Machine Learning Jam

A gentle introduction
to Machine Learning

The goal

- Take a real Kaggle data science contest

- Write some code and have fun
- Write a classifier, from scratch
- Compare & contrast functional languages
- Learn some Machine Learning concepts

- Bonus goal: send results to Kaggle contest?

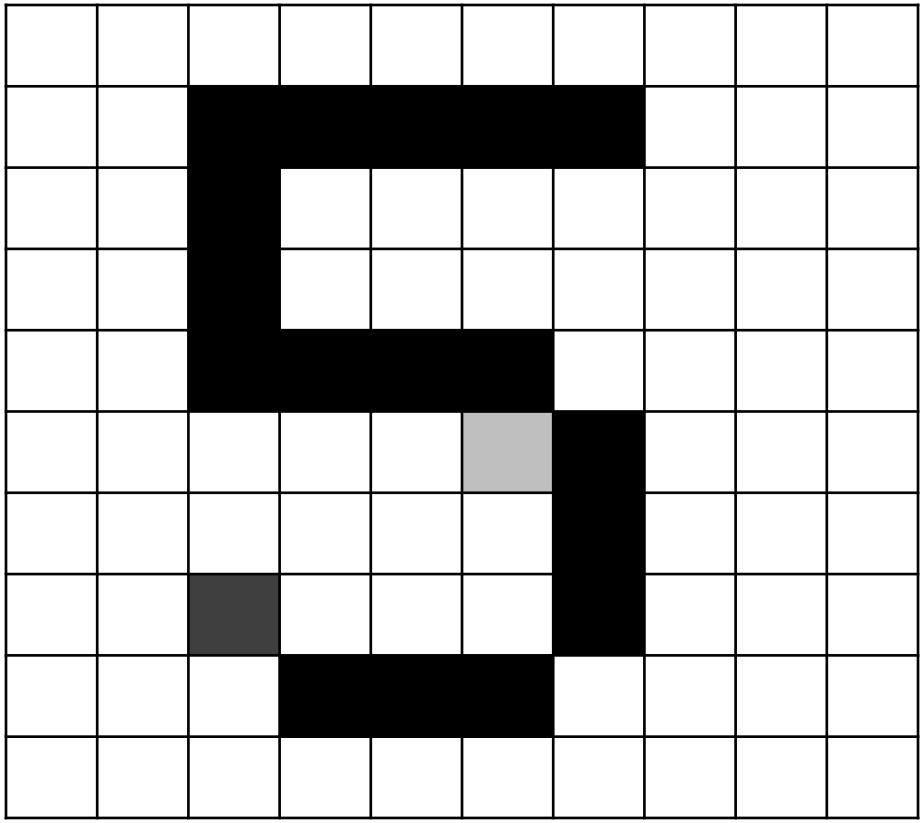
What you may need to know

Kaggle Digit Recognizer contest

- Full description on Kaggle.com

- Dataset: hand-written digits (0, 1, … , 9)
- Goal = automatically recognize digits
- Training sample = 50,000 examples
- Contest: classify 20,000 “unknown” digits

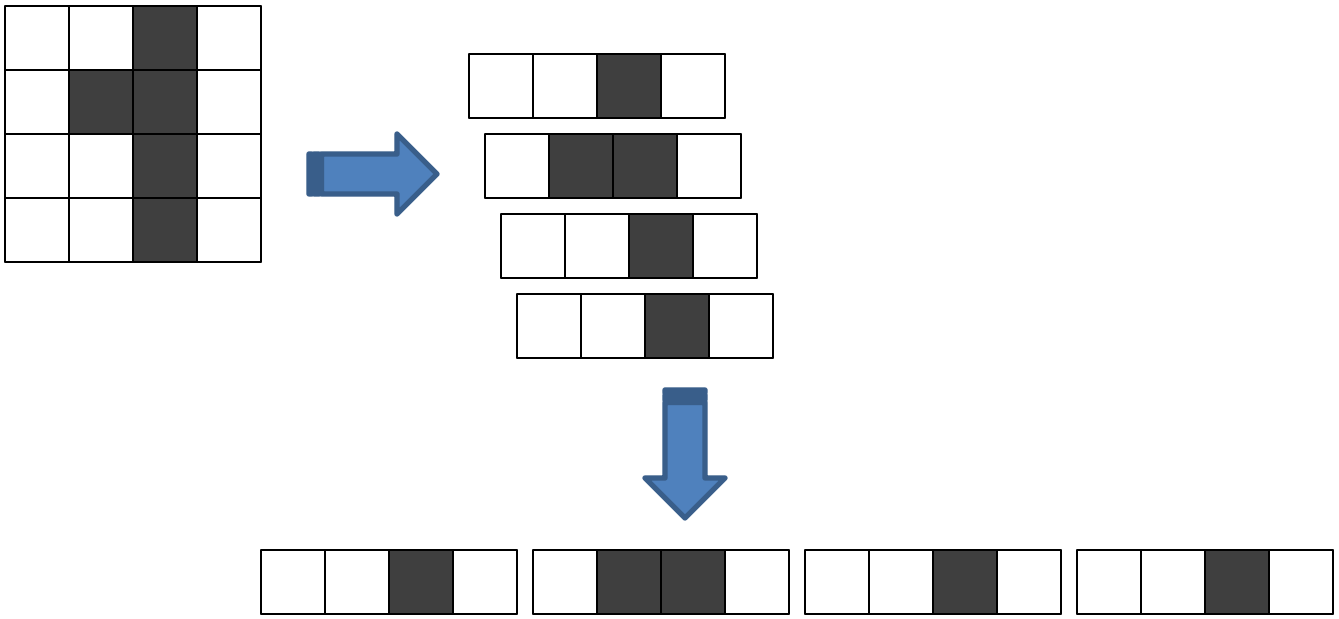
The data “looks like that”



