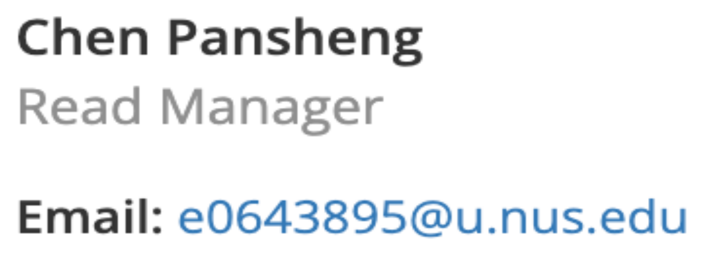
EE5907 Pattern Recognition
EE5027 Statistical Pattern Recognition

Robby Tan

Topics

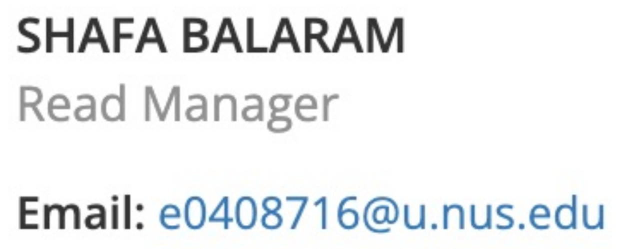
- Robby Tan (EE5907 Part I + EE5027)
  - Contact: robby.tan@nus.edu.sg
  - Bayesian Inference
  - Linear Models for Regression and Classification
- Bai Soon (EE5907 Part II + EE5026)
  - LDA, SVM, PCA
  - Clustering and Applications
  - Deep learning

GA: EE5027 + First Half of EE5907



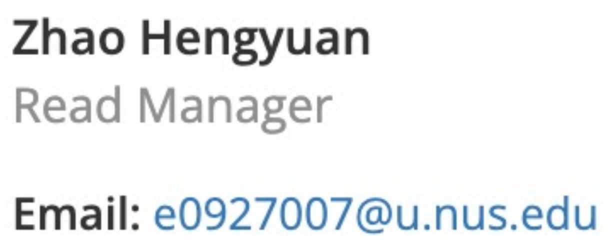
Chen Pansheng
Read Manager

Email: e0643895@u.nus.edu



SHAFA BALARAM
Read Manager

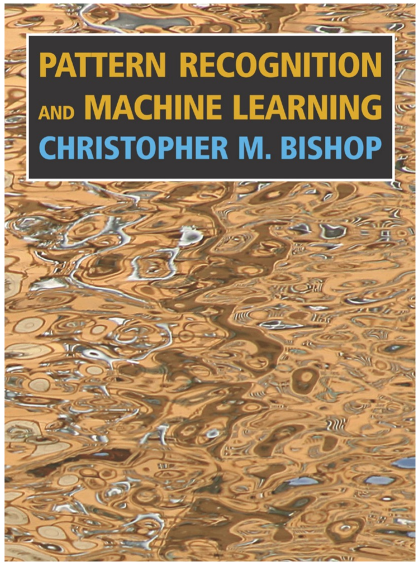
Email: e0408716@u.nus.edu

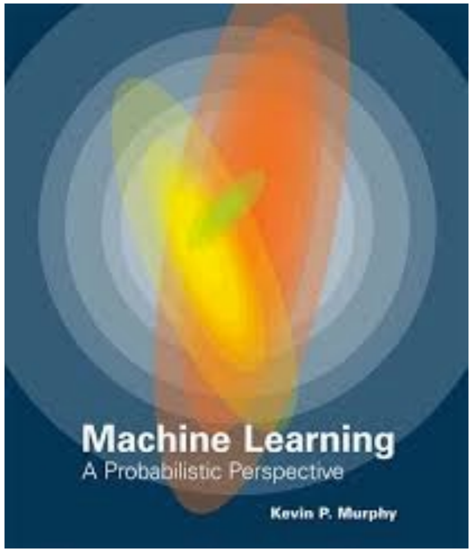


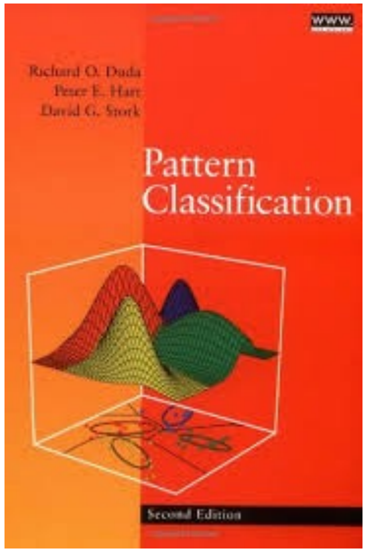
Zhao Hengyuan
Read Manager

Email: e0927007@u.nus.edu

Main References







PRML -- free download : https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/

Computer Vision: Models, learning and inference, Simon Prince, 2012
Free: www.computervisionmodels.com

Pre-requisites

- Pre-requisites
  - Probability, statistics, and linear algebra (vector spaces and matrix theory) as taught in typical undergraduate courses
  - Probability and statistics are <u>especially</u> important for first half of EE5907 + EE5027
  - Programming in Python (using Jupyter Notebook)
- <u>You will find this class very difficult if you don’t have the pre-requisites</u>
- <u>If this is your first graduate class, you will also find it to be substantially more difficult than your undergrad courses</u>

State-of-the-Art

- EE5907/EE5027/EE5026 do NOT cover state-of-the-art. Teach concepts needed to understand state-of-the-art

- For state-of-the-art:
  - Conference papers: CVPR, ICCV, ECCV, ICML, ICLR, NeurIPS, etc
  - Journal articles: IJCV, T-IP, JMLR, T-PAMI, etc
  - EE6733: Advanced Topics on Vision and Machine Learning

- EE5934/EE6934: Deep Learning

LumiNUS and Website

- Lecture notes will be made available on the course website (the first part): https://tanrobby.github.io/teaching/ece_pattern_recognition/index.html
- Assignments are submitted via LumiNUS
- Deadlines are strict:
  - Late submission will be deducted 2 points (out of 10) for every 24 hours.

Assessments

- EE5907
  - 2 CAs (2 x 20%)
    - Individual projects (absolutely no copying permitted)
  - Final Exam (60%)
    - More information to come
- EE5027
  - 1 CA (40%) same as EE5907 CA1
  - Final exam (60%)
    - More information to come

Academic Integrity

Academic honesty is compulsory in finishing the assignments and the exams:

1. Exchanging codes is not allowed.
2. Using codes from the previous years or from the internet is prohibited, unless stated otherwise in the lectures.
3. Copying texts of the reports from other groups is strictly prohibited.
4. Generally, cheating, academic misconduct, plagiarism, and fabrication of any submitted material (including code and text) are not tolerated.

Any violation to the academic honesty will imply failure to pass the course.

Introduction

What is Artificial Intelligence (AