20210920 Distance and Similarity Continues

Monday, September 20, 2021 5:05 PM

Distance & Similarity

Distance & Similarity

Boston University CS 506 - Lance Galletti

Data

n data points

$$\begin{pmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1m} \\ \vdots & \ddots & \vdots & & \vdots \\ x_{i1} & \dots & x_{ij} & \dots & x_{im} \\ \vdots & & \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{nj} & \dots & x_{nm} \end{pmatrix}$$

m features

Feature Space

From our data we can generate a **feature space** of all possible values for the set of features in our data.

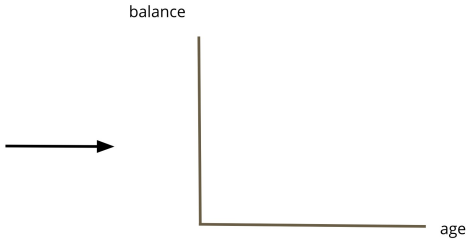
name	age	balance
Jane	25	150
John	30	100

- Starting out with unsupervised learning
- Looking for structure of dataset (not predicting)
- Understanding how other data points relate to each other
- How do we compare individual points

Feature Space

From our data we can generate a **feature space** of all possible values for the set of features in our data.

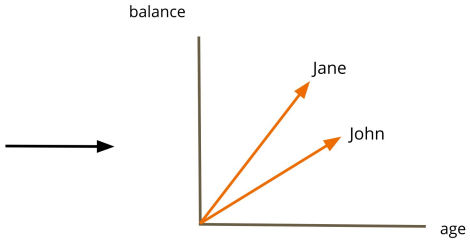
name	age	balance
Jane	25	150
John	30	100



Feature Space

From our data we can generate a **feature space** of all possible values for the set of features in our data.

name	age	balance
Jane	25	150
John	30	100



Our feature space is the Euclidean plane

Distance

In order to uncover interesting structure from our data, we need a way to **compare** data points.

A **dissimilarity function** is a function that takes two objects (data points) and returns a **large value** if these objects are **dissimilar**.

A special type of dissimilarity function is a **distance** function

- Are two data points very similar or not very similar

Distance

d is a distance function if and only if:

- d(i, j) = 0 if and only if i = j
- d(i, j) = d(j, i)
- d(i, j) ≤ d(i, k) + d(k, j)

We don't **need** a distance function to compare data points, but why would we prefer using a distance function?

- Distance has some intuitive understanding
- Triangle inequality (can't be shorter to go through intermediary point)

