K-Nearest Neighbors

Social Data Mining

Grzegorz Chrupała

What is a learning example?

- We saw some types of example targets/outputs:
  - (Real) number
  - Class label
  - Sequence of labels

How about inputs?

- Textual
  - emails
  - tweets
- Visual
  - photos
  - video
  - scanned handwriting
- Audio
  - voice recordings
  - music tracks
- Physical
  - people / animals
  - plants
  - other objects

Generic algorithms require common representations

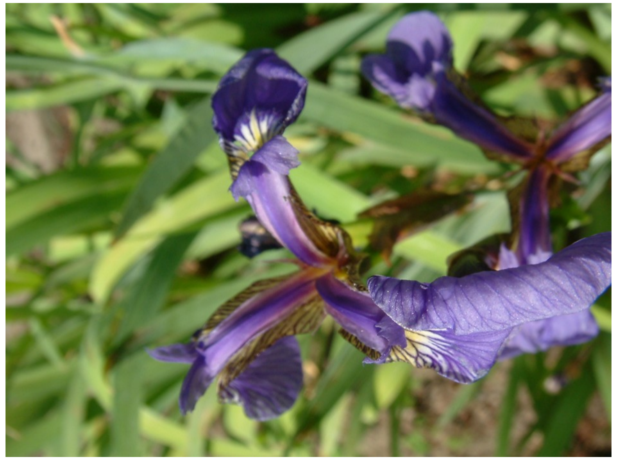
Decompose objects into FEATURES

Features

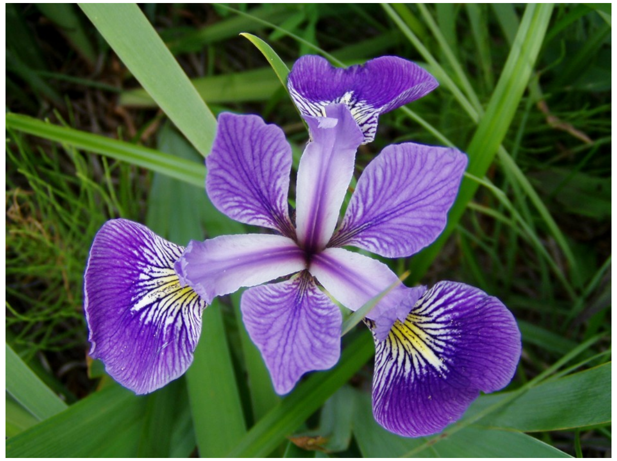
- We decompose inputs into features
- A feature is a measurable aspect of an object

- Features are often extracted before learning
- Some learning algorithms can extract features from some types of input (e.g. images or text)

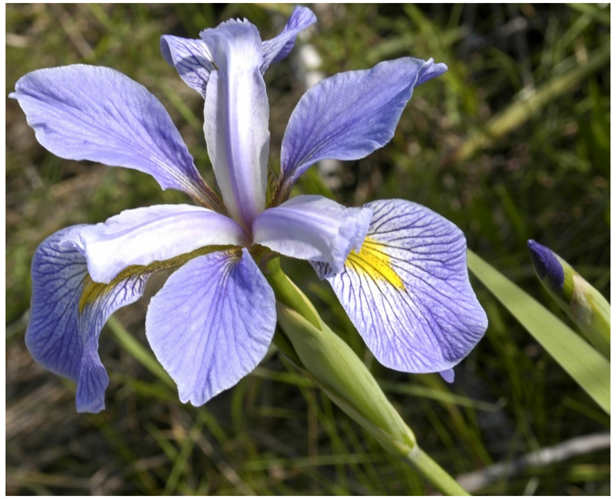
We want to distinguish between three species of the iris plant



Iris setosa



Iris versicolor



Iris virginica

"Kosaciec szczecinkowaty Iris setosa". Licensed under CC BY-SA 3.0 via Wikimedia Commons -
https://commons.wikimedia.org/wiki/File:Kosaciec_szczecinkowaty_Iris_setosa.jpg#mediaviewer/File:Kosaciec_szczecinkowaty_Iris_setosa.jpg
"Iris versicolor 3". Licensed under CC BY-SA 3.0 via Wikimedia Commons -
https://commons.wikimedia.org/wiki/File:Iris_versicolor_3.jpg#mediaviewer/File:Iris_versicolor_3.jpg
"Iris virginica" by Frank Mayfield - originally posted to Flickr as Iris virginica shrevei BLUE FLAG. Licensed under CC BY-SA 2.0 via Wikimedia Commons -
https://commons.wikimedia.org/wiki/File:Iris_virginica.jpg#mediaviewer/File:Iris_virginica.jpg

How do we extract features?

