Xi'an Jiaotong-Liverpool University
西交利物浦大学

INT305 Machine Learning
Lecture 1

Jimin Xiao
Department Intelligence Science
Jimin.xiao@xjtlu.edu.cn

Greetings!

- Welcome to INT305 Machine Learning on-site/online edition!
- I am Jimin Xiao, your instructor, some of you already know me (CAN207).
- I hope you all are safe and healthy!

- As you know, course plan considering the COVID-19
  - Week 1-4 online teaching.
  - Week 5-14 onsite teaching with online option. (Tentative plan)

Course Goals and Prerequisites

- We're going to learn:
  - Theory of Machine Learning
    - Machine Learning Problem
    - Optimization Problem
  - Machine Learning Algorithms
    - Learning Algorithms
    - Optimization Algorithms
  - Practical Machine Learning
    - Learning from Artificial and Real-World Data

- Assume solid foundation in programming and math
  - Python Programming, Linear Algebra, Probability, Vector Calculus

Where to Get Information ?

- Lectures and Labs
- Learning Mall Online
  - Announcements, Lecture and Labs materials, Readings and Links
  - Discussion Forum
  - Coursework
  - Coursework with Lab practice

- Office hours
  - Friday 13:00-15:00, Jimin Xiao, SD563

- Textbooks
  - Bishop: Pattern Recognition and Machine Learning, by Chris Bishop.
  - Hastie, Tibshirani, and Friedman: The Elements of Statistical Learning

Teaching Team and Schedule

- Instructor: Jimin Xiao
  - jimin.xiao@xjtlu.edu.cn SD563 Monday 16:00-18:00
- Teaching Assistants:
  - Xinqiao Zhao
  - Jian Wang
  - Junwei Wu
- On-site
  - Lecture : Monday 16-18, SC162
  - Lab: Week 9-10, SD554

Assessment

1. Coursework 1 : 15%
   - The coursework requires no lab practice.

2. Coursework 2 : 15%
   - This coursework reuqires lab practice.

3. Final Exam : 70%
   - Final Exam is the most important part for assessment. It will be a open book exam.

This Course

- Broad introduction to machine learning
  - Algorithms and principles for supervised learning
    - nearest neighbors, decision trees, ensembles, linear regression, logistic regression, SVMs
  - Unsupervised learning: PCA, K-means, mixture models
  - Basics of reinforcement learning
- Coursework is aimed at advanced undergrads. We will use multivariate calculus, probability, and linear algebra.

Learning?

What is learning?

"The activity or process of gaining knowledge or skill by studying, practicing, being taught, or experiencing something."

Merriam Webster dictionary

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Tom Mitchell

What is machine learning?

- For many problems, it's difficult to program the correct behavior by hand
  - recognizing people and objects
  - understanding human speech
- Machine learning approach: program an algorithm to automatically learn from data, or from experience
- Why might you want to use a learning algorithm?

What is machine learning?

- For many problems, it's difficult to program the correct behavior by hand
  - recognizing people and objects
  - understanding human speech
- Machine learning approach: program an algorithm to automatically learn from data, or from experience
- Why might you want to use a learning algorithm?
  - hard to code up a solution by hand (e.g. vision, speech)
  - system needs to adapt to a changing environment (e.g. spam detection)
  - want the system to perform *better* than the human programmers
  - privacy/fairness (e.g. ranking search results)

What is machine learning?

- It's similar to statistics...
  - Both fields try to uncover patterns in data
  - Both fields draw heavily on calculus, probability, and linear algebra, and share many of the same core algorithms
- But it's not statistics!
  - Stats is more concerned with helping scientists and policymakers draw good conclusions; ML is more concerned with building autonomous agents
  - Stats puts more emphasis on interpretability and mathematical rigor; ML puts more emphasis on predictive performance, scalability, and autonomy

Relations to AI

- Nowadays, "machine learning" is often brought up with "artificial intelligence" (AI)
- AI does not always imply a learning based system
  - Symbolic reasoning
  - Rule based system
  - Tree search
  - etc.
- Learning based system $\rightarrow$ learned based on the data $\rightarrow$ more flexibility, good at solving pattern recognition problems.

Relations to Human Learning

- Human learning is:
  - Very data efficient
  - An entire multitasking system (vision, language, motor control, etc.)
  - Takes at least a few years :)
- For serving specific purposes, machine learning doesn't have to look like human learning in the end.
- It may borrow ideas from biological systems, e.g., neural networks.
- It may perform better or worse than humans.

What is machine learning?

- Types of machine learning
  - **Supervised learning:** have labeled examples of the correct behavior
  - **Reinforcement learning:** learning system (agent) interacts with the world and learns to maximize a scalar reward signal
  - **Unsupervised learning:** no labeled examples – instead, looking for "interesting" patterns in the data

History of machine learning?

- 1957 — Perceptron algorithm (implemented as a circuit!)
- 1959 — Arthur Samuel wrote a learning-based checkers program that could defeat him
- 1969 — Minsky and Papert's book *Perceptrons* (limitations of linear models)
- 1980s — Some foundational ideas
  - Connectionist psychologists explored neural models of cognition
  - 1984 — Leslie Valiant formalized the problem of learning as PAC learning
  - 1988 — Backpropagation (re-)discovered by Geoffrey Hinton and colleagues
  - 1988 — Judea Pearl's book *Probabilistic Reasoning in Intelligent Systems* introduced Bayesian networks

History of machine learning?

- 1990s — the “AI Winter”, a time of pessimism and low funding
- But looking back, the ’90s were also sort of a golden age for ML research
  - Markov chain Monte Carlo
  - variational inference
  - kernels and support vector machines
  - boosting
  - convolutional networks
  - reinforcement learning
- 2000s — applied AI fields (vision, NLP, etc.) adopted ML
- 2010s — deep learning
  - 2010–2012 — neural nets smashed previous records in speech-to-text and object recognition
  - increasing adoption by the tech industry
  - 2016 — AlphaGo defeated the human Go champion
  - 2018-now — generating photorealistic images and videos
  - 2020 — GPT3 language model
- now — increasing attention to ethical and societal implications

Machine learning in computer vision

Computer vision: Object detection, semantic segmentation, pose estimation, and almost every other task is done with ML.

Figure 4. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1).