Minkowski Distance

For **x**, **y** points in **d**-dimensional real space

I.e. **x = [x_1 , ... , x_d]** and **y = [y_1 , ... , y_d]**

p ≥ 1

$$L_p(x, y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

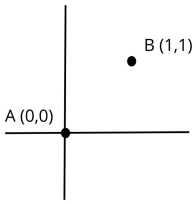
When **p** = 2 -> Euclidean Distance

When **p** = 1 -> Manhattan Distance

- Sum of each pairwise differences between coordinates

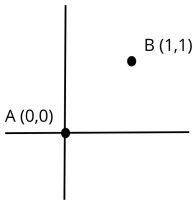
Example

d = 2



Example

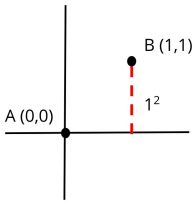
d = 2



p = 2 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

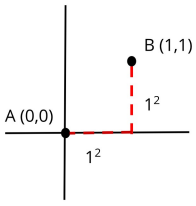
d = 2



p = 2 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

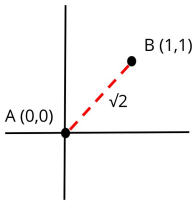
d = 2



p = 2 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

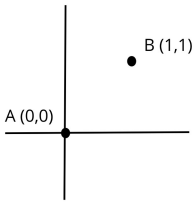
d = 2



p = 2 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

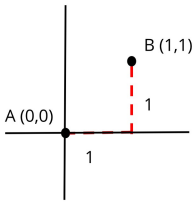
d = 2



p = 1 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

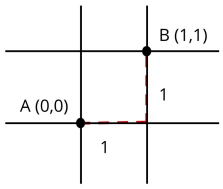
d = 2



p = 1 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

d = 2

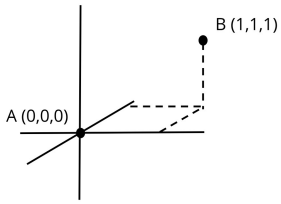


p = 1

$$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

d = 3

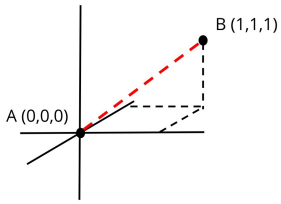


p = 2

$$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

d = 3

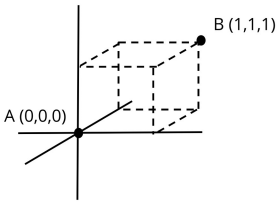


p = 2

$$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

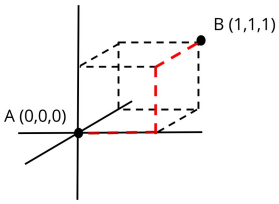
d = 3



p = 1 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example

d = 3



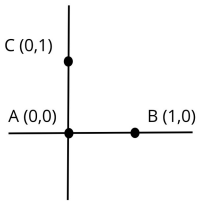
p = 1 $$L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Minkowski Distance

Is L_p a distance function when **0 < p < 1** ?

Minkowski Distance

Is L_p a distance function when $0 < p < 1$?

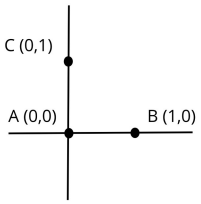


Minkowski Distance

Is L_p a distance function when $0 < p < 1$?

D(B,A) = D(A, C) = 1

D(B, C) = $2^{1/p}$

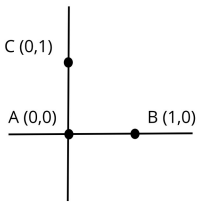


Minkowski Distance

Is L_p a distance function when $0 < p < 1$?

D(B,A) + D(A, C) = 2

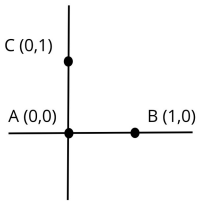
D(B, C) = $2^{1/p}$



But... if **$p < 1$** then **$1/p > 1$**

Minkowski Distance

Is L_p a distance function when **0 < p < 1** ?



D(B,A) + D(A, C) = 2

$D(B, C) = 2^{1/p}$

So **D(B, C) > D(B, A) + D(A, C)** which violates the triangle inequality

- Violates claim that it would be a distance function

Cosine Similarity

A **similarity** function is a function that takes two objects (data points) and returns a **large value** if these objects are **similar**.

$$s(x, y) = \cos(\theta)$$

where θ is the angle between **x** and **y**

- Cosine of the ANGLE between the two datapoints

Cosine Similarity

To get a corresponding **dissimilarity** function, we can usually try

$$d(x, y) = 1 / s(x, y)$$

or

$$d(x, y) = k - s(x, y) \text{ for some } k$$

Here, we can use

$$d(x, y) = 1 - s(x, y)$$

- Can get a dissimilarity function from a similarity function and vice versa
  - Often 1/z , 1-z ...

Cosine Similarity

When should you use **cosine (dis)similarity** over **euclidean distance**?

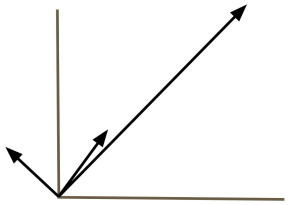
When **direction** matters more than **magnitude**

- When is Cosine similarity better?
  - Direction more important than magnitude
  - If order matters?

Cosine Similarity

When should you use **cosine (dis)similarity** over **euclidean distance**?

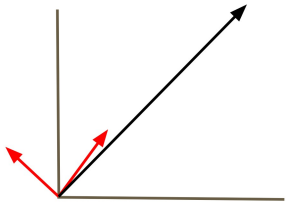
When **direction** matters more than **magnitude**



Cosine Similarity

When should you use **cosine (dis)similarity** over **euclidean distance**?

When **direction** matters more than **magnitude**



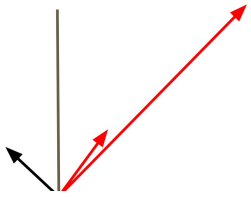
Close under
Euclidean distance

- Maybe in your dataset, it matters more that they point in the same direction than if they are physically distance
- Text documents:
  - Might be more interested if two text documents talking about the same topic RATHER than the same amount of words, same specificity of words…
  - Abstract and the paper itself are on the same topic
  - More words will be in more dimensions and will likely be very far away from the abstract spatially
- It is a quantitative method (NOT qualitative)

Cosine Similarity

When should you use **cosine (dis)similarity** over **euclidean distance**?