- If inputs are physical samples of flowers

- If inputs are photographs of flowers

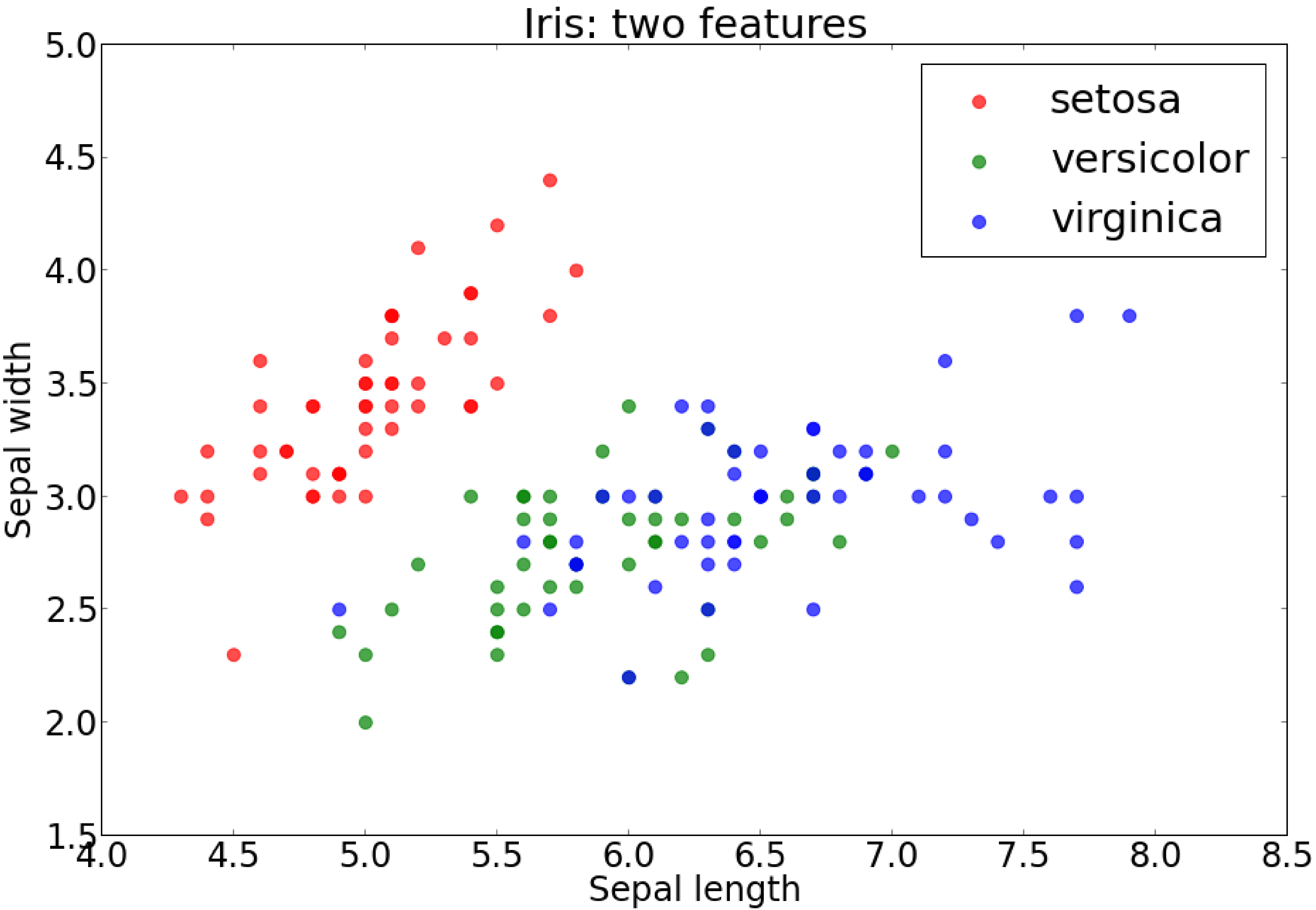
- If inputs are physical samples of flowers
  - Manual or automatic measurements
    - size of petals, leaves, color, weight, ...
- If inputs are photographs of flowers
  - Image processing: edges, color, gradients, ...
  - Automatic learning of features from pixels

The iris dataset

INPUT				OUTPUT
6.9	3.2	5.7	2.3	virginica
5.4	3.4	1.5	0.4	setosa
7.2	3.0	5.8	1.6	virginica
6.3	3.3	4.7	1.6	versicolor
5.8	2.7	3.9	1.2	versicolor
7.2	3.6	6.1	2.5	virginica
5.4	3.9	1.7	0.4	setosa

Features:

Sepal_Length, Sepal_Width, Petal_Length, Petal_Width


Iris: two features
setosa
versicolor
virginica
Sepal width
Sepal length
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5

Census income

INPUT					**TARGET**
age	**edu**	**occupation**	**race**	**sex**	**income**
39	13	Adm-clerical	White	Male	<=50K
50	13	Exec-managerial	White	Male	<=50K
38	9	Handlers-cleaners	White	Male	<=50K
53	7	Handlers-cleaners	Black	Male	<=50K
28	13	Prof-specialty	Black	Female	<=50K
37	14	Exec-managerial	White	Female	<=50K
49	5	Other-service	Black	Female	<=50K
52	9	Exec-managerial	White	Male	>50K
31	14	Prof-specialty	White	Female	>50K
42	13	Exec-managerial	White	Male	>50K
37	10	Exec-managerial	Black	Male	>50K
30	13	Prof-specialty	Asian	Male	>50K
23	13	Adm-clerical	White	Female	<=50K
32	12	Sales	Black	Male	<=50K

Categorical features

- Some algorithms can easily use categorical features such as `occupation` or `race` or `sex`
- In many cases we'll convert them to numerical features

Categorical → Numerical

race	sex
White	Male
Black	Male
Black	Female
White	Female
White	Male
Asian	Male

White	Black	Asian	Male	Female
1	0	0	1	0
0	1	0	1	0
0	1	0	0	1
1	0	0	0	1
1	0	0	1	0
0	0	1	1	0

