CAP 5610: Machine Learning

Lecture 1:
Introduction

Instructor: Dr. Gita Sukthankar
Email: gitars@eecs.ucf.edu

Reference Materials From: GuoJun Qi, Fei Liu, Tom Mitchell

Agenda

- Course information
- Homework (financial aid): Machine Learning Research Interests
- Introduction to ML

Contact Info

- Office Hours: T 1:30-3:00pm, Th 10:00-11:30am in HEC 232 (but please send email as needed)
- TA: Neda Hajiakhoond Bidoki (hajiakhoond@knights.ucf.edu)

Prerequisites

- Undergraduate AI course (UCF: CAP 4630) OR
- Commensurate background in computer vision, pattern recognition, machine learning, statistics OR
- Online machine courses

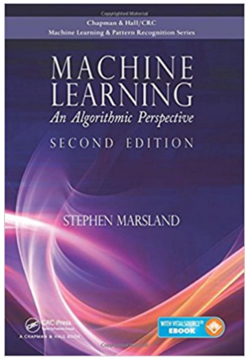
You need to know the following:

- A programming language, preferably python
- Basic knowledge of probability/statistics, linear algebra, and calculus.

You can supplement your knowledge with online tutorials.

Textbook

- Recommended (Optional):
  - Machine Learning: An Algorithmic Perspective (2nd edition), Stephen Marsland, 2014
  - We will cover everything except Chap 5, 10,17, and 18
- Online References (Webcourses):
  - Pattern Recognition and Machine Learning, Chris Bishop, 2006
  - Machine Learning: A Probabilistic Perspective, Kevin Murphy, 2012
  - Introduction to Deep Learning, Goodfellow, Bengio, and Courville, 2016



Webcourses

- All lectures will be posted as pdfs and will form the major component of what appears on the exam.
- Make sure your email settings are correct so we can use webcourses to contact you as needed.

Evaluation

- Homework (45%): Machine Learning Implementation/Evaluation
  - Three homeworks (15%) each
  - Code plus written summary of results
- Midterm Exam (25%)
  - In class exam based on lecture slides with math problems and short answer questions
- Final Project (30%)
  - Literature survey on a machine learning topic OR
  - Technical report (CS conference paper):
    - Introduction, Problem Description, Method, Results, Conclusion
  - Presentations will take place during the final exam period (Dec 5, 10-1pm)
  - Most popular topic choices: variational auto encoders, GANs, deep RL

Grading Policy

- +/- grades are awarded
- Assignments should be submitted in a timely fashion via webcourses by midnight on the due date.
- Late assignments will be penalized by 25% per day.
  - Unpopular but necessary to help TA.
- You are expected to abide by UCF's plagiarism and cheating policies.
- Any code obtained from other sources must be documented appropriately.

Tips

- Pick a machine learning book that you like and start reading it.
  - If you prefer online blogs, those are great too!
- Start the homework assignments immediately; submit them in a timely fashion.
- Remember to study for the exam
- Pick a final project that interests you and plays to your strengths.
- Remember to allocate time to write your final paper and create your presentation.

Topics

- Machine learning training protocols and data preparation
- Simple supervised classifiers: k nearest neighbor and decision trees
- Naive Bayesian classifier
- Linear and logistic regression
- Support vector machine and kernel methods
- Neural networks and deep learning
- Unsupervised learning: clustering and PCA
- Reinforcement learning
- Model fitting and EM
- Ensemble learning

Topic list is standard to what you'll find in any statistical machine learning textbook:
Plus the following:

- Reinforcement learning
- Probabilistic models
- Deep learning
- Applications

Research Interests

- Undergrad/masters/Ph.D.?
- CS vs. non CS?
- Completed online classes?
- Conducting research in machine learning?
- Research interests:
    - Computer vision/robotics
    - Natural language processing
    - Social media data mining
    - Evolutionary algorithms
    - Medicine
    - Recommender systems/marketing/business
    - Other science/engineering

Machine Learning Research Interests

- Ungraded assignment used to document student engagement for financial aid
- Due Aug 27
- Submit 1-2 sentences describing your interest in machine learning (or why you decided to take the class)

What is machine learning?

What is machine learning?

