkowski Distance

Minkowski Distance

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}$$

- where i = (xi1, xi2, …, xip) and j = (xj1, xj2, …, xjp) are two p-dimensional data objects, and h is the order (the distance so defined is also called L-h norm)

Note: Euclidean and Manhattan Distances are special cases of Minkowski

Vector Based Measures

In some situations, distances measures provide a skewed view of data

Dot product of vertices: $sim(X,Y) = X \bullet Y = \sum_i x_i \times y_i$

Cosine Similarity = Normalized Dot Product

$$sim(X,Y) = \frac{X \bullet Y}{\|X\| \times \|y\|} = \frac{\sum_i (x_i \times y_i)}{\sqrt{\sum_i x_i^2} \times \sqrt{\sum_i y_i^2}}$$

Example: Information Retrieval

Documents are represented as “bags of words” (vectors when used computationally)

- A vector is an array of floating point (or binary in case of bit maps)
- Has direction and magnitude
- Each vector has a place for every term in collection (most are sparse)

Document Ids

	nova	galaxy	heat	actor	film	role
A	1.0	0.5	0.3			
B	0.5	1.0				
C		1.0	0.8	0.7		
D		**0.9**	**1.0**	**0.5**		
E			1.0		1.0	
F				0.7		
G	0.5	0.7			0.9	
H		0.6		1.0	0.3	0.2
I		0.7	0.5		0.3	

a document vector

$$D_i = w_{d_{i1}}, w_{d_{i2}}, \ldots, w_{d_{it}}$$

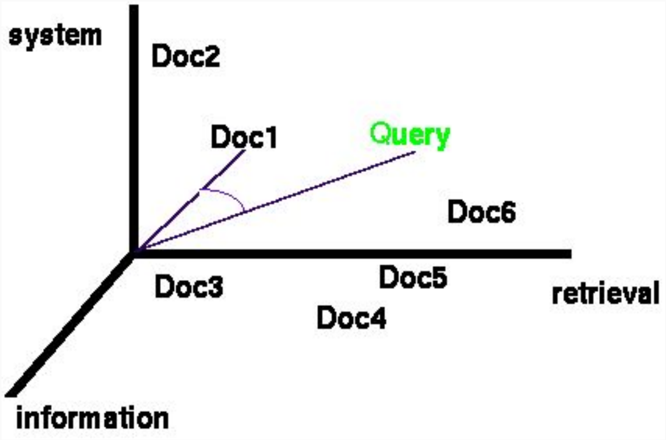
$$Q = w_{q1}, w_{q2}, \ldots, w_{qt}$$

$w = 0$ if a term is absent

Documents and Queries

Documents are represented as vectors in term space

- Typically values in each dimension correspond to the frequency of the corresponding term in the document
- Queries represented as vectors in the same vector-space
- Cosine similarity between the query and documents is often used to rank retrieved documents



Example: Document Similarities

	T1	T2	T3	T4	T5	T6	T7	T8
Doc1	0	4	0	0	0	2	1	3
Doc2	3	1	4	3	1	2	0	1
Doc3	3	0	0	0	3	0	3	0
Doc4	0	1	0	3	0	0	2	0
Doc5	2	2	2	3	1	4	0	2

Dot-Product(Doc2,Doc4) = <3,1,4,3,1,2,0,1> * <0,1,0,3,0,0,2,0>
0 + 1 + 0 + 9 + 0 + 0 + 0 + 0 = 10

Norm (Doc2) = SQRT(9+1+16+9+1+4+0+1) = 6.4
Norm (Doc4) = SQRT(0+1+0+9+0+0+4+0) = 3.74

Cosine(Doc2, Doc4) = 10 / (6.4 * 3.74) = 0.42

Correlation as Similarity

In cases where there could be high mean variance across objects, Pearson is used

$$corr(x, y) = \frac{\mathrm{cov}(x, y)}{stdev(x) \cdot stdev(y)}$$

Often used in recommender systems based on Collaborative Filtering

Distance-Based Classification

Classify new instances based on similarity to instances we've seen before.