Such new features are known as

- Dummy variables
- Indicator features
- Binarized features

Use features to predict targets

Simple idea: Similarity

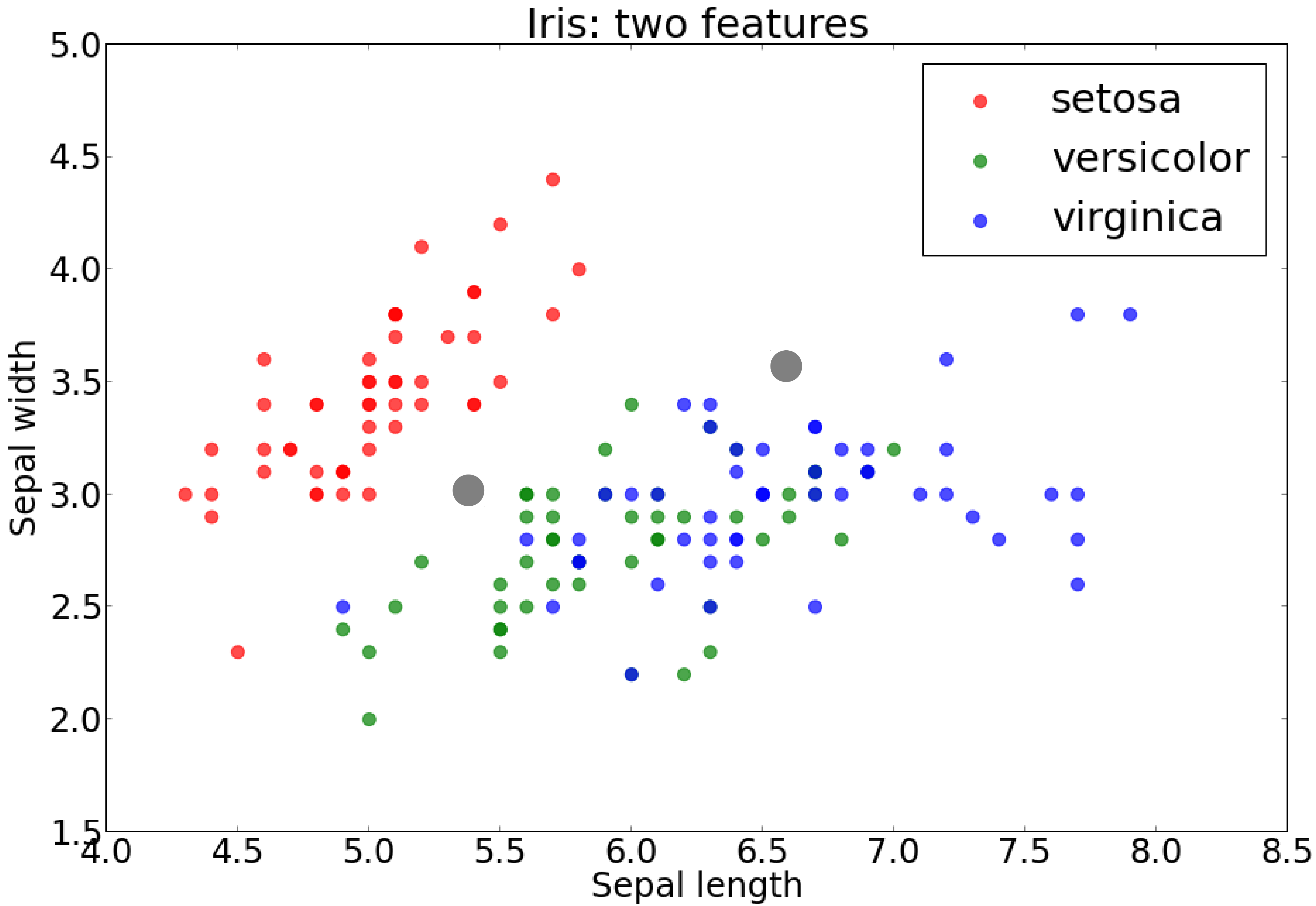
- Given a new example x_j
- We look for the most similar example in training set
- Predict the same target for x_j