<u>Herbert Simon</u>: "Learning is any process by which a system improves performance from experience."

<u>Arthur Samuel</u>: "Field of study that gives computers the ability to learn without being explicitly programmed."

<u>Tom Mitchell</u>: "Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

What can machine learning do?

- A supervised learning **task**

Data (patient records) → A prediction task: will the patient suffer an emergent C-section?

Age: 23
FirstPregnancy: no
Anemia: no
Diabetes: no
PreviousPrematureBirth: no
Ultrasound: ?
Elective C–Section: ?
Emergency C–Section: ?
...

T. Mitchell's notes

Past experiences

- You already know some emergent (non-emergent) C-section cases

Positive example

Emergent C-section

Age: 23
FirstPregnancy: no
Anemia: no
Diabetes: YES
PreviousPrematureBirth: no
Ultrasound: abnormal
Elective C–Section: no
Emergency C–Section: ?
...

Negative example

Non-emergent C-section

Age: 23
FirstPregnancy: no
Anemia: no
Diabetes: no
PreviousPrematureBirth: no
Ultrasound: ?
Elective C–Section: ?
Emergency C–Section: ?
...

Machine learning aims to:

- Extract knowledge from the past experiences and predict information about the future cases
- **Training** set of examples: Past experience (labeled examples)
- **Test** set of examples: future cases to predict on (unlabeled examples)
- A **model** is trained from the training set, which summarizes the knowledge from the past experience

An example of the **model**

- Rule-based model for predicting emergent C-section

```
If   No previous vaginal delivery, and
     Abnormal 2nd Trimester Ultrasound, and
     Malpresentation at admission
Then Probability of Emergency C-Section is 0.6
```

- Applying the model to predict information about the future case

Input and output

- Input: training set
  - Training set = {(x_i, y_i)|x_i is the data, y_i is the label}
- Output:
  - Model can be viewed as a function, which maps data x to label y

$$y=h(x): X \rightarrow Y$$

  - The set of all possible functions constitute hypotheses H={h|h:X->Y}

- test set = {(x_j, ?)|x_j the data whose label will be predicted by the trained model}

Oracle function

- Assume we have an oracle function h_o which always outputs a correct prediction on an input data x
- Machine learning algorithms aim to find a function h from a set of hypotheses H to approximate this oracle function as well as possible

$$h^* = \min_{h \in H} E_{x \sim D} err\big(h(x), h_o(x)\big)$$

where *E* denotes the expectation, *D* is the distribution of all possible examples in the real world, and err is a function measuring the discrepancy between the outputs from h and oracle function h_o.

What's the challenge?

- Ideal objective of machine learning algorithm

$$h^* = \min_{h \in H} E_{x\sim D} err\big(h(x), h_o(x)\big)$$

- Solution: using training set to approximate the objective

$$E_{x\sim D} err\big(h(x), h_o(x)\big) \approx \frac{1}{n}\sum_{i=1}^{n} err(h(x_i), y_i)$$

How good is the approximation?

- Using the sample mean to approximate the distribution mean

$$E_{x\sim D}err\big(h(x), h_o(x)\big) \approx \frac{1}{n}\sum_{i=1}^{n} err(h(x_i), y_i)$$

- The law of large number: the sample mean will approach to the distribution mean as n goes to infinity (asymptotically).
- Learning theory: quantifying the discrepancy between sample mean and distribution mean of error under a given number of training examples

Some notes

- We do not require that the oracle function must belong to hypothesis set H $y \neq h_o(x)$

- The training set may have noise
  - the output y of an input x may not be correct

Choose error/loss function

- Depending on the nature of output variable
  - Discrete value {0,1,…,C}: err(h(x),y) is 0 if h(x) and y is the same, or 1 otherwise
  - Continuous value, squared difference err(h(x),y)= $(h(x) - y)^2$
  - Vector of continuous numbers, squared Euclidean distance err(h(x),y)

More examples

- Handwritten digit recognition (zip code)
  - MNIST (Mixed National Institute of Standards and Technology) dataset
  - 60,000 training examples: written by American census Bureau employees
  - 10,000 test examples: written by American high school students
  - recognizing the digits from 0 to 9.