Real data

- 28 x 28 pixels
- Grayscale: each pixel 0 (white) to 255 (black)
- Flattened: one record = Number + 784 Pixels
- CSV file with one line of column headers

Illustration (simplified data)



Label (The number represented by these pixels)

1,0,0,255,0,0,255,255,0,0,0,255,0,0,0,255,0

Pixels (28 x 28 = 784 fields, from 0 to 255)

What's a Classifier?

- "Give me an unknown data point, and I will predict what **class** it belongs to"
- In this case, classes = 0, 1, 2, … 9
- Unknown data point = scanned digit, without the class it belongs to

The KNN Classifier

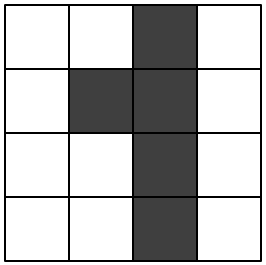
- KNN = K-Nearest-Neighbors algorithm

- Given an unknown subject to classify,
- Look up all the known examples,
- Find the K closest examples,
- Take a majority vote

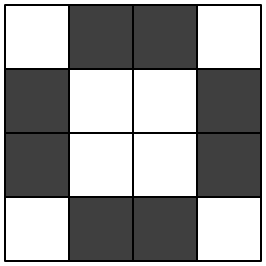
Illustration: 1 nearest neighbor

Examples

1

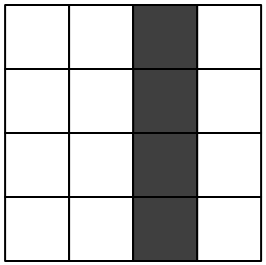


0



Unknown

?



Which example from the training set is nearest / closest to the Unknown item we want to classify?

What does “close” mean?

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_i - q_i)^2 + ... + (p_n - q_n)^2}.$$

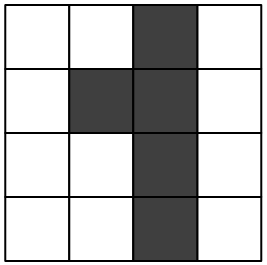
- To define “close” we need a distance
- We can use the distance between images as a measure for “close”
- Other distances can be used as well

- Note: Square root not important for our use case, can be omitted

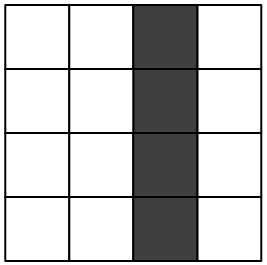
Illustration: 1 nearest neighbor

Examples **Unknown** **Differences** **Distances**

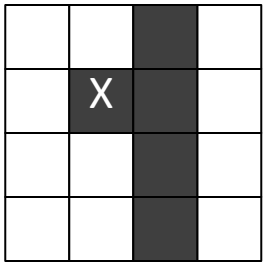
1



?