Simplest form of MBR (Minimum Bounding Rectangle): Rote Learning

- Learning by memorization
- Save all previously encountered instances; given a new instance, find one from the memorized set that most closely "resembles" the new one; assign new instance to the same class as the "nearest neighbor"
- More general methods try to find "k" nearest neighbors

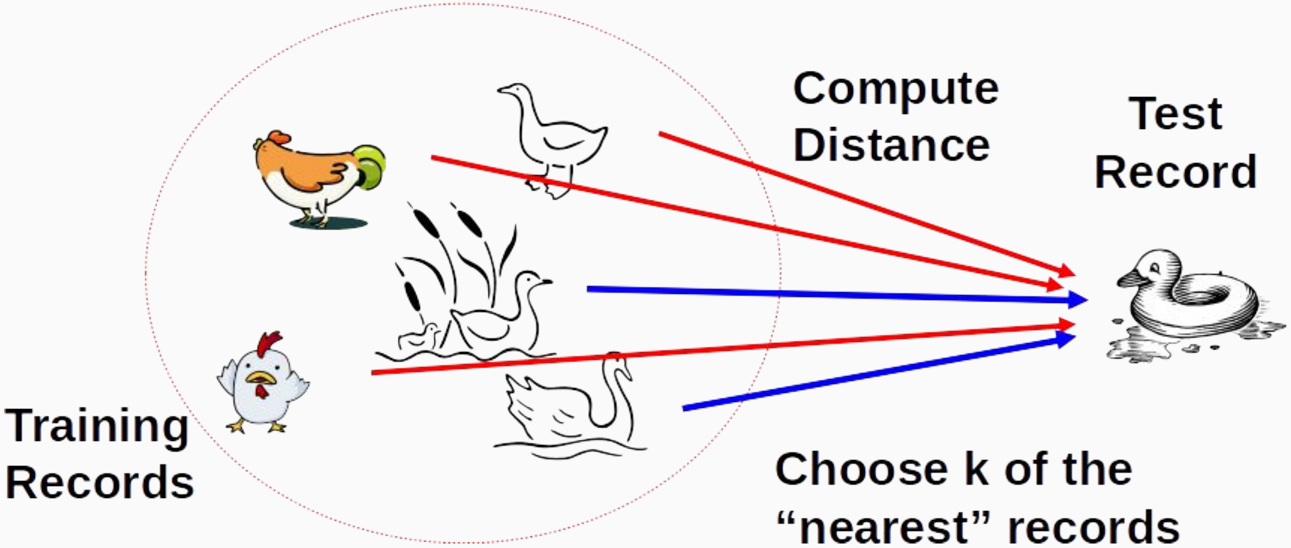
Challenges

How do we define "resembles"?

k-Nearest Neighbors

kNN Classifier

If it walks like a duck, quacks like a duck, then it's probably a duck



kNN Strategy

Given object x, find the k most similar objects to x

- The k nearest neighbors
- Variety of distance or similarity measures can be used to identify and rank neighbors
- Note that this requires comparison between x and all objects in the database