- How good is machine learning algorithm on this task?
  - Best performance: 0.27% test error (better than human performance)

How to represent the data in computer?

- Data representation that can be processed by computer.

Age: 23
FirstPregnancy: no
Anemia: no
Diabetes: YES
PreviousPrematureBirth: no
Ultrasound: abnormal
Elective C–Section: no
Emergency C–Section: ?
...

A table of attributes
Integer: 23
Boolean: No
Boolean: No
Boolean: YES
Boolean: No
Enumeration: Abnormal
Boolean: No

Feature extraction

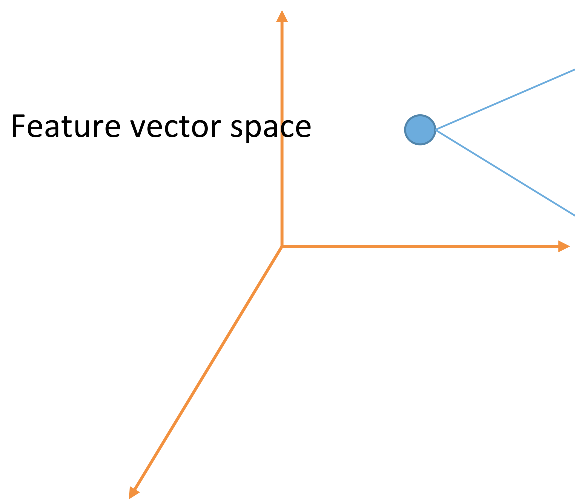
- Data representation for hand-written digits

Table of attributes
Pixel value: 0
Pixel value: 255
Pixel value: 255
Pixel value: 255
Pixel value: 0
Pixel value: 0
Pixel value: 0

Feature vectors

Feature vector space

- Each point in the space represents a **feature vector** whose entry is an attribute of the data.



A feature vector
Integer: 23
Boolean: No
Boolean: No
Boolean: YES
Boolean: No
Enumeration: Abnormal
Boolean: No

Principle of feature extraction

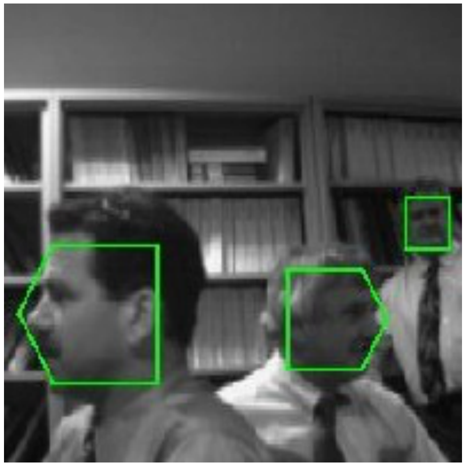
- Extracting features that are relevant to the defined task
  - MNIST – pixel values in the image
  - Emergent C-section – age, first pregnant, anemia,
- Domain knowledge
  - Image processing
  - Medicine

What machine learning cannot do

- Garbage in, garbage out
  - All features are completely irrelevant to the task, machine learning can do nothing for you.
- Good features play the key role in machine learning
  - Domain knowledge can be helpful in identifying features.
- Open source software makes it easy to implement machine learning algorithms but testing the algorithms requires more skill than implementing them.

Other applications

- Face detection & recognition
- Object detection & recognition
- Speech recognition
- Webpage classification
- Spam email detection
- ...



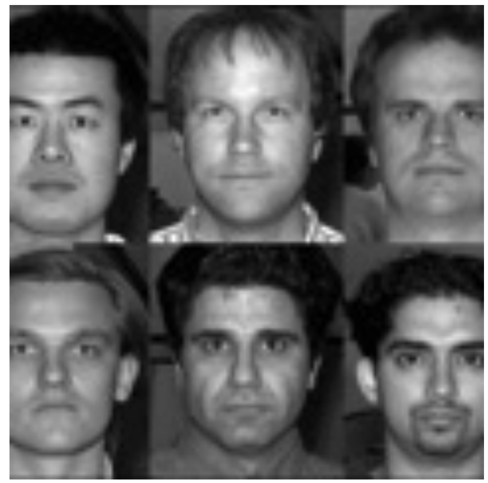
Input: image
Output: face location

Where is the face?