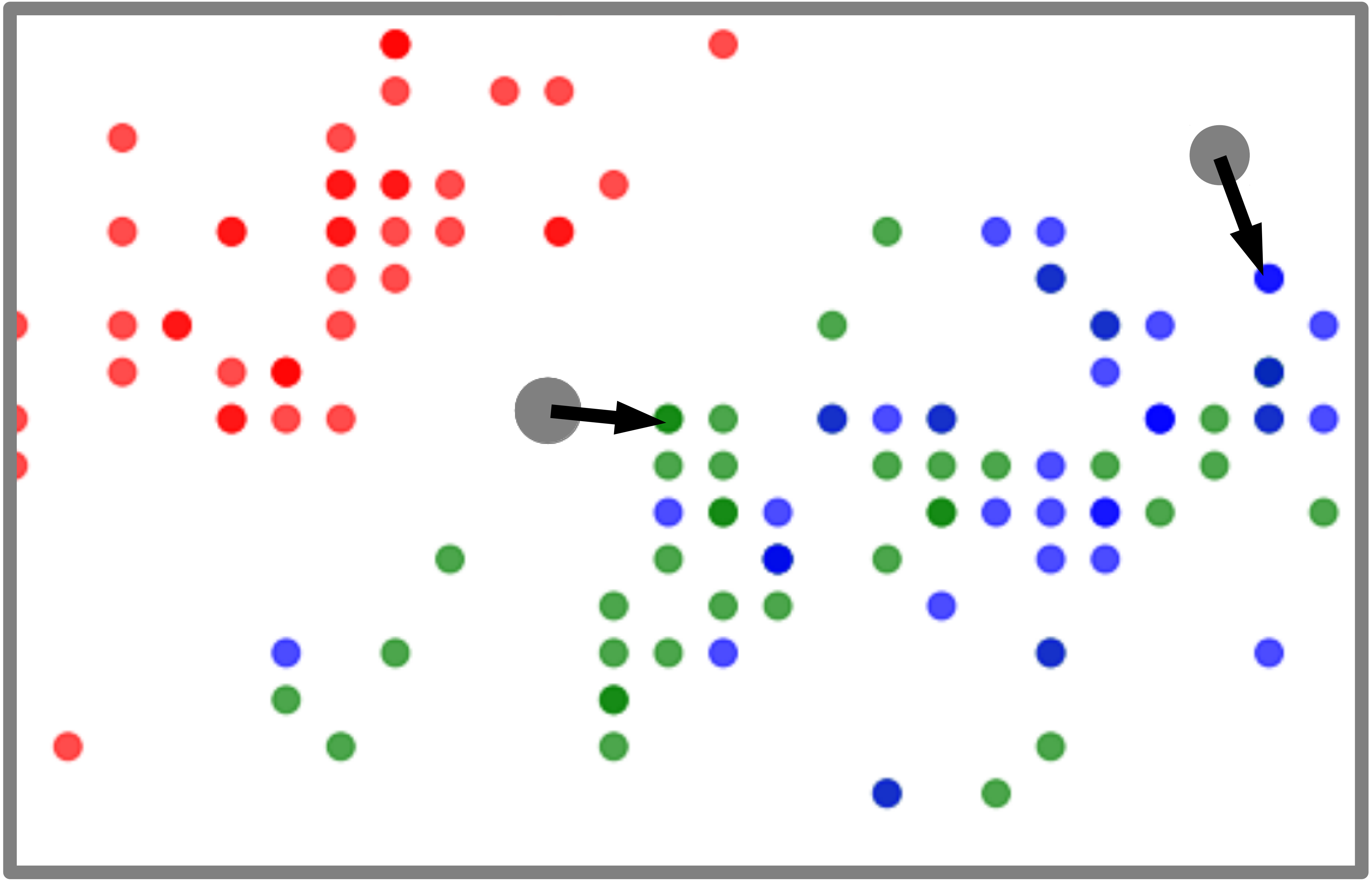
How do we measure similarity

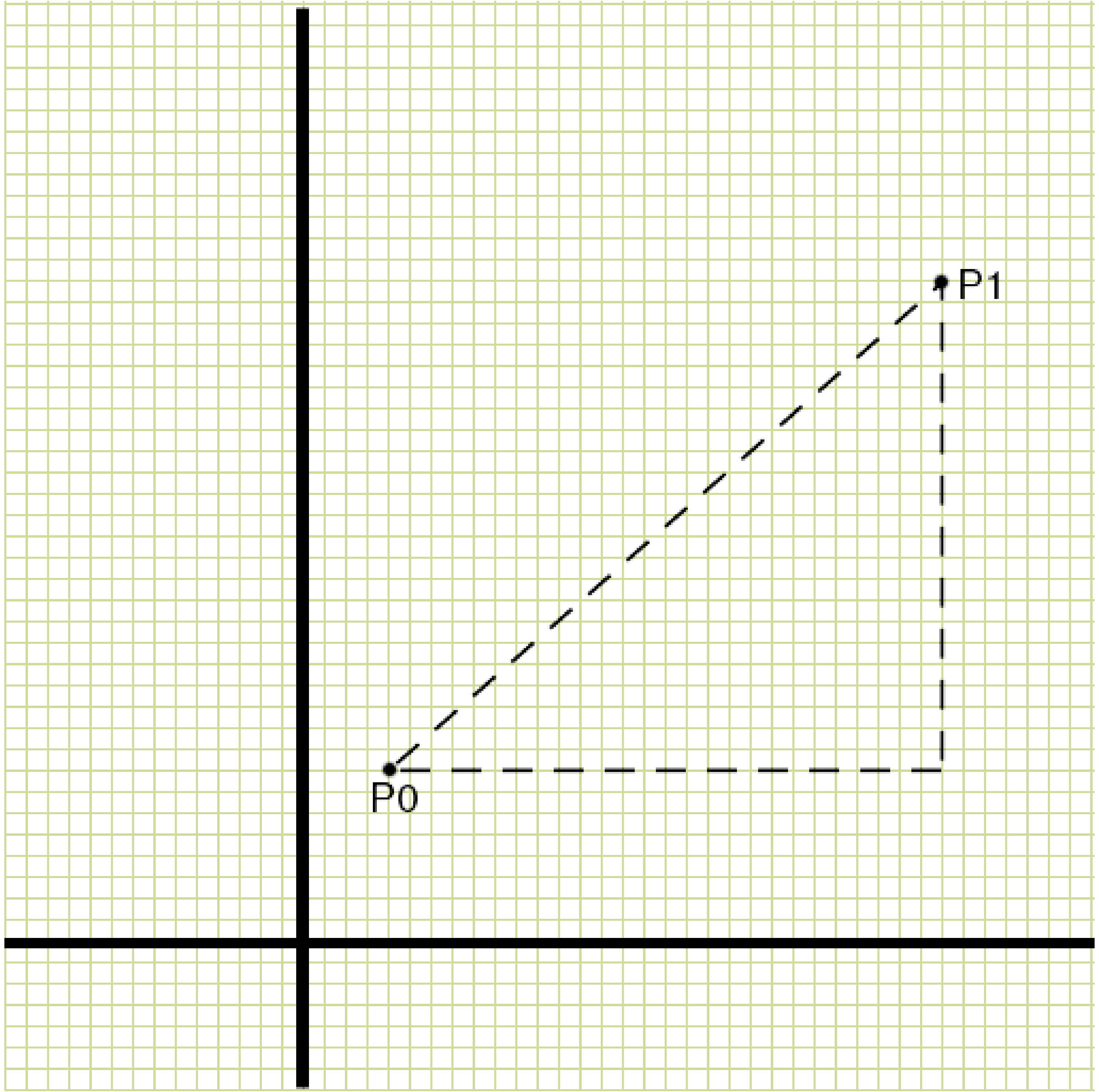
- Distance – opposite of similarity
- Find the nearest training example

How do we measure similarity

- Distance – opposite of similarity
- Find the nearest training example


Iris: two features
setosa
versicolor
virginica
Sepal width
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
Sepal length
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5