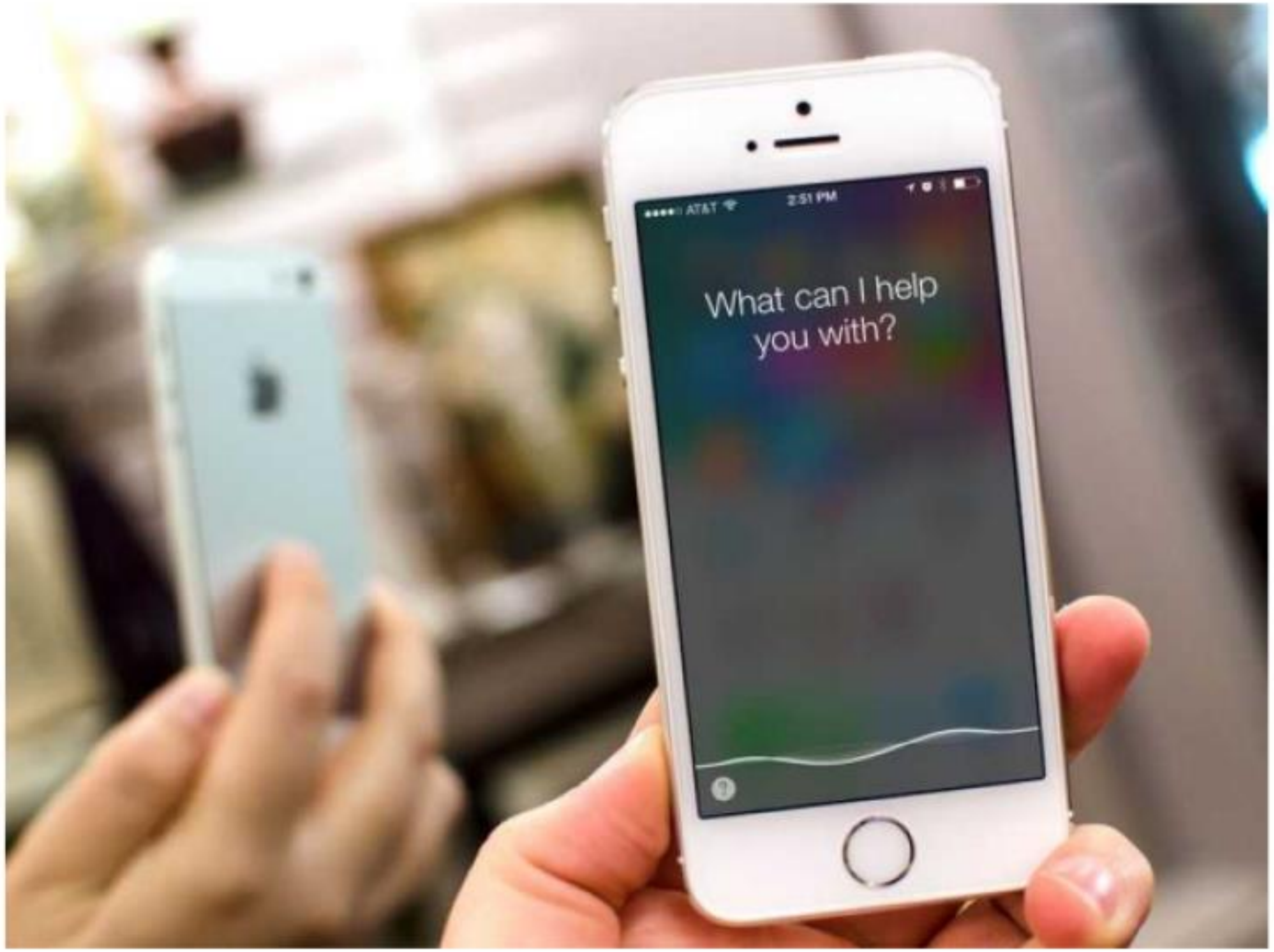
Instance segmentation

DAQUAR 1553
What is there in front of the sofa?
Ground truth: table
IMG+BOW: table (0.74)
2-VIS+BLSTM: table (0.88)
LSTM: chair (0.47)

COCOQA 5078
How many leftover donuts is the red bicycle holding?
Ground truth: three
IMG+BOW: two (0.51)
2-VIS+BLSTM: three (0.27)
BOW: one (0.29)

Machine learning in speech processing

Speech: Speech to text, personal assistants, speaker identification...



Machine learning in NLP

NLP: Machine translation, sentiment analysis, topic modeling, spam filtering.

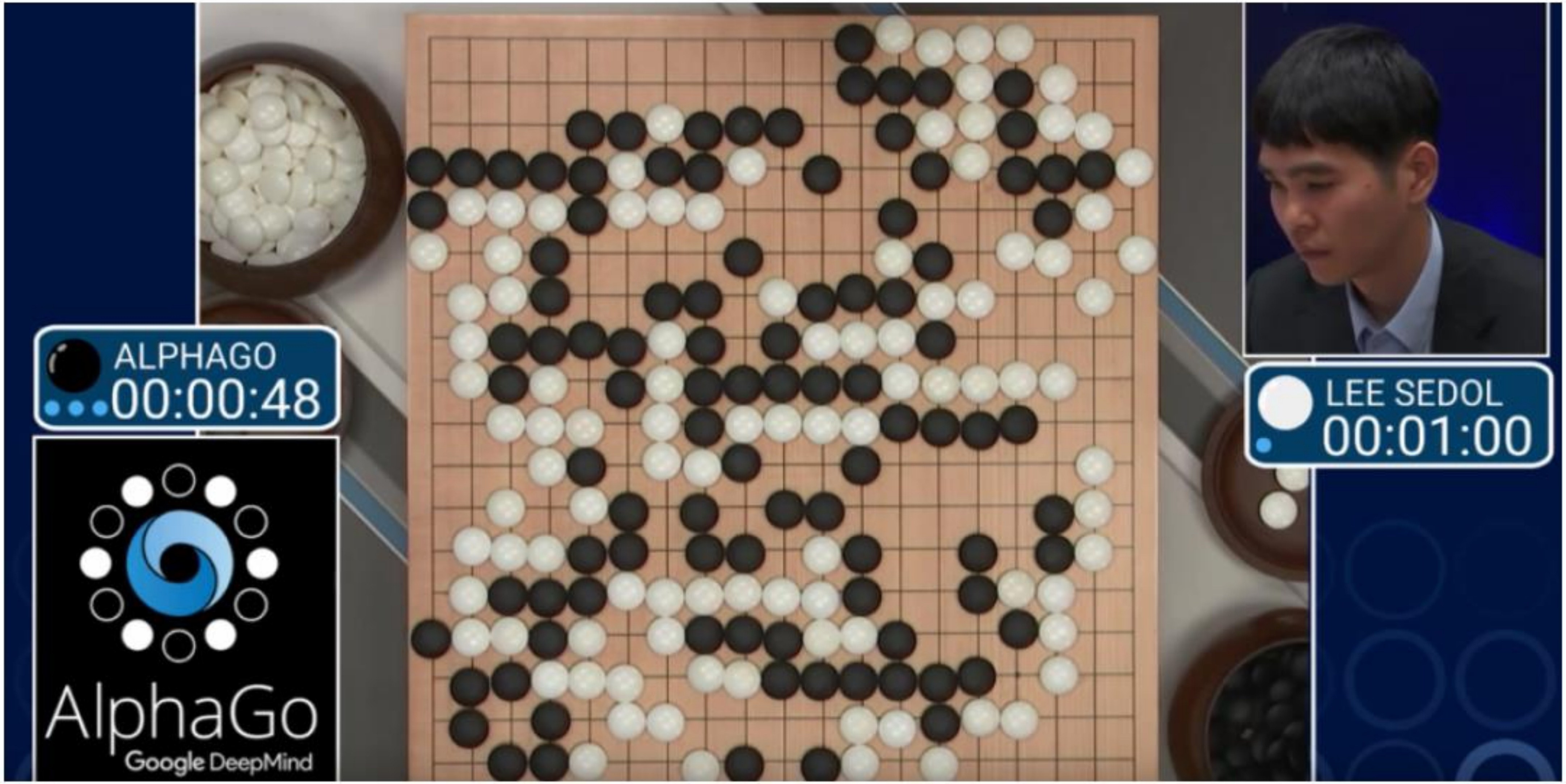
Real world example: The New York Times

LDA analysis of 1.8M New York Times articles:

music	book	art	game	show
band	life	museum	knicks	film
songs	novel	show	nets	television
rock	story	exhibition	points	movie
album	books	artist	team	series
jazz	man	artists	season	says
pop	stories	paintings	play	life
song	love	painting	games	man
singer	children	century	night	character
night	family	works	coach	know