From CMU face detection project

Other applications

- Face detection & recognition
- Object detection & recognition
- Speech recognition
- Webpage classification
- Spam email detection
- ...



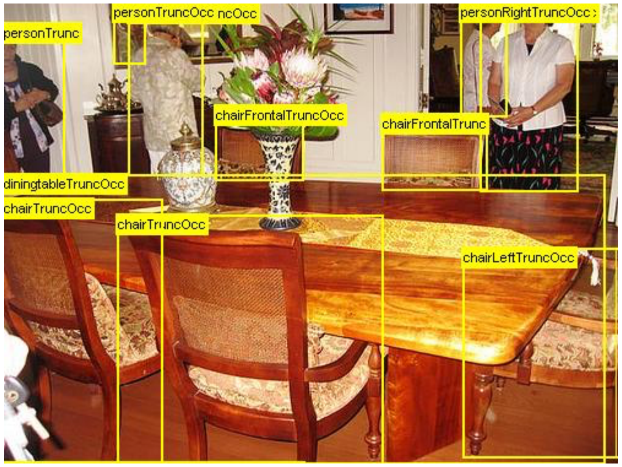
Input: face
Output: identity

Who is this?

From Yale face dateset

Other applications

- Face detection & recognition
- Object detection & recognition
- Speech recognition
- Webpage classification
- Spam email detection
- …



Localize and classify objects in image

From PASCAL VOC 2012 dataset

Other applications

- Face detection & recognition
- Object detection & recognition
- Speech recognition
- Webpage classification
- Spam email detection
- ...

From speech to text

Input: cepstral coefficients
Output: words/sentences

Other applications

- Face detection & recognition
- Object detection & recognition
- Speech recognition
- Webpage classification
- Email spam detection
- …

Company home page

vs

Personal home page

vs

University home page

vs

…

Input: webpage caption, URL, keywords, incoming/outgoing links

Output: webpage category (company, personal, university)

Other applications

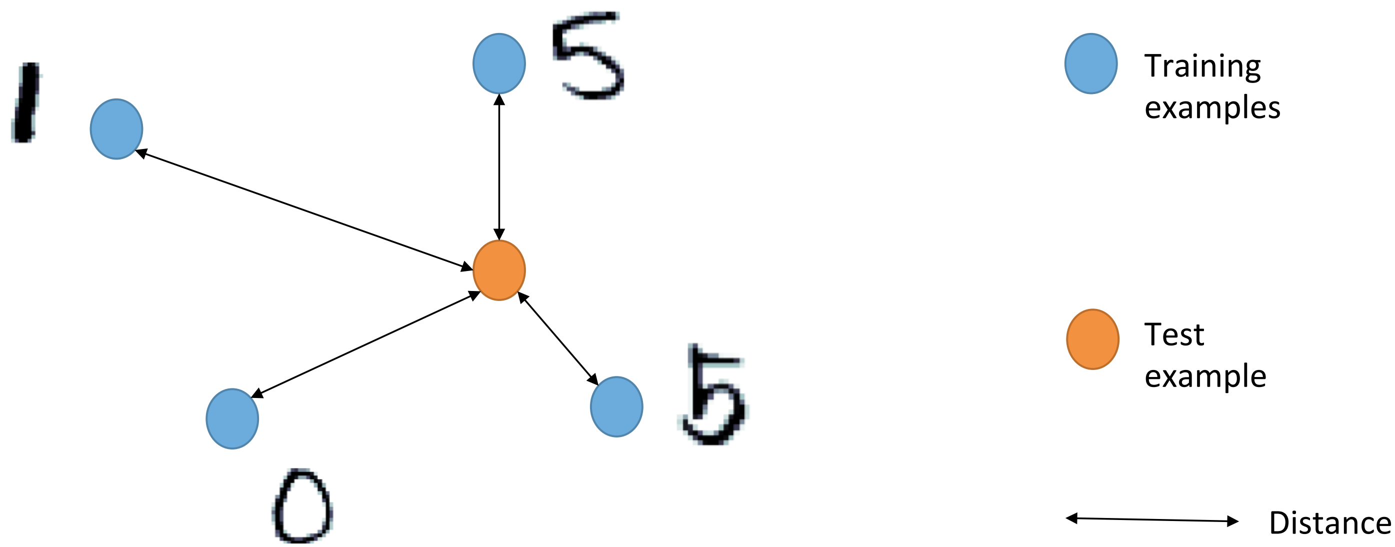
- Face detection & recognition
- Object detection & recognition
- Speech recognition
- Webpage classification
- Email spam detection
- …