Source: http://resumbrae.com/ub/dms423/05/triangleOnCartesian.png

Distance

- Euclidean distance – like in physical space
  - In 2 dimensions

$$D(\mathbf{u},\mathbf{v}) = \sqrt{((u_1 - v_1)^2 + (u_2 - v_2)^2)}$$

  - In N dimensions

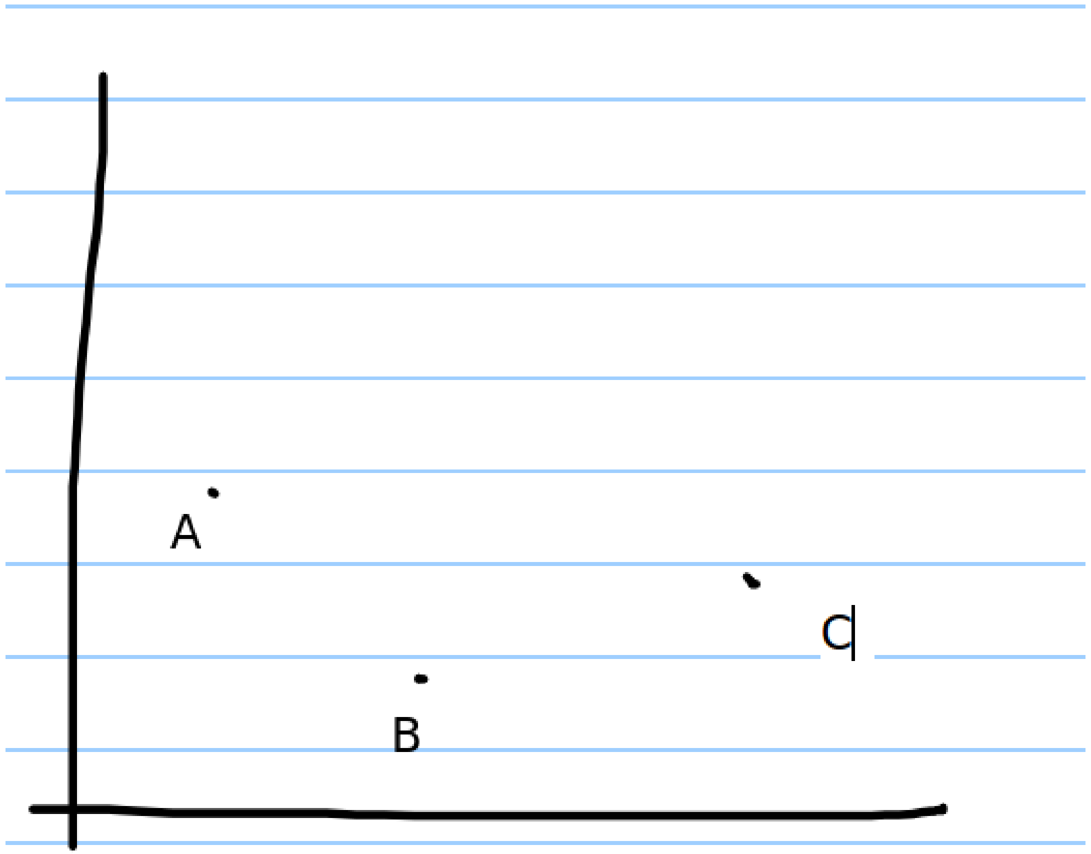
$$D(\mathbf{u},\mathbf{v}) = \sqrt{\left(\sum_{i=1}^{N}(u_i - v_i)^2\right)}$$

Other distances

Cosine similarity

$$D_{\cos}(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i=1}^{N} u_i v_i}{\sqrt{(\sum_{i=1}^{N} u_i^2)}\sqrt{(\sum_{i=1}^{N} v_i^2)}}$$

Euclidean vs Cosine



When would we want to use cosine distance instead of Euclidean distance?

Hamming distance

- Distance between fixed-length sequences
- Number of positions at which strings differ

$$D_{\text{hamming}}(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{N} [a_i \neq b_i]$$

- D(123, 124)
- D(karolin, kathrin)
- D(gattaca, gaccata)

Is Hamming distance useful for text?

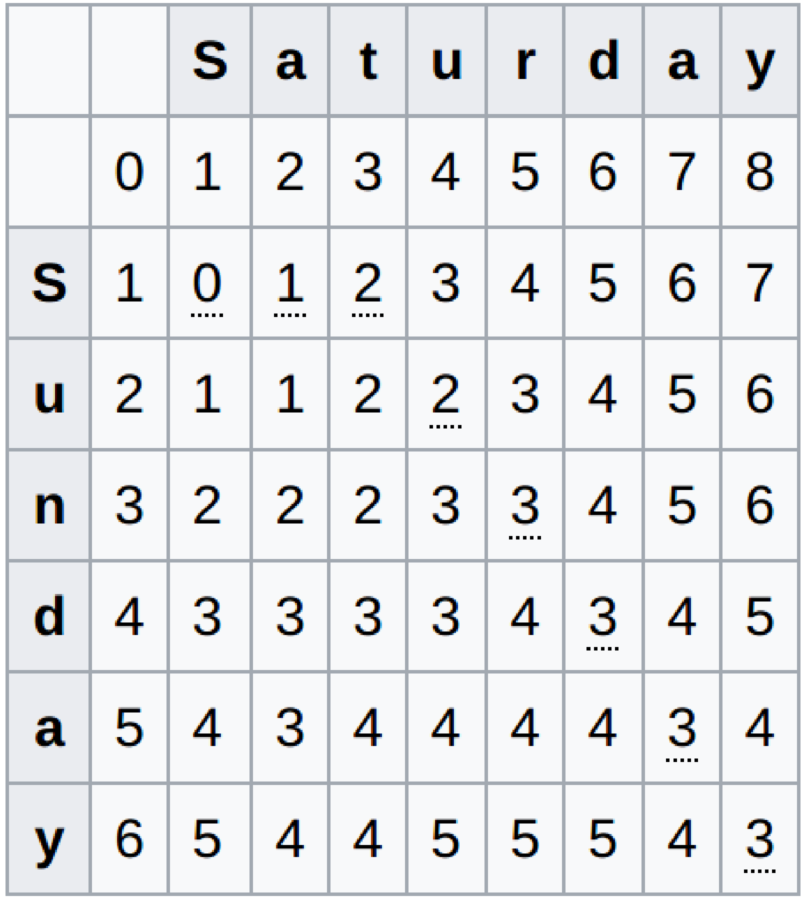
- What features do we want in a distance metric for text?

- collaboration
- laborious

Levenshtein edit distance

- Number of operations needed to convert string A into string B
  - Deletions
  - Insertions
  - (Substitutions)

Levenshtein distance – dynamic programming table



		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	4	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Finding the nearest neighbor

- Check the distances to all the training points, and pick the point with the smallest distance
- We need to remember the target of this point

Learning?

- In what sense can K-NN be said be learning?