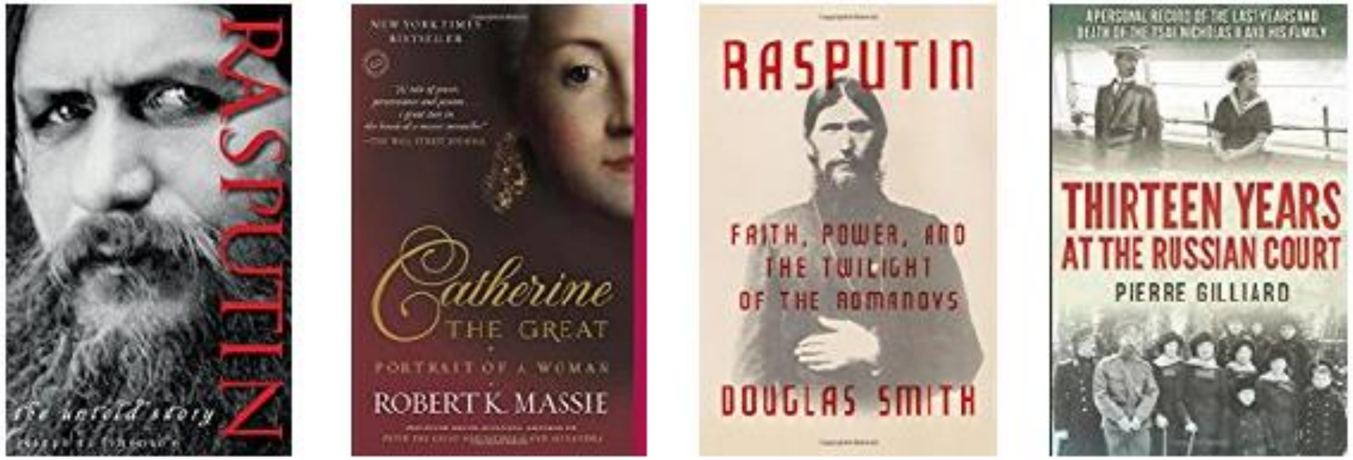
theater	clinton	stock	restaurant	budget
play	bush	market	sauce	tax
production	campaign	percent	menu	governor
show	gore	fund	food	county
stage	political	investors	dishes	mayor
street	republican	funds	street	billion
broadway	dole	companies	dining	taxes
director	presidential	stocks	dinner	plan
musical	senator	investment	chicken	legislature
directed	house	trading	served	fiscal

Machine learning in Game play

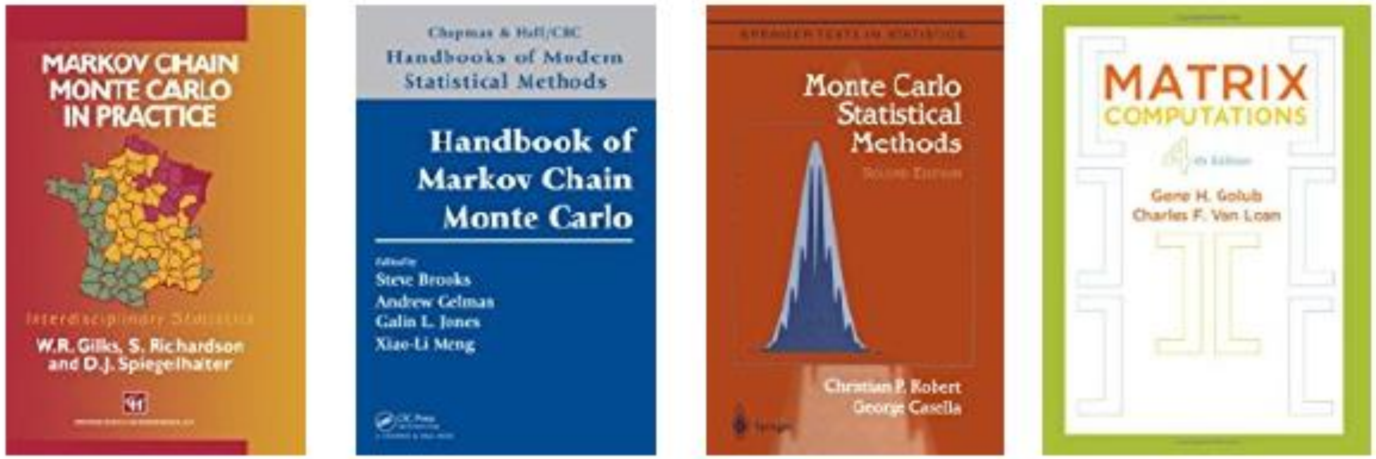


Machine learning in E-commerce

Inspired by your shopping trends

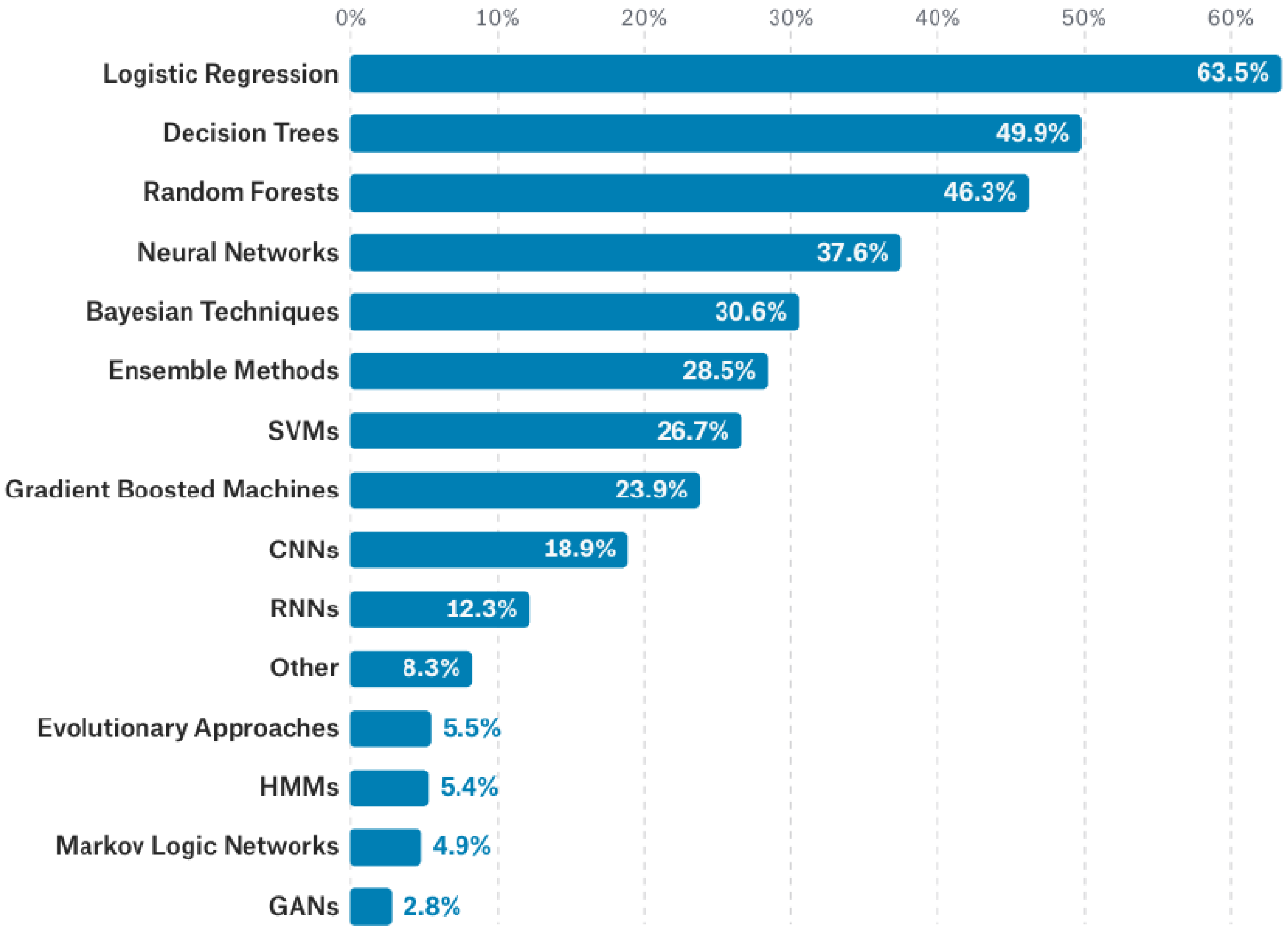


Related to items you've viewed See more



Why this class?