Input: sender, length, keywords …

Output: spam/nonspam

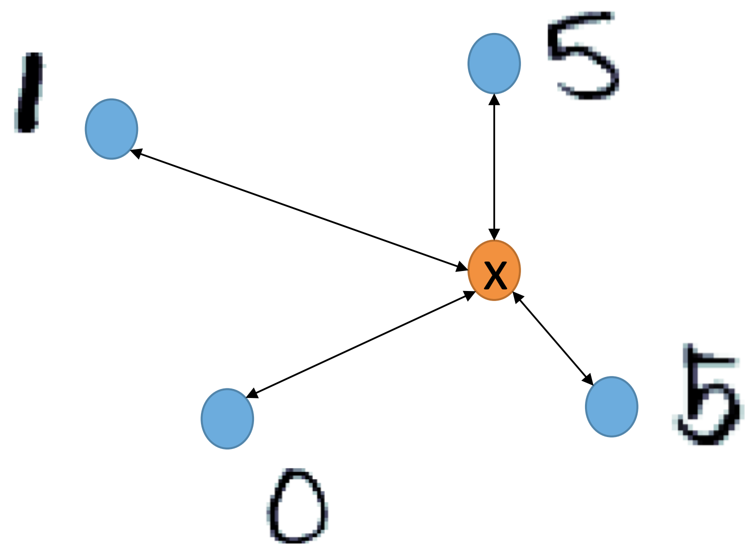
A simple machine learning algorithm

- K nearest neighbor (KNN)



A simple machine learning algorithm

- K nearest neighbor (KNN)



Given a test example x, its label is predicted as the most frequent label among K training examples nearest to x.

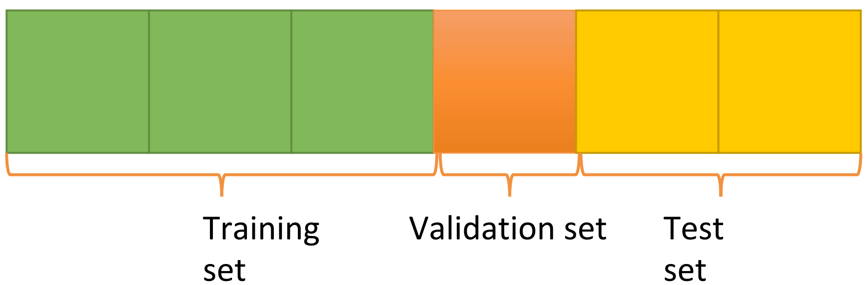
Test Protocol

Test protocol (1)

- Typical setting: Split the data into training/test set
  - Apply the knowledge from training set to predict on the test set
  - Computing the prediction accuracy or error on the test set

Test protocol (2)