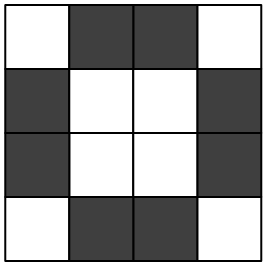


1

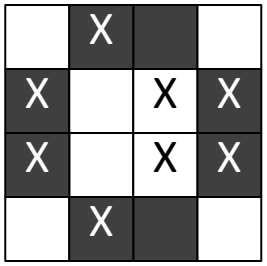


$\sqrt{255^2}$

0



0

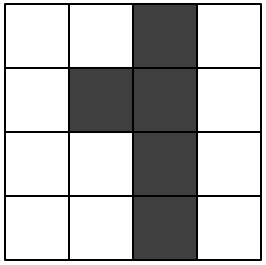


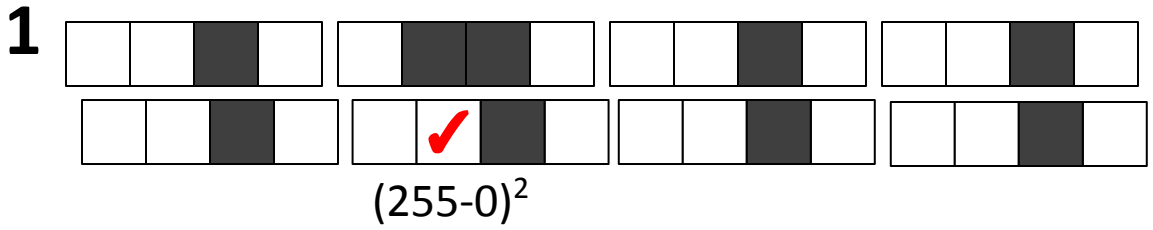
$\sqrt{(255^2+255^2+\ldots+255^2)}$

Illustration: 1 nearest neighbor

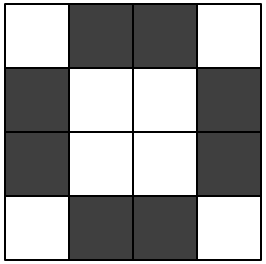
Examples

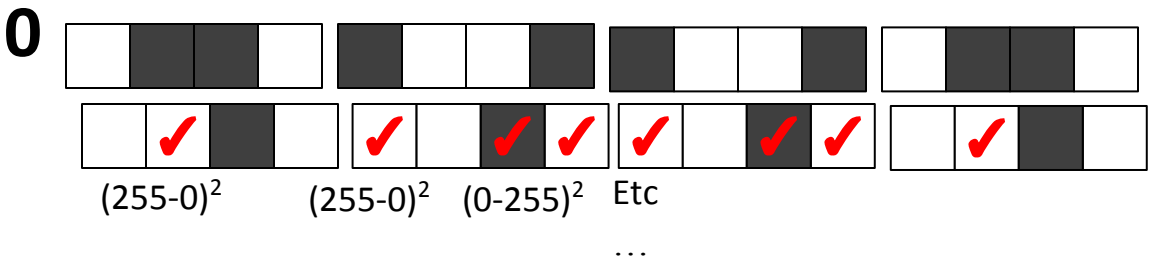
1





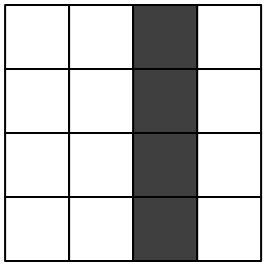
0





Unknown

?



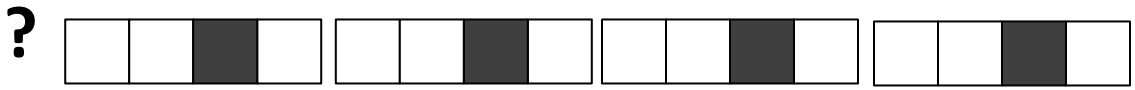
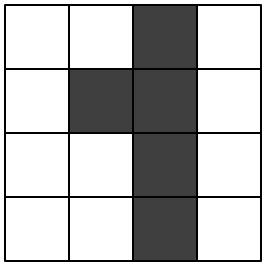


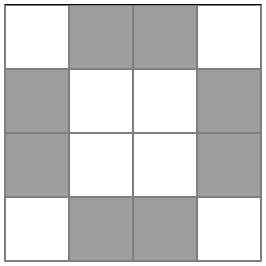
Illustration: 1 nearest neighbor

Examples

1

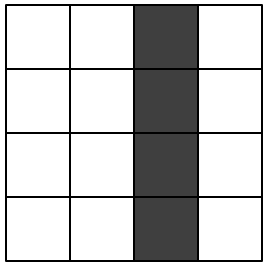


0



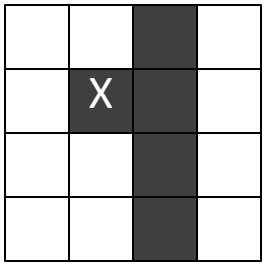
Unknown

?

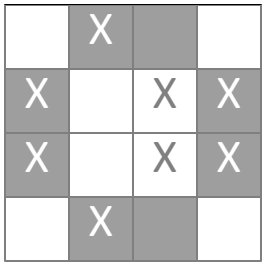


Differences

1



0



Distances

This item is closest: we classify our unknown as an instance of the same number: a 1

Questions?

Handout:

http://is.gd/lambdajam2013learning

Organization

- Form teams
- 1:00 – 2:45: code
- 2:45 – 3:00: prepare demo
- 3:00 – 4:00: demos (5 minutes each)

Handout:

http://is.gd/lambdajam2013learning

Form Teams

- Lambda Jam is an opportunity to discover other languages
- We need people who are ready and willing to help others learn: please come to the stage
- Everyone else: pick a language, find a group!
- Mentors also have language expertise and are excited to help!

Handout:

http://is.gd/lambdajam2013learning

Let's start coding!

Suggested path

- Use Euclidean distance first
- Build a 1-neighbor classifier
- What % of examples in Validation are correctly classified?

- … go wild ☺

Handout:

http://is.gd/lambdajam2013learning

Better Faster Stronger

- Different distance metric?
  - Windowing? Cubic?
- Bigger training set?
  - Embrace the metal \m/
- Scalability?
  - Eminently parallelizable.
- Other ideas?
  - More accuracy in less time is more better

Handout:

http://is.gd/lambdajam2013learning

Presentations!

Post code in a gist, tweet with hashtags
#lambdajam #jamming

How accurate?

How fast?

How elegant?

How scalable?