2017 Kaggle survey of data science and ML practitioners: what data science methods do you use at work?



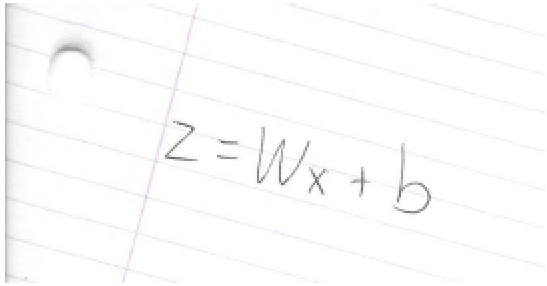
ML workflow

ML workflow sketch:

1. Should I use ML on this problem?
   - Is there a pattern to detect?
   - Can I solve it analytically?
   - Do I have data?
2. Gather and organize data.
   - Preprocessing, cleaning, visualizing.
3. Establishing a baseline.
4. Choosing a model, loss, regularization, ...
5. Optimization (could be simple, could be a Phd...).
6. Hyperparameter search.
7. Analyze performance & mistakes, and iterate back to step 4 (or 2).

Implementing machine learning systems

- You will often need to derive an algorithm (with pencil and paper), and then translate the math into code.
- Array processing (NumPy)
  - **vectorize** computations (express them in terms of matrix/vector operations) to exploit hardware efficiency
  - This also makes your code cleaner and more readable!




```
z = np.zeros(m)
for i in range(m):
    for j in range(n):
        z[i] += W[i, j] * x[j]
    z[i] += b[i]
```

```
z = W @ x + b
```

Implementing machine learning systems

- Neural net frameworks: PyTorch, TensorFlow, JAX, etc.
  - automatic differentiation
  - compiling computation graphs
  - libraries of algorithms and network primitives
  - support for graphics processing units (GPUs)
- Why take this class if these frameworks do so much for you?
  - So you know what to do if something goes wrong!
  - Debugging learning algorithms requires sophisticated detective work, which requires understanding what goes on beneath the hood.
  - That's why we derive things by hand in this class!

Nearest Neighbor

Preliminaries and Nearest Neighbor Methods

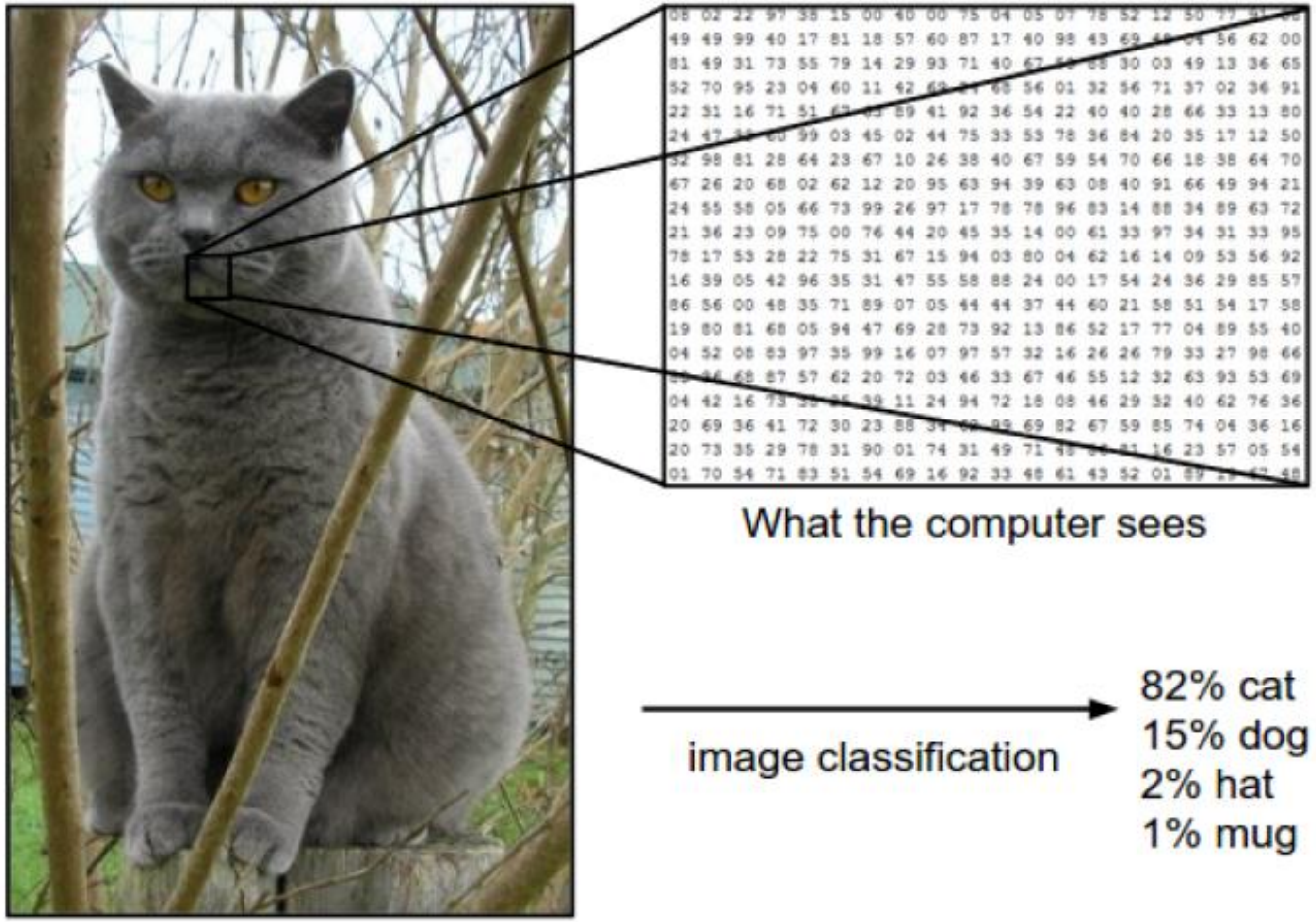
Introduction

- Today (and for much of this course) we focus on supervised learning.
- This means we are given a training set consisting of inputs and corresponding labels, e.g.

Task	Inputs	Labels
object recognition	image	object category
image captioning	image	caption
document classification	text	document category
speech-to-text	audio waveform	text
⋮	⋮	⋮

Input Vectors

What an image looks like to the computer:

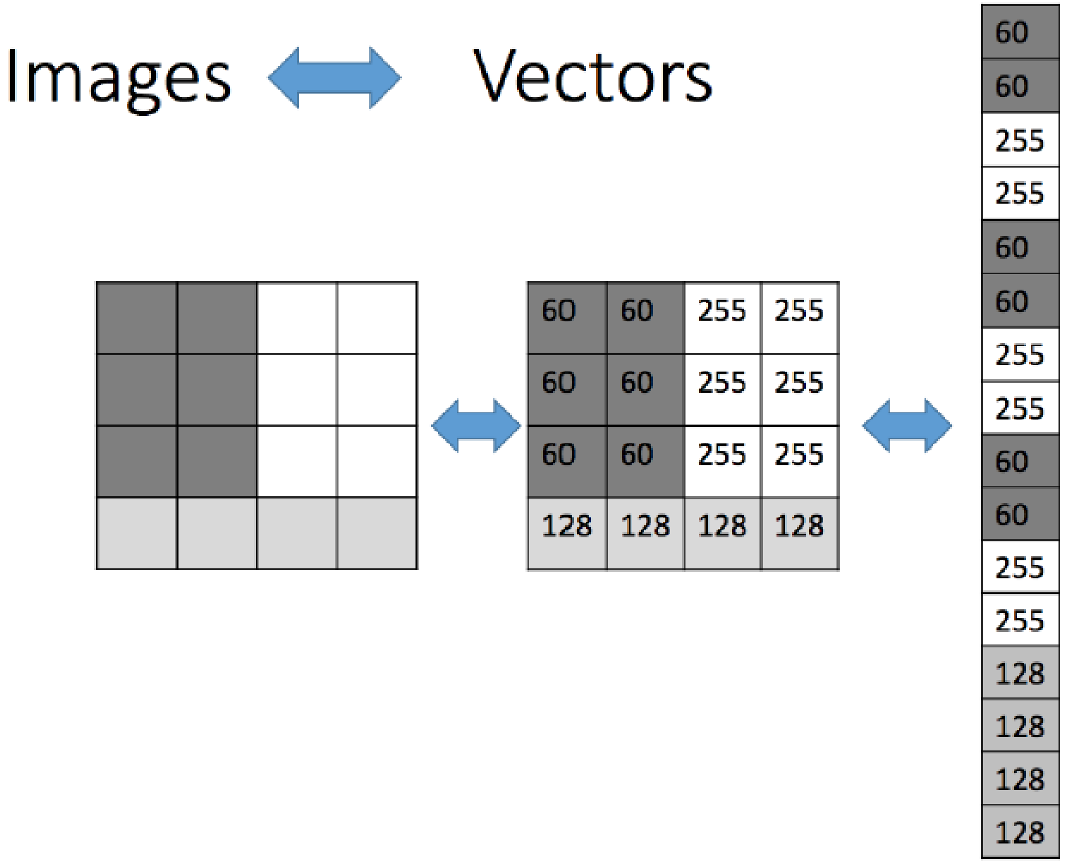


Input Vectors

- Machine learning algorithms need to handle lots of types of data: images, text, audio waveforms, credit card transactions, etc.
- Common strategy: represent the input as an input vector in $\mathbb{R}^d$
  - Representation = mapping to another space that's easy to manipulate
  - Vectors are a great representation since we can do linear algebra!

Input Vectors

Can use raw pixels:



Can do much better if you compute a vector of meaningful features.

Input Vectors

- Mathematically, our training set consists of a collection of pairs of an input vector $\mathbf{x} \in \mathbb{R}^d$ and its corresponding target, or label, t
  - Regression: t is a real number (e.g. stock price)
  - Classification: t is an element of a discrete set $\{1, \ldots, C\}$
  - These days, t is often a highly structured object (e.g. image)
- Denote the training set $\{(\mathbf{x}^{(1)}, t^{(1)}), \ldots, (\mathbf{x}^{(N)}, t^{(N)})\}$
  - Note: these superscripts have nothing to do with exponentiation!

Input Vectors

- Mathematically, our training set consists of a collection of pairs of an input vector $\mathbf{x} \in \mathbb{R}^d$ and its corresponding target, or label, t
  - Regression: t is a real number (e.g. stock price)
  - Classification: t is an element of a discrete set $\{1, \ldots, C\}$
  - These days, t is often a highly structured object (e.g. image)
- Denote the training set $\{(\mathbf{x}^{(1)}, t^{(1)}), \ldots, (\mathbf{x}^{(N)}, t^{(N)})\}$
  - Note: these superscripts have nothing to do with exponentiation!

Nearest Neighbors

- Suppose we're given a novel input vector $\mathbf{x}$ we'd like to classify.
- The idea: find the nearest input vector to $\mathbf{x}$ in the training set and copy its label.
- Can formalize "nearest" in terms of Euclidean distance

$$||\mathbf{x}^{(a)} - \mathbf{x}^{(b)}||_2 = \sqrt{\sum_{j=1}^{d} (x_j^{(a)} - x_j^{(b)})^2}$$

Algorithm:

1. Find example $(\mathbf{x}^*, t^*)$ (from the stored training set) closest to $\mathbf{x}$. That is:

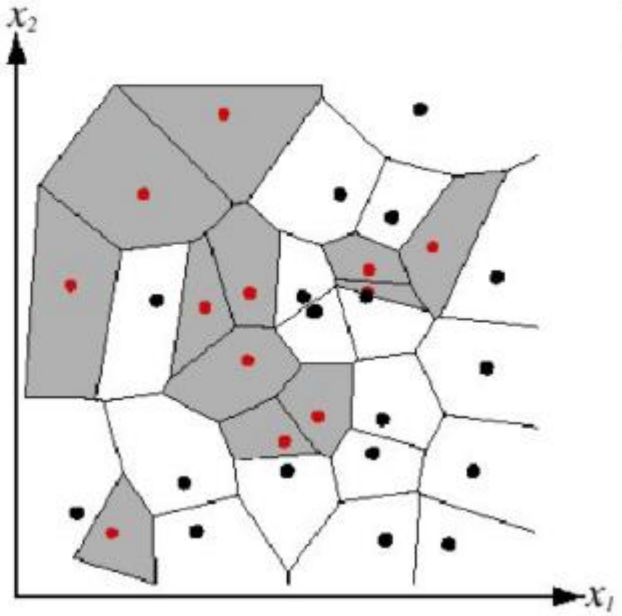
$$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{train. set}}{\operatorname{argmin}} \ \text{distance}(\mathbf{x}^{(i)}, \mathbf{x})$$

2. Output $y = t^*$

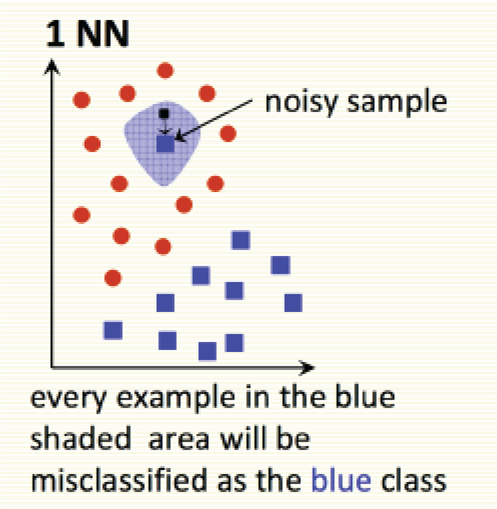
- Note: we don't need to compute the square root. Why?

Nearest Neighbors: Decision Boundaries

We can visualize the behavior in the classification setting using a Voronoi diagram.



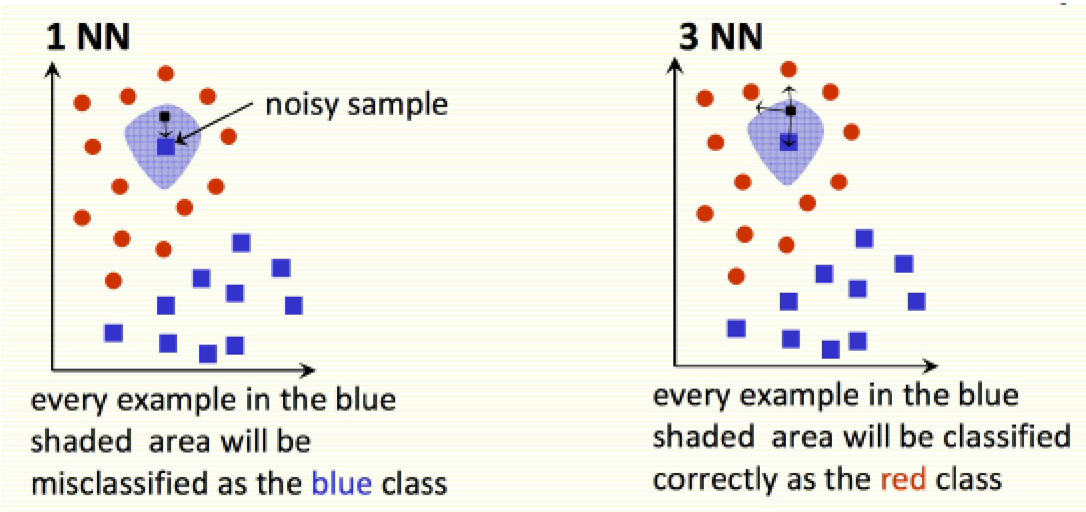
Nearest Neighbors



[Pic by Olga Veksler]

- Nearest neighbors sensitive to noise or mis-labeled data ("class noise"). Solution?