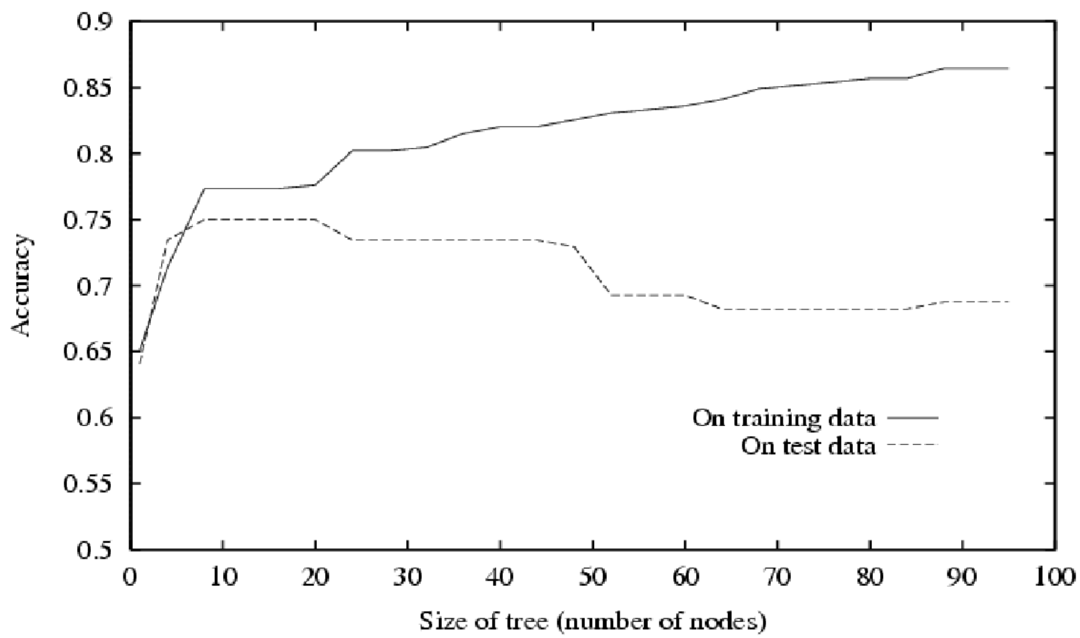
- When you have parameters to tune: split into training/validation/test sets
  - Tune the parameters of the algorithm on validation set (e.g., K in KNN)
  - Choose the parameters which give the best performance on validation set



Why not tune on training set?

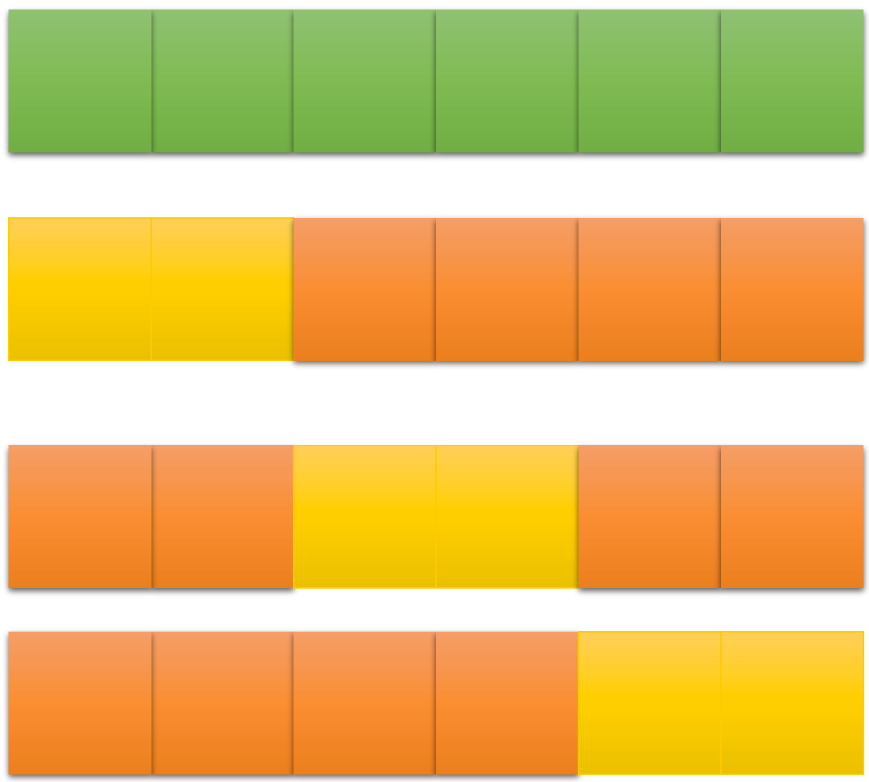
- Overfitting problem
  - The goal of a machine learning model is to generalize to unseen data (test examples) to predict their performance
  - Tuning on training set runs risk of fitting the model too much to training examples, trapping the model to overemphasize past experience