- Illustrates most basic forms learning: **memorization**
- No **abstraction**

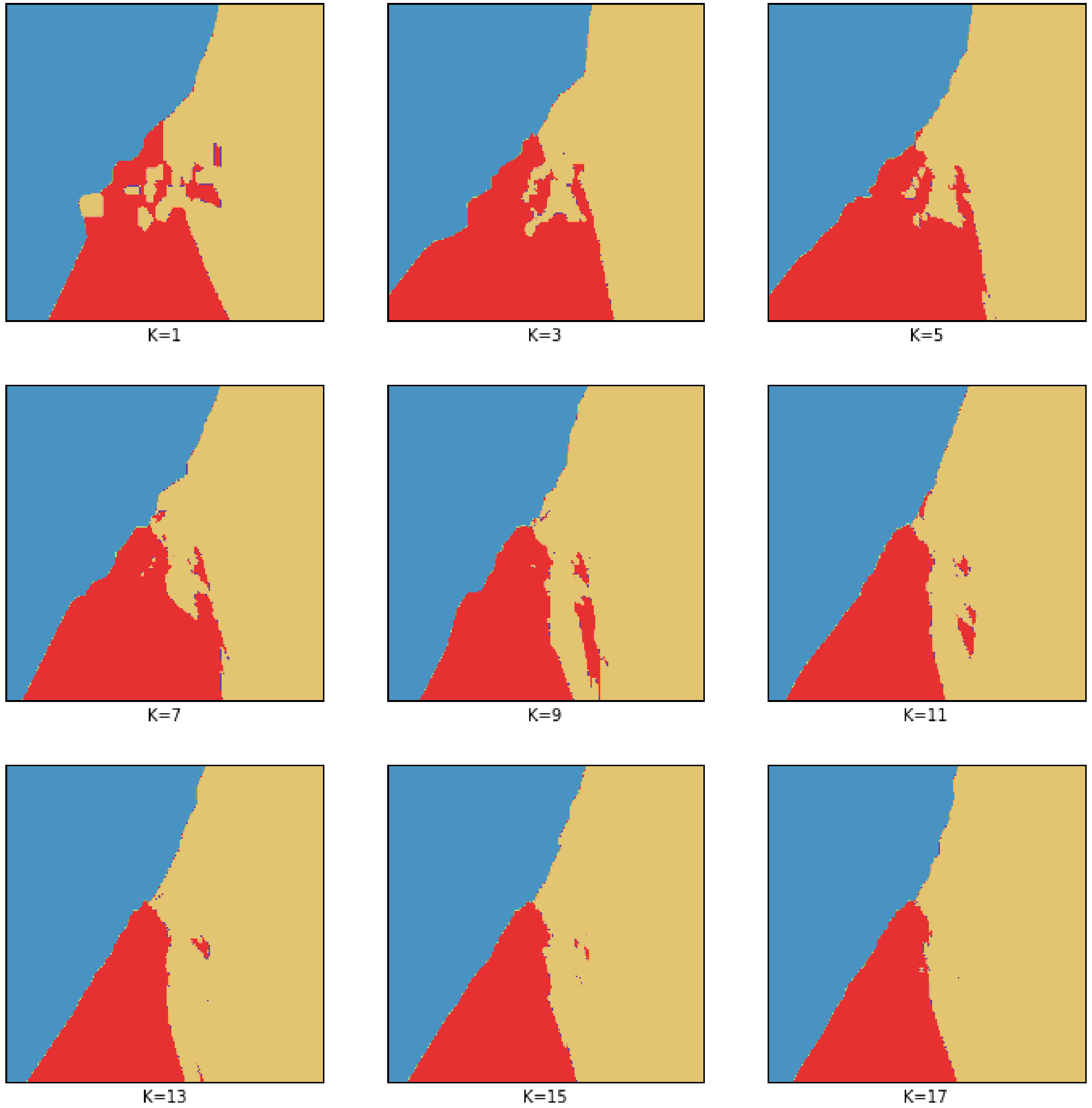
K-Nearest Neighbors

- Instead of only the closest example, we can look at several
- Predict the target which is most common among these

The role of K

- What effect does K have on the classification?