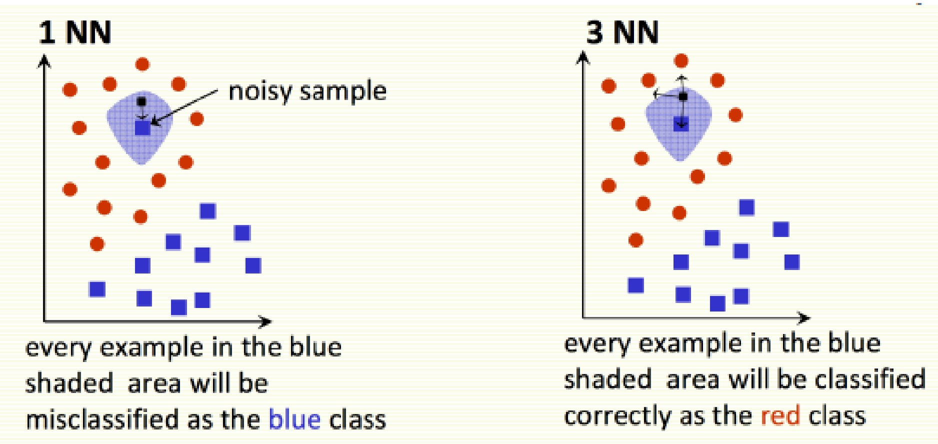
k-Nearest Neighbors



'ic by Olga Veksler]

- Nearest neighbors sensitive to noise or mis-labeled data ("class noise"). Solution?
- Smooth by having k nearest neighbors vote

k-Nearest Neighbors



'ic by Olga Veksler]

- Nearest neighbors sensitive to noise or mis-labeled data ("class noise"). Solution?
- Smooth by having k nearest neighbors vote

Algorithm (kNN):

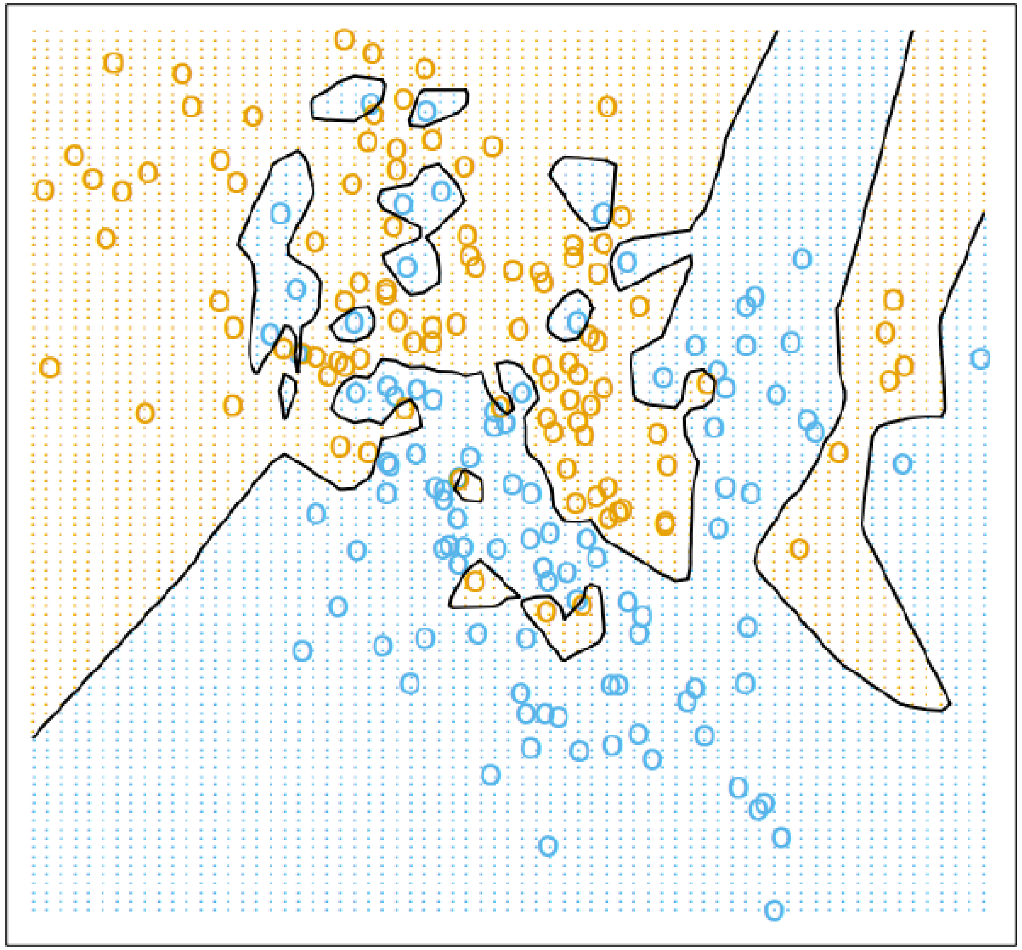
1. Find k examples $\{\mathbf{x}^{(i)}, t^{(i)}\}$ closest to the test instance $\mathbf{x}$
2. Classification output is majority class

$$y = arg \max_{t^{(z)}} \sum_{i=1}^{k} \mathbb{I}(t^{(z)} = t^{(i)})$$

$\mathbb{I}\{\text{statement}\}$ is the identity function and is equal to one whenever the statement is true. We could also write this as $\delta(t^{(z)}, t^{(i)})$, with $\delta(a, b) = 1$ if $a = b$, 0 otherwise.