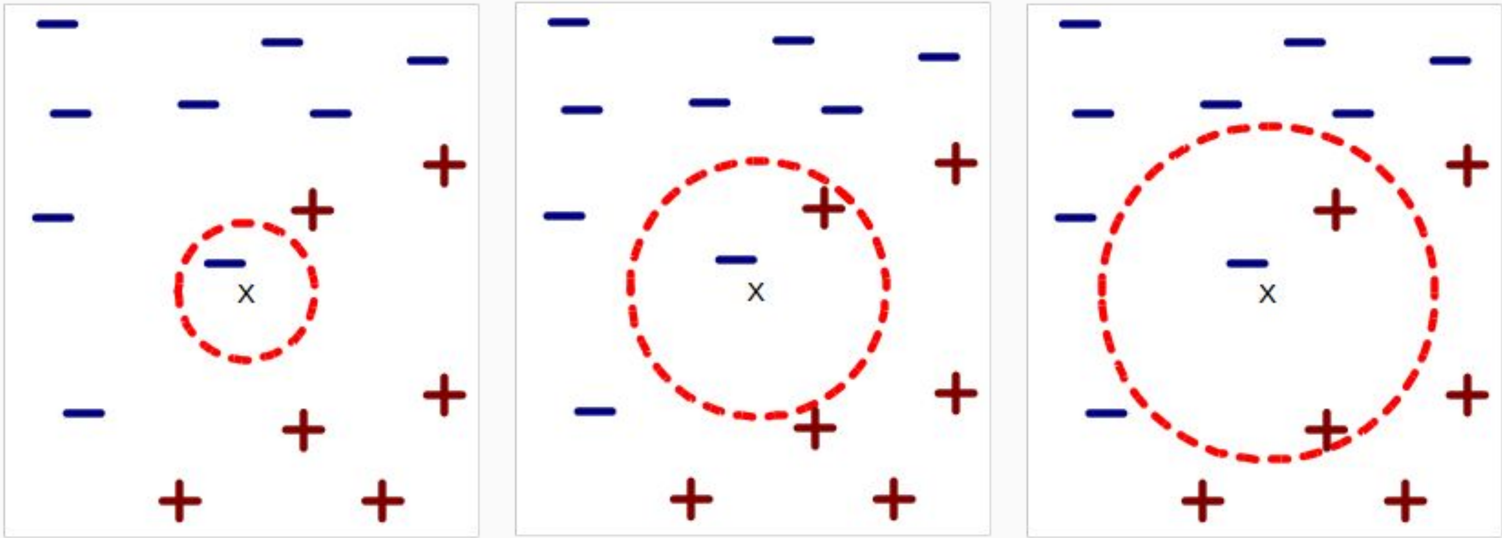
Classification

- Find the class label for each of the k neighbor
- Use a voting or weighted voting approach to determine the majority class among the neighbors (a combination function)
- Weighted voting means the closest neighbors count more
- Assign the majority class label to x

Prediction

- Identify the value of the target attribute for the k neighbors
- Return the weighted average as the predicted value of the target attribute for x

kNN Strategy



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

Combination Functions

Voting: the “democracy” approach

- poll the neighbors for the answer and use the majority vote
- the number of neighbors (k) is often taken to be odd in order to avoid ties
    - works when the number of classes is two
    - if there are more than two classes, take k to be the number of classes plus 1

Impact of k on predictions

- in general different values of k affect the outcome of classification
- we can associate a confidence level with predictions (this can be the % of neighbors that are in agreement)
- problem is that no single category may get a majority vote
- if there is strong variations in results for different choices of k, this an indication that the training set is not large enough

Voting Approach

Will a new customer respond to solicitation?

ID	Gender	Age	Salary	Respond?
1	F	27	19,000	no
2	M	51	64,000	yes
3	M	52	105,000	yes
4	F	33	55,000	yes
5	M	45	45,000	no
new	F	45	100,000	?

Using the voting method without confidence

	Neighbors	Answers	k =1	k = 2	k = 3	k = 4	k = 5
D_man	4,3,5,2,1	Y,Y,N,Y,N	yes	yes	yes	yes	yes
D_euclid	4,1,5,2,3	Y,N,N,Y,Y	yes	?	no	?	yes

Using the voting method with a confidence

	k =1	k = 2	k = 3	k = 4	k = 5
D_man	yes, 100%	yes, 100%	yes, 67%	yes, 75%	yes, 60%
D_euclid	yes, 100%	yes, 50%	no, 67%	yes, 50%	yes, 60%

Document Categorization

	T1	T2	T3	T4	T5	T6	T7	T8	Cat
DOC1	2	0	4	3	0	1	0	2	**Cat1**
DOC2	0	2	4	0	2	3	0	0	**Cat1**
DOC3	4	0	1	3	0	1	0	1	**Cat2**
DOC4	0	1	0	2	0	0	1	0	**Cat1**
DOC5	0	0	2	0	0	4	0	0	**Cat1**
DOC6	1	1	0	2	0	1	1	3	**Cat2**
DOC7	2	1	3	4	0	2	0	2	**Cat2**
DOC8	3	1	0	4	1	0	2	1	?