Iris: decision boundaries



K=1

K=3

K=5

K=7

K=9

K=11

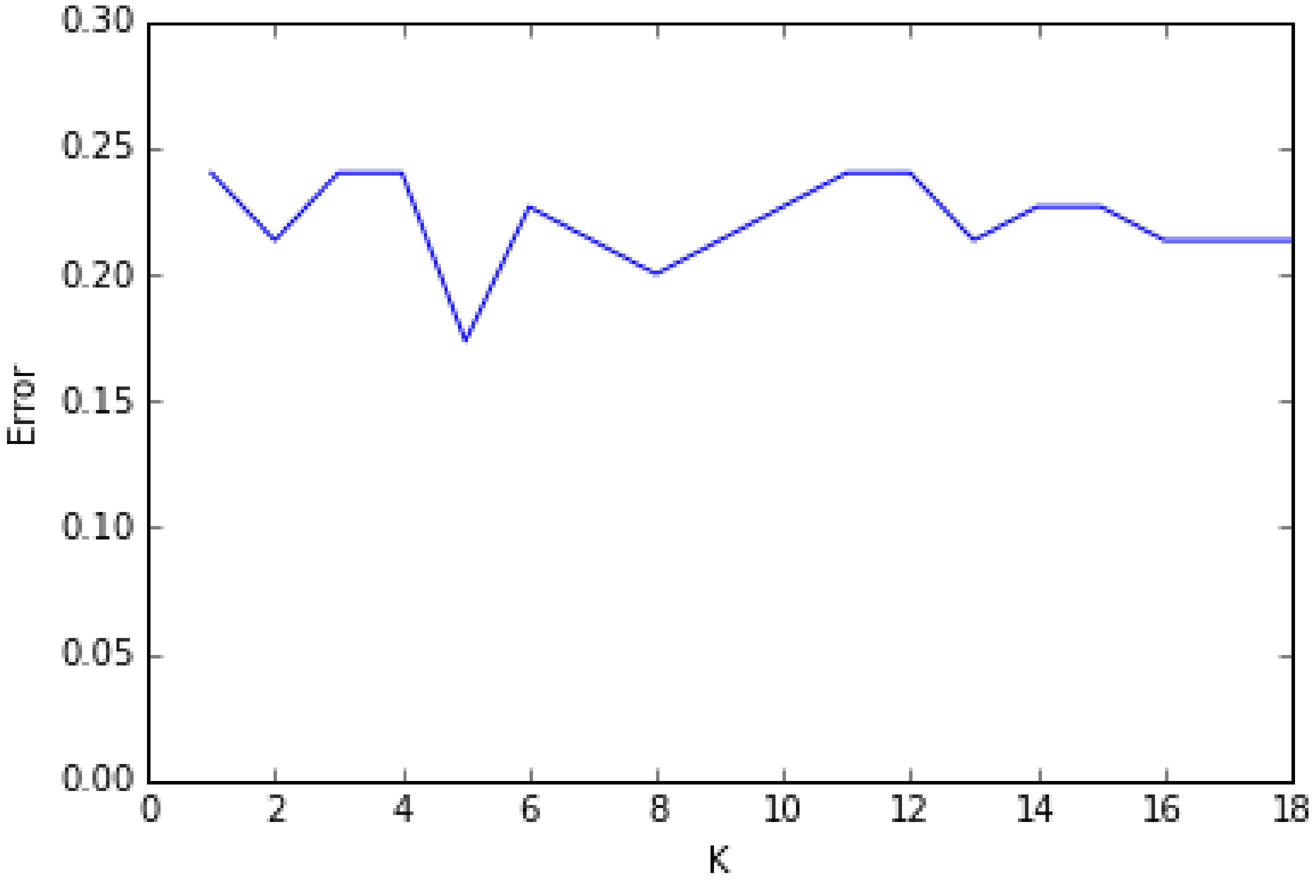
K=13

K=15

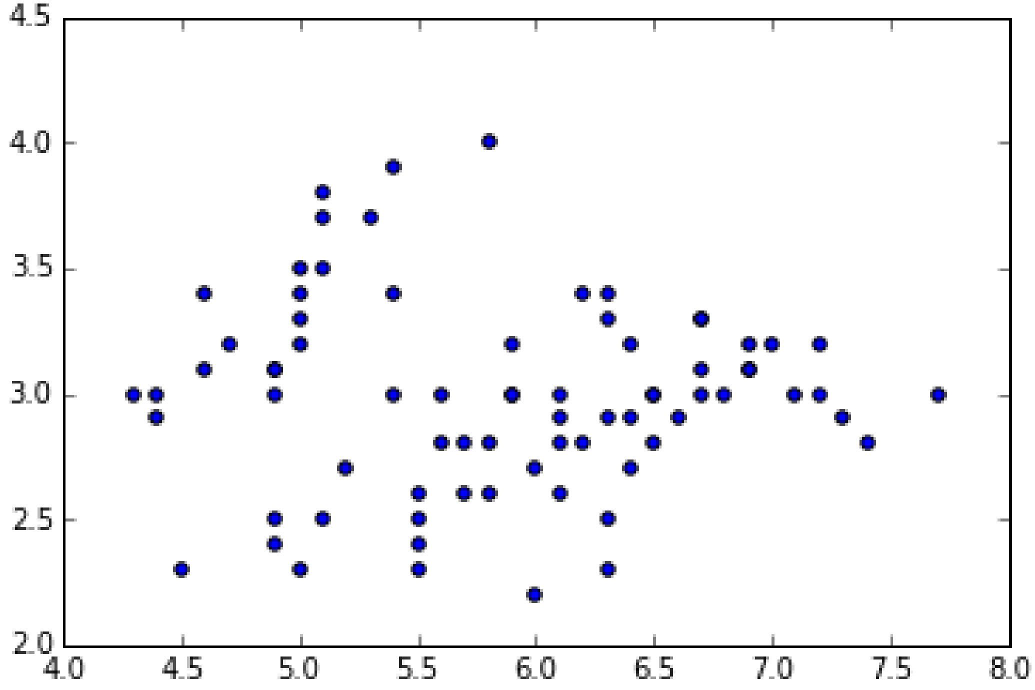
K=17

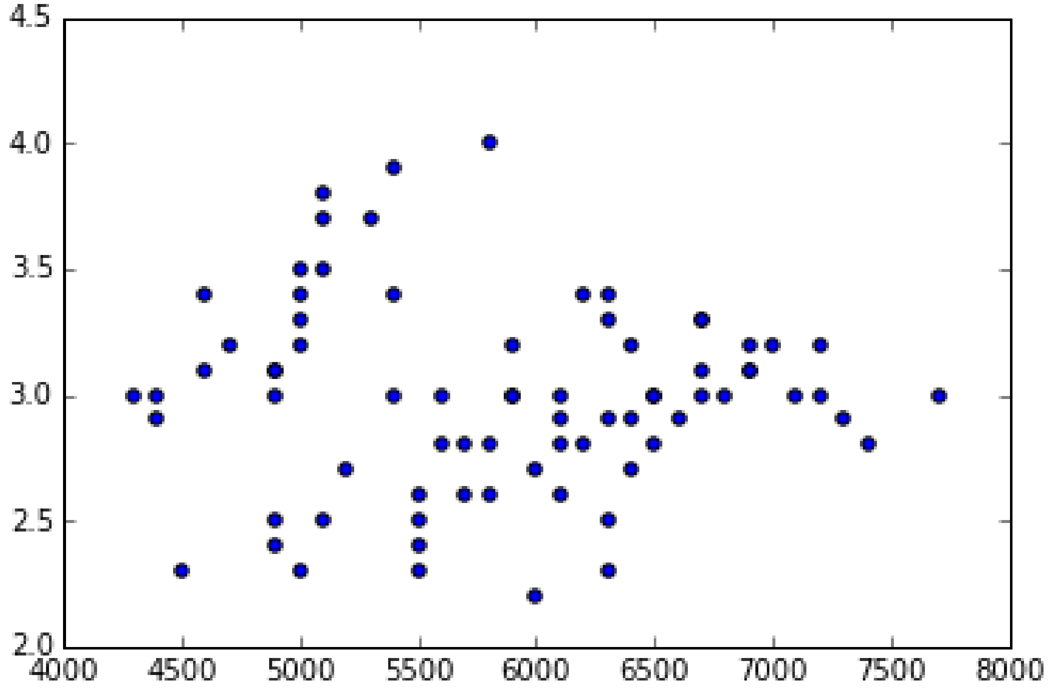
How to choose K?