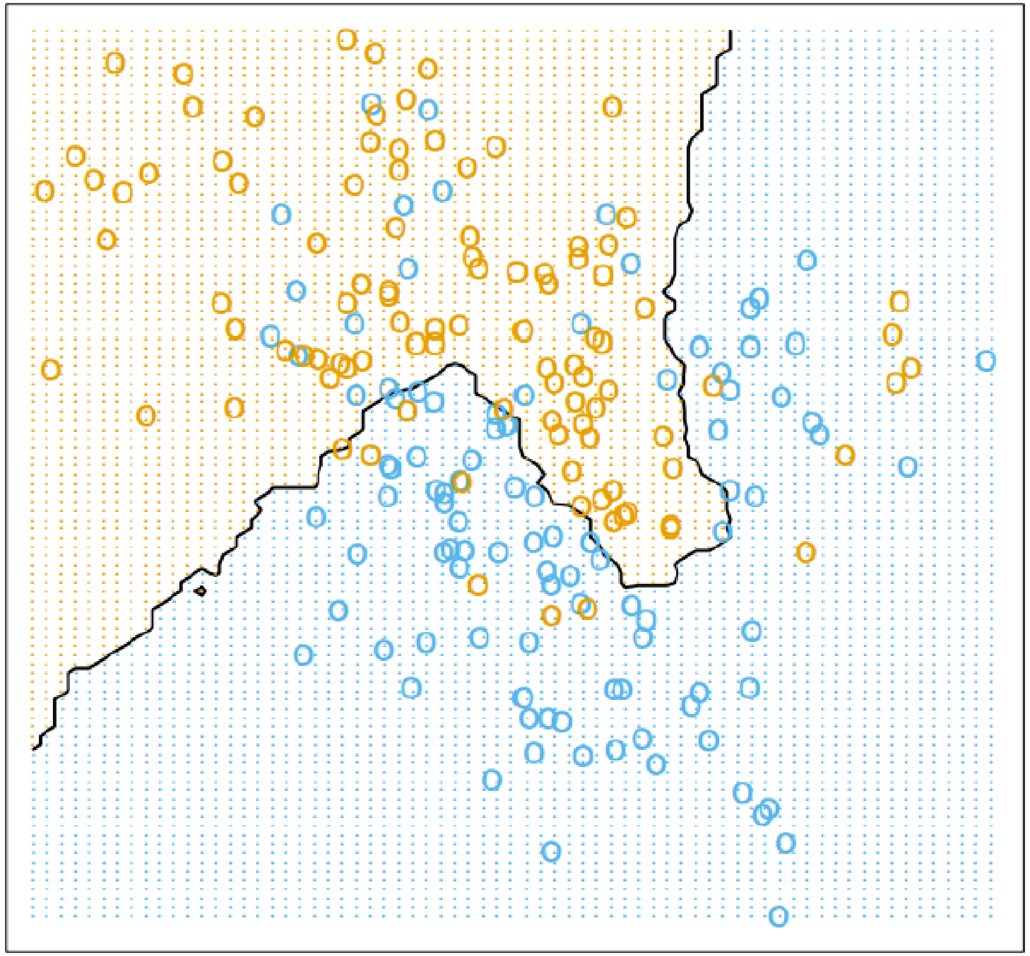
K-Nearest neighbors

k=1



K-Nearest neighbors

k=15



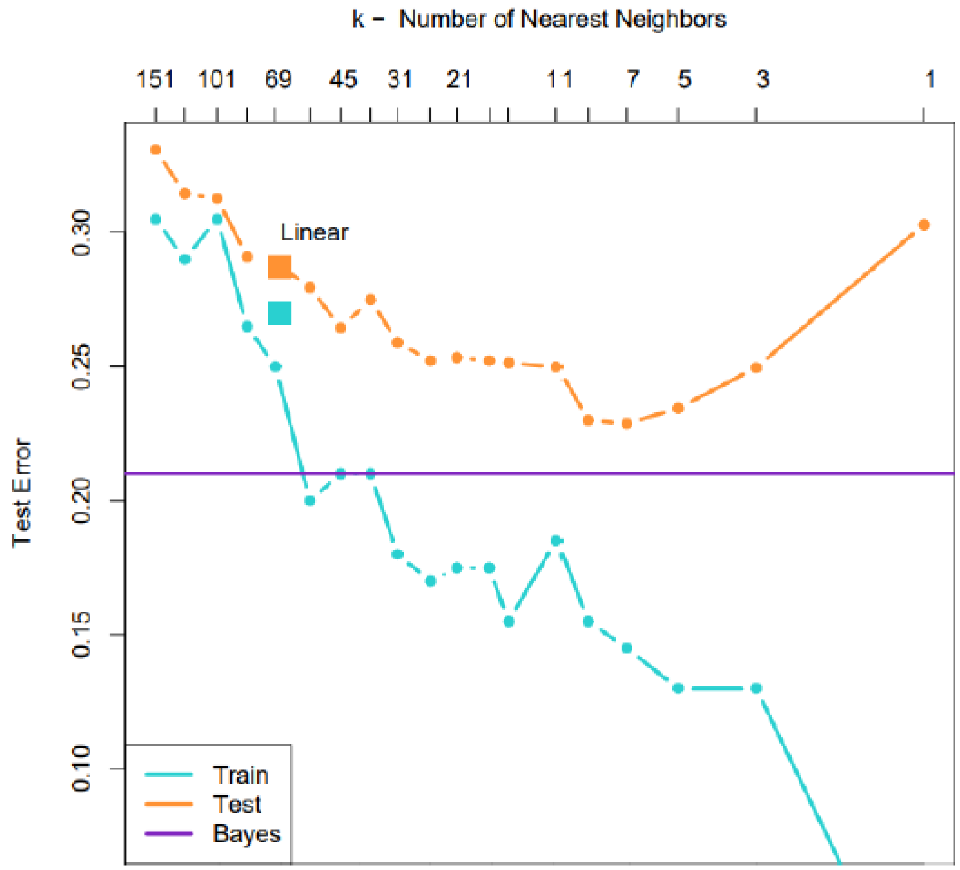
k-Nearest Neighbors

Tradeoffs in choosing k?

- Small k
  - Good at capturing fine-grained patterns
  - May overfit, i.e. be sensitive to random idiosyncrasies in the training data
- Large k
  - Makes stable predictions by averaging over lots of examples
  - May underfit, i.e. fail to capture important regularities
- Balancing k
  - Optimal choice of k depends on number of data points n.
  - Nice theoretical properties if $k \to \infty$ and $\frac{k}{n} \to 0$.
  - Rule of thumb: choose $k < \sqrt{n}$.
  - We can choose k using validation set (next slides).

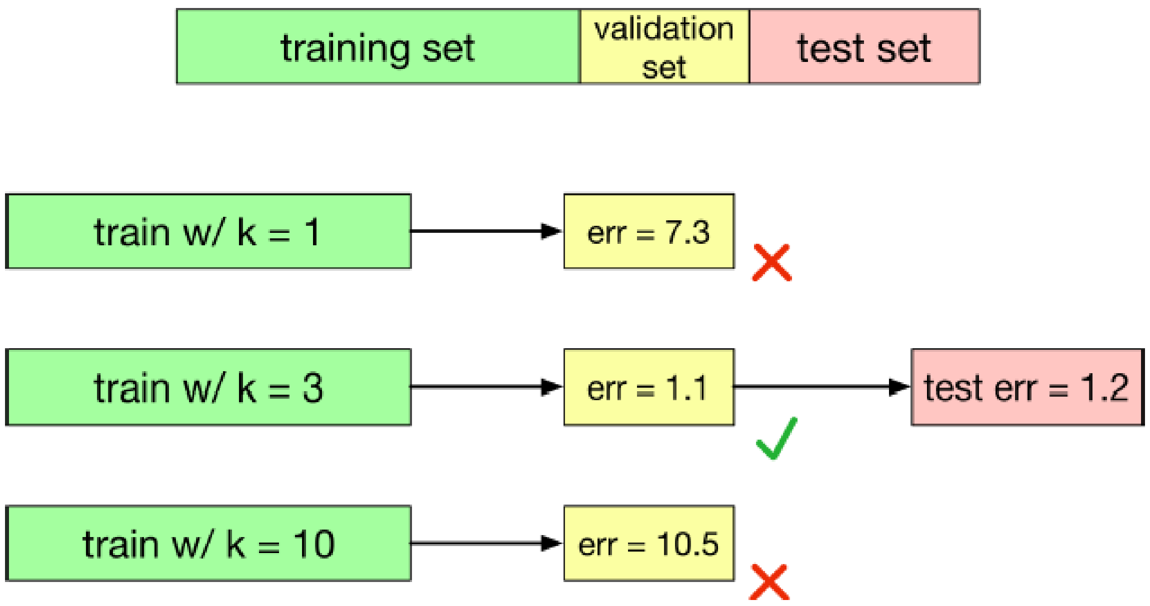
k-Nearest Neighbors

- We would like our algorithm to generalize to data it hasn't seen before.
- We can measure the generalization error (error rate on new examples) using a test set.



k-Nearest Neighbors

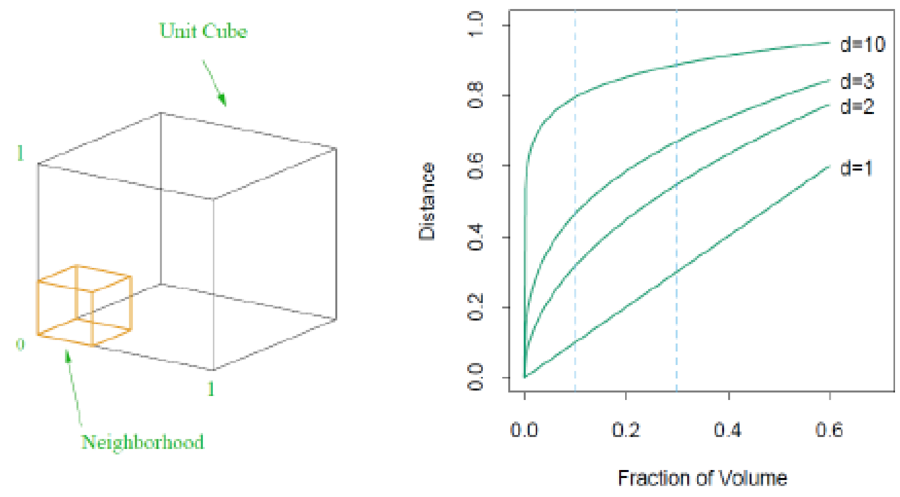
- k is an example of a hyperparameter, something we can't fit as part of the learning algorithm itself
- We can tune hyperparameters using a validation set:



- The test set is used only at the very end, to measure the generalization performance of the final configuration.