Document Categorization

	T1	T2	T3	T4	T5	T6	T7	T8	Norm	Sim(D8,Di)
DOC1	2	0	4	3	0	1	0	2	5.83	0.61
DOC2	0	2	4	0	2	3	0	0	5.74	0.12
DOC3	4	0	1	3	0	1	0	1	5.29	0.84
DOC4	0	1	0	2	0	0	1	0	2.45	0.79
DOC5	0	0	2	0	0	4	0	0	4.47	0.00
DOC6	1	1	0	2	0	1	1	3	4.12	0.73
DOC7	2	1	3	4	0	2	0	2	6.16	0.72
DOC8	3	1	0	4	1	0	2	1	5.66	

Sim(D8,D7) = (D8 * D7) / (Norm(D8).Norm(D7))
= (3x2+1x1+0x3+4x4+1x0+0x2+2x0+1x2) / (5.66 x 6.16)
= 25 / 34.87 = 0.72

Document Categorization

Simple voting:

Cat for DOC 8 = Cat2 with confidence 2/3 = 0.67

Weighted voting:

Cat for DOC 8 = Cat2
Confidence: (0.84 + 0.73) / (0.84 + 0.79 + 0.73) = 0.66

	T1	T2	T3	T4	T5	T6	T7	T8	Cat	Sim(D8,Di)
DOC1	2	0	4	3	0	1	0	2	**Cat1**	0.61
DOC2	0	2	4	0	2	3	0	0	**Cat1**	0.12
DOC3	4	0	1	3	0	1	0	1	**Cat2**	0.84
DOC4	0	1	0	2	0	0	1	0	**Cat1**	0.79
DOC5	0	0	2	0	0	4	0	0	**Cat1**	0.00
DOC6	1	1	0	2	0	1	1	3	**Cat2**	0.73
DOC7	2	1	3	4	0	2	0	2	**Cat2**	0.72
DOC8	3	1	0	4	1	0	2	1	5.66	

Combination Functions

Weighted Voting: not so "democratic"

- similar to voting, but the vote some neighbors counts more
- "shareholder democracy?"
- question is which neighbor's vote counts more?

How can weights be obtained?

- **Distance-based**
  - closer neighbors get higher weights
  - "value" of the vote is the inverse of the distance (may need to add a small constant)
  - the weighted sum for each class gives the combined score for that class
  - to compute confidence, need to take weighted average
- **Heuristic**
  - **weight for each neighbor is based on domain-specific characteristics of that neighbor**

Advantage of weighted voting

- introduces enough variation to prevent ties in most cases
- helps distinguish between competing neighbors

Example: Collaborative Filtering

Movie Rating SYstem

Rating Scale: 1 = “hate it”; 7 = “love it”

Historical DB of users includes ratings of movies

Karen is a new user who has rated 3 movies, but has not yet seen “Independence Day”; should we recommend it to her?

	Sally	Bob	Chris	Lynn	Karen
Star Wars	7	7	3	4	7
Jurassic Park	6	4	7	4	4
Terminator II	3	4	7	6	3
Independence Day	7	6	2	2	?

Example: Collaborative Filtering

	Star Wars	Jurassic Park	Terminator 2	Indep. Day	Average	Cosine	Distance	Euclid	Pearson
Sally	7	6	3	7	5.33	0.983	2	2.00	0.85
Bob	7	4	4	6	5.00	0.995	1	1.00	0.97
Chris	3	7	7	2	5.67	0.787	11	6.40	-0.97
Lynn	4	4	6	2	4.67	0.874	6	4.24	-0.69
Karen	7	4	3	?	4.67	1.000	0	0.00	1.00

K	Pearson
1	6
2	6.5
3	5

K is the number of nearest neighbors used in to find the average predicted ratings of Karen on Indep. Day.

Pearson(Sally, Karen) = ((7-5.33)*(7-4.67) + (6-5.33)*(4-4.67) + (3-5.33)*(3-4.67))
/ SQRT(((7-5.33)2 +(6-5.33)2 +(3-5.33)2) * ((7- 4.67)2 +(4- 4.67)2 +(3- 4.67)2)) = 0.85

Collaborative Filtering

In practice a more sophisticated approach is used to generate the predictions based on the nearest neighbors

To generate predictions for a target user a on an item i: $p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{k} (r_{u,i} - \bar{r}_u) \times sim(a,u)}{\sum_{u=1}^{k} sim(a,u)}$

- ra = mean rating for user a
- u1, …, uk are the k-nearest-neighbors to a
- ru,i = rating of user u on item I
- sim(a,u) = Pearson correlation between a and u

This is a weighted average of deviations from the neighbors' mean ratings (and closer neighbors count more)

Wrapping-up the Lecture

Questions

What is the difference between a data and distance matrix?

What is the intuition behind k-Nearest-Neighbors Classifier?