Overfitting



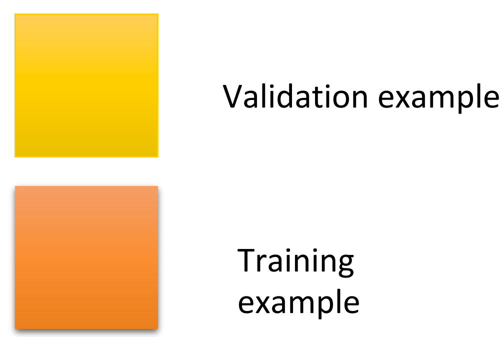
Example extracted from T. Mitchell's slides

k-Fold Cross-Validation

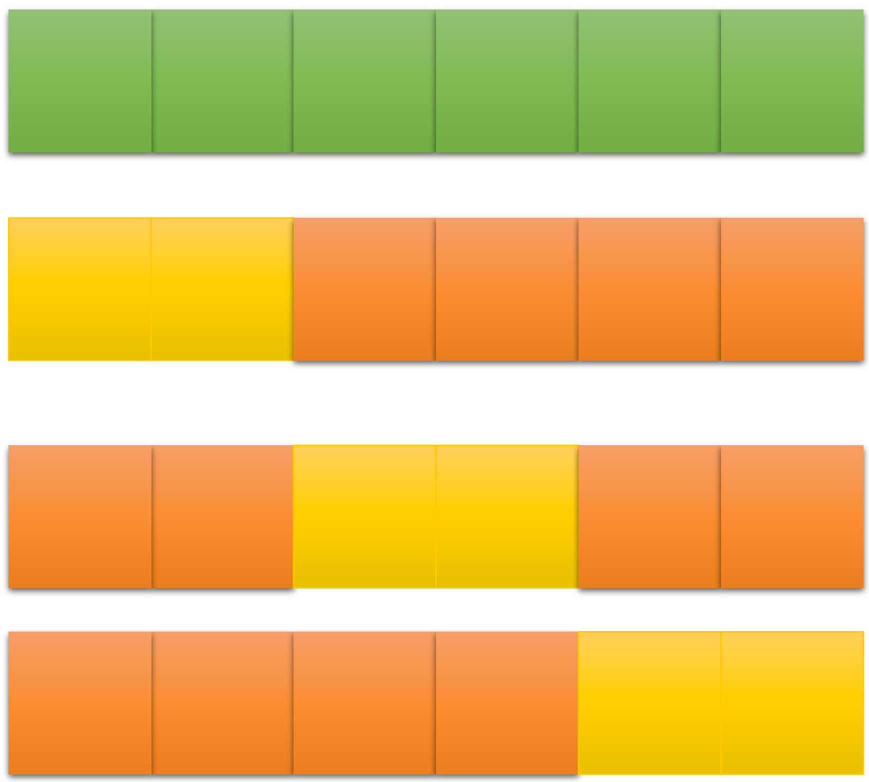


k-fold cross-validation

- Fix the parameter to be tuned
- Divide dataset into k subsets of equal size
- Every time, use k-1 subsets to train a model
- Test the trained model on the remaining subset of validation examples
- Repeat k times
- Average the accuracy/error over k times experiments