Pitfalls: The Curse of Dimensionality

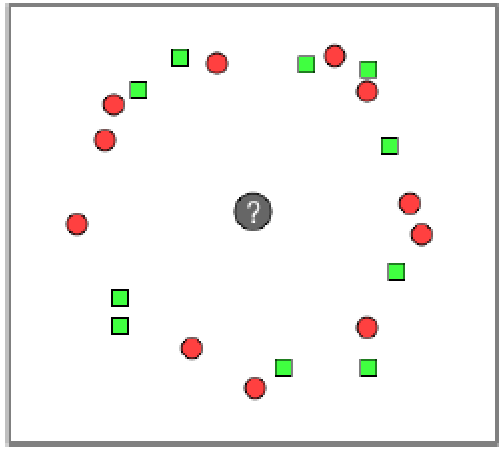
- Low-dimensional visualizations are misleading! In high dimensions, "most" points are far apart.
- If we want the nearest neighbor to be closer than ϵ, how many points do we need to guarantee it?
- The volume of a single ball of radius ϵ is $\mathcal{O}(\epsilon^d)$
- The total volume of $[0,1]^d$ is 1.
- Therefore $\mathcal{O}\left((\frac{1}{\epsilon})^d\right)$ balls are needed to cover the volume.



Pitfalls: The Curse of Dimensionality

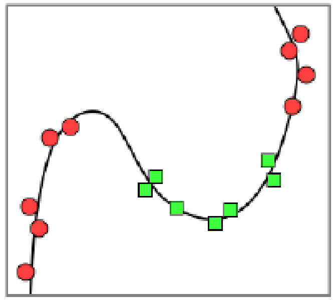
- In high dimensions, "most" points are approximately the same distance.
- We can show this by applying the rules of expectation and covariance of random variables in surprising ways.

- Picture to keep in mind:



Pitfalls: The Curse of Dimensionality

- Saving grace: some datasets (e.g. images) may have low intrinsic dimension, i.e. lie on or near a low-dimensional manifold.



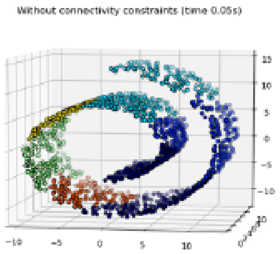
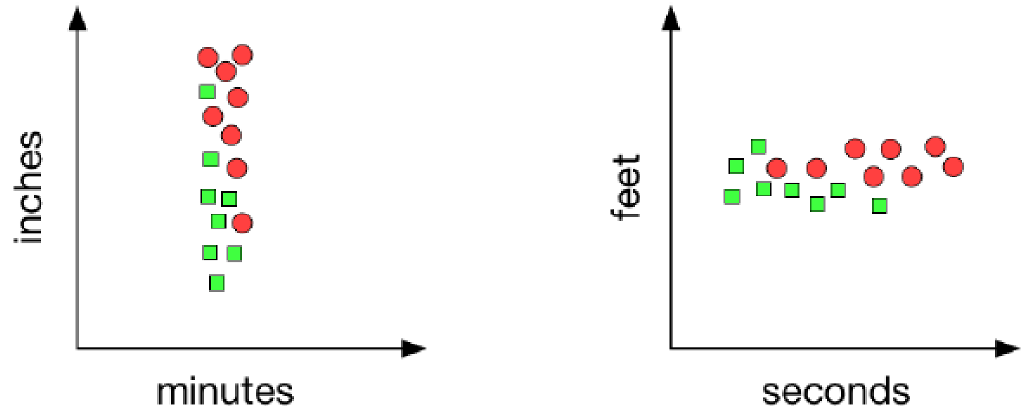


Image credit:
`https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_swiss_roll.html`

- The neighborhood structure (and hence the Curse of Dimensionality) depends on the intrinsic dimension.
- The space of megapixel images is 3 million-dimensional. The true number of degrees of freedom is much smaller.

Pitfalls: The Curse of Dimensionality

- Nearest neighbors can be sensitive to the ranges of different features.
- Often, the units are arbitrary:



- Simple fix: normalize each dimension to be zero mean and unit variance. I.e., compute the mean μ_j and standard deviation σ_j, and take

$$\tilde{x}_j = \frac{x_j - \mu_j}{\sigma_j}$$

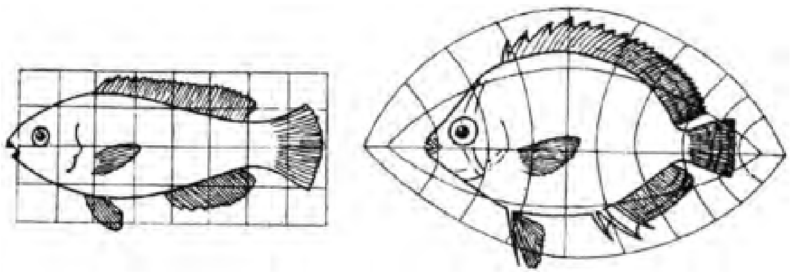
- Caution: depending on the problem, the scale might be important!

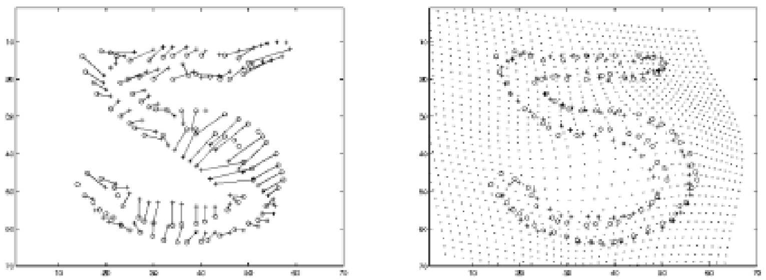
Pitfalls: Computational Cost

- Number of computations at training time: 0
- Number of computations at test time, per query (naïve algorithm)
  - Calculate D-dimensional Euclidean distances with N data points: $\mathcal{O}(ND)$
  - Sort the distances: $\mathcal{O}(N \log N)$
- This must be done for *each* query, which is very expensive by the standards of a learning algorithm!
- Need to store the entire dataset in memory!
- Tons of work has gone into algorithms and data structures for efficient nearest neighbors with high dimensions and/or large datasets.

Example: Digit Classication

- KNN can perform a lot better with a good similarity measure.
- Example: shape contexts for object recognition. In order to achieve invariance to image transformations, they tried to warp one image to match the other image.
  - Distance measure: average distance between corresponding points on *warped* images
- Achieved 0.63% error on MNIST, compared with 3% for Euclidean KNN.
- Competitive with conv nets at the time, but required careful engineering.





[Belongie, Malik, and Puzicha, 2002. Shape matching and object recognition using shape contexts.]

Conclusions

- Simple algorithm that does all its work at test time — in a sense, no learning!
- Can control the complexity by varying k
- Suffers from the Curse of Dimensionality
- Next time: parametric models, which learn a compact summary of the data rather than referring back to it at test time.