When **direction** matters more than **magnitude**



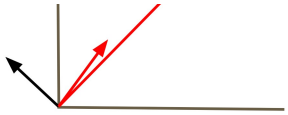
Close under Cosine

- From Frank Pacini to Everyone:

  Is there a difference between the minkowski distance and p-norm

  - It's the same thing

Close under Cosine Similarity



From Frank Pacini to Everyone:

Is there a difference between the minkowski distance and p-norm

- It's the same thing
- The norm is just the distance from the 0
- LP? norm is basically Minkowski distance

Jaccard Similarity

How similar are the following documents?

	w_1	w_2	...	w_d
x	1	0	...	1
y	1	1	...	0

- Jaccard Similarity example
- Two articles
  - Represented by word
  - Each word is an attribute
    - HW0 had 279 attributes
    - This may have 100X that
    - Numerical value will be whether word is present (1) or not (0)

Jaccard Similarity

One way is to use the Manhattan distance which will return the size of the set difference

	w_1	w_2	...	w_d
x	1	0	...	1
y	1	1	...	0

$$L_1(x, y) = \sum_{i=1}^{d} |x_i - y_i|$$

- You could us Manhattan distance
  - Returns the size of the set differences

Jaccard Similarity

One way is to use the Manhattan distance which will return the size of the set difference

	w_1	w_2	...	w_d
x	1	0	...	1
y	1	1	...	0

$$L_1(x, y) = \sum_{i=1}^{d} |x_i - y_i|$$

Will only be 1 when $x_i \neq y_i$

- 1 is exactly the set difference

Jaccard Similarity

But how can we distinguish between these two cases?

	w_1	w_2	...	w_{d-1}	w_d
x	1	1	1	0	1
y	1	1	1	1	0

Only differ on the last two words

	w_1	w_2
x	0	1
y	1	0

Completely different

- Manhattan distance will be the same for both of these cases
- Only 2 differences in a huge document is EXTREMELY similar

Jaccard Similarity

But how can we distinguish between these two cases?

	w_1	w_2	...	w_{d-1}	w_d
x	1	1	1	0	1
y	1	1	1	1	0

Only differ on the last two words

	w_1	w_2
x	0	1
y	1	0

Completely different

Both have Manhattan distance of 2

- JSIM for first one: 0.6?
- JSIM if totally different: 0 (low value, so very dissimilar)
- (Pay attention to when talking about similarity vs distance)

Jaccard Similarity

We need to account for the size of the intersection!

$$JSim(x, y) = \frac{|x \cap y|}{|x \cup y|}$$

$$JDist(x, y) = 1 - \frac{|x \cap y|}{|x \cup y|}$$

- Incorporates intersection and Union
- Size of intersection / size of Union
- Distance = 1 - (intersection/Union)

Implement these distance functions in the CS506 python package

- Challenge: On your own time:
  - Try to show that this is a distance function by demonstrating that is fulfills all 3 properties
    - With the non-trivial case being the triangle inequality
  - Can post on Piazza if you have an idea of how that works, or post notes to shared repo
- This week: Python and Git lab
- Next week: will need to implement these distance functions
  - Need to pip install pacakges
    - Package manager
    - Can install package and utilize functions people have written
  - Python package for 3rd lab
    - pip install CS506
      - Needs to be save in a specific location to do this?
  - Can create own python package if you want
  - Will also go over have to test your python package

20210920 v2

Monday, September 20, 2021 4:33 PM

Git fundamental due today
HW0 is due Wednesday
Nothing to really submit for github fundamental, just checking if you know how to submit

HW due at midnight ET

Not enough pitches to fill all the time

- Students can ask questions in the chat while the speaker is around
- Can get questions directed through prof

- Submit project preferences ASAP: hard deadline tonight or tomorrow
- If you submitted individual projects, he has made some comments in google docs so please address those
- Should be able to change preference form after submission
- May be auto-graders for HW

- How you code should count for a lot
  - He wants the means to count more than end
  - Have good quality comments, etc.
  - For HW especially
  - Commit history is not important for HW
- For Lab:
  - The PR for the first lab should not be included into the second PR
- Don't want to have a bunch of commits that are small incremental changes
  - Want to be able to amend commits
- Commit history is important in general, but not necessarily important for HW

==========

Some of second lab:

- Declares Python package
- Need to write this setup …

- tox is the pythonic way to test stuff

- Will need to implement these different functions
  - Will currently raise NotImplementedError

tox will install package and run test locally
Will see if you pass by running this simple command
Can add more test if you want

(This is next week's lab)