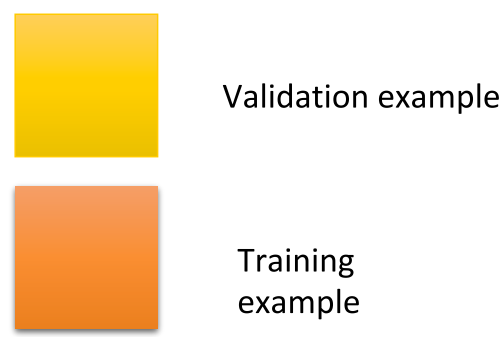


Special case: Leave one out (LOO) cross-validation



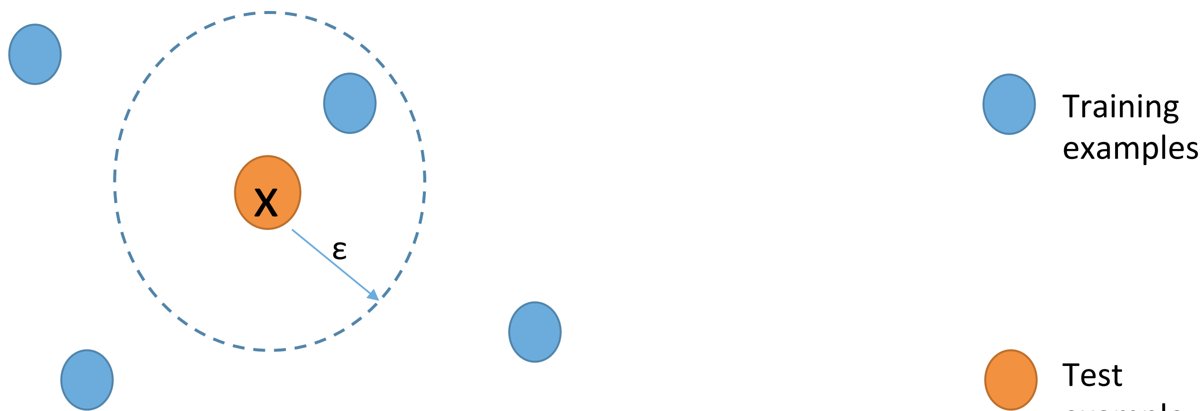
Set k=n

- Every time, only one example is left for validation, others are used as training examples
- Used for small datasets



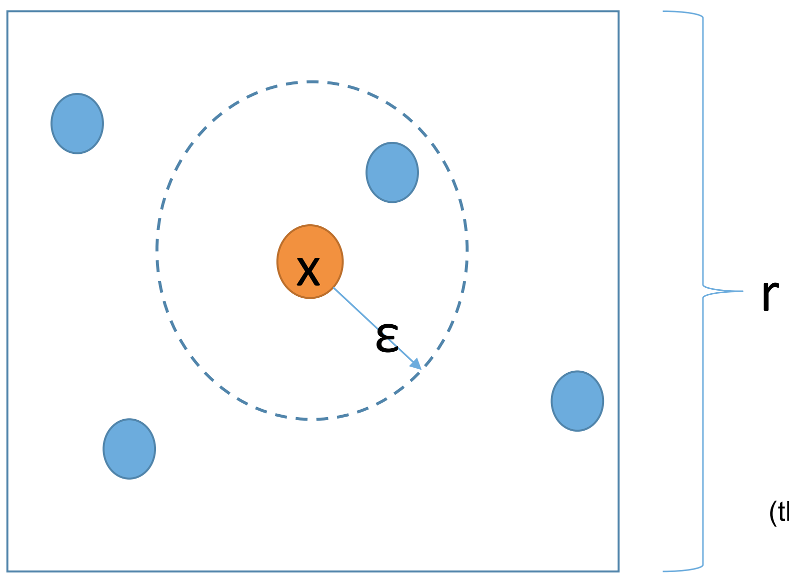
Curse of Dimensionality

- Consider a variant nearest neighbor algorithm – ε-nearest neighbor
  - Search for all training examples within a sphere of radius ε centered at test example x in D-dimensional feature vector space
  - Test example is predicted by majority voting among the training examples in this sphere



Curse of Dimensionality

- Assume all training examples are uniformly distributed in a hypercube of size r (we should have $2\varepsilon < r$), how likely does a training example fall into the ε-sphere in D-dimensional space?



Volume of sphere (n-ball) $\mathrm{V}_{\mathrm{sphere}} = \dfrac{2\varepsilon^D \pi^{D/2}}{D\Gamma(D/2)}$

Volume of hypercube $V_{hypercube} = r^D$

The chance that an example falls into the sphere

$\dfrac{V_{sphere}}{V_{hypercube}} \to 0$, as D goes to infinity.

(think about how the formula changes moving from 2D to 3D)

Curse of Dimensionality

- Exponential volume increase means that data sample becomes inadequate
- No training example will fall into the sphere neighborhood as dimensionality goes to infinity.
- Other explanations:
  - Euclidean distances become indiscernible as dimensionality goes big.
  - K. Beyer, J. Goldstein, R. Ramakrishnan, U. Shaft. (1999). "When is “Nearest Neighbor” Meaningful?". Proc. 7th International Conference on Database Theory - ICDT'99.

Next Time

Review of probability theory and distributions

References to Read

All these references cover the same thing

- Chap 1-2: Marsland's Machine Learning
- Chap 1: Bishop's Pattern Recognition and Machine Learning
- Chap 1: Murphy's ML: Probabilistic Approach