Tuning on validation data

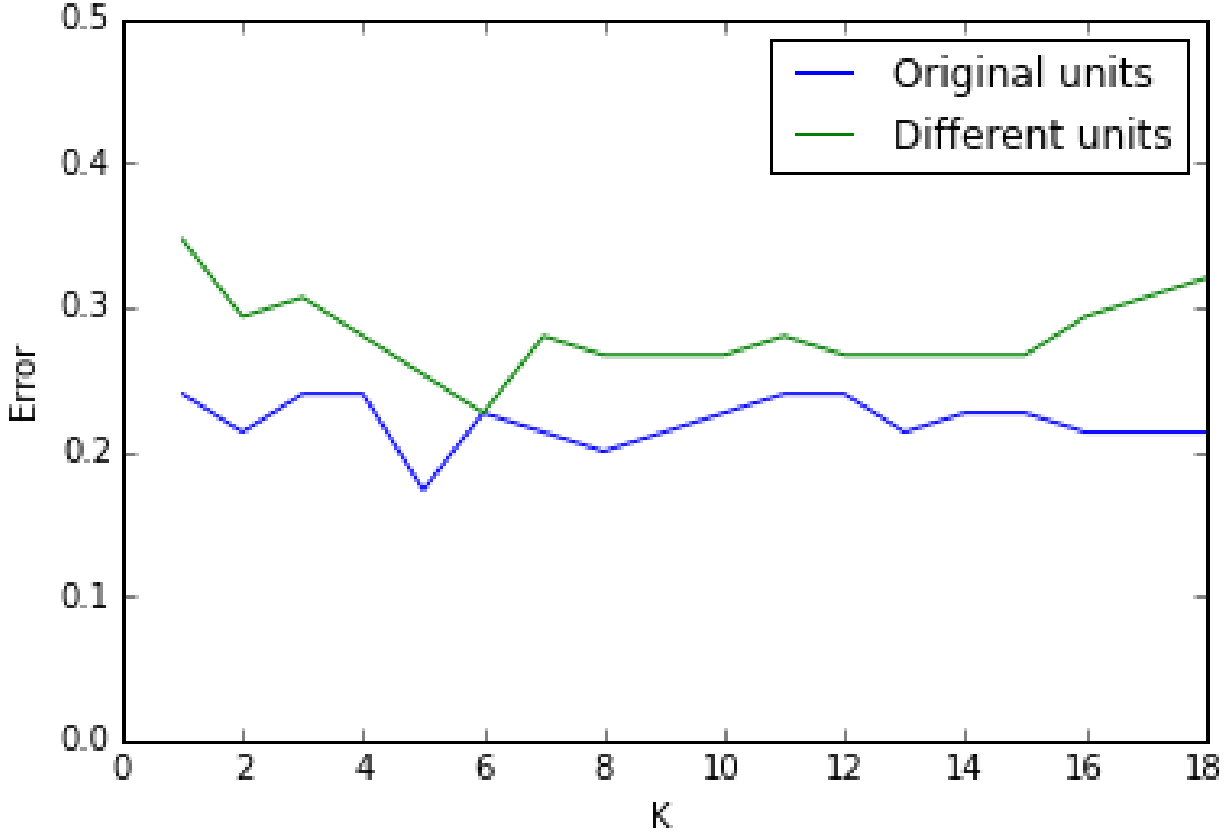


Units. Do they matter?



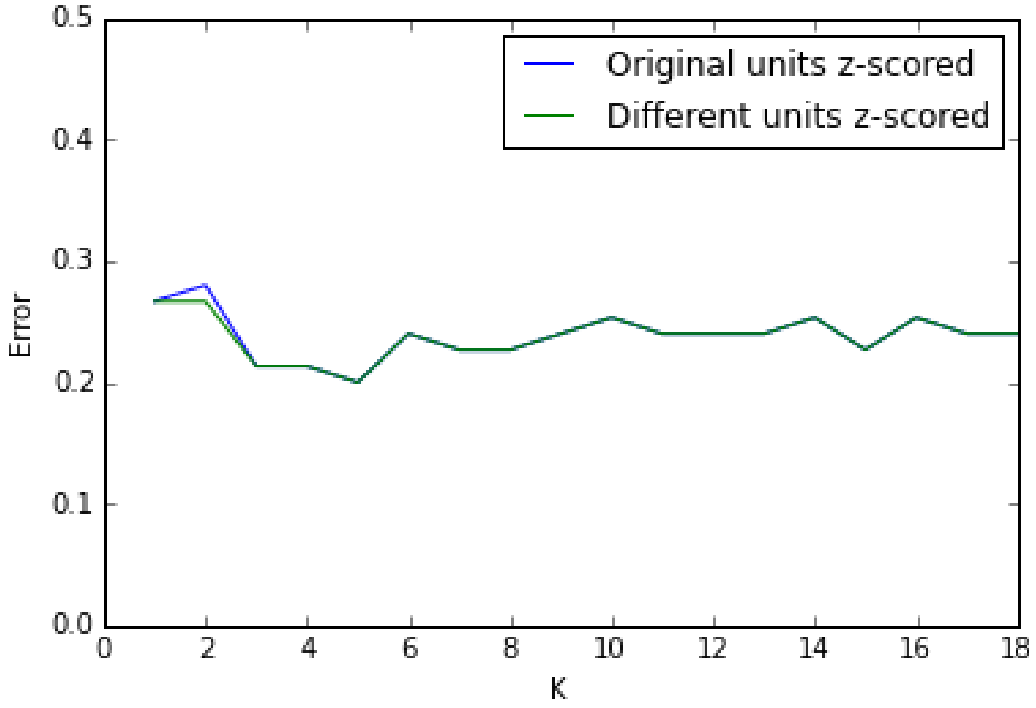


They do


Original units
Different units
Error
K
0.0
0.1
0.2
0.3
0.4
0.5
0
2
4
6
8
10
12
14
16
18

What's the solution?

- Standardize units → z-score dimensions



Problem

- Some dimensions may be more informative about the class than others
- Can we take this into account in the K-NN algorithm? How?

Using examples

Imagine you're studying for a very competitive exam – how do you use learning material?

Example sets

- Training set:
  - Observe patterns, infer rules
- Development set:
  - Monitor performance, choose best learning options
- Test set:
  - REAL EXAM
  - Not accessible in advance

Summary

- In order to learn from examples we decompose complex objects into features
- Often we need to convert categorical features into indicator features
- K-NN exemplified learning-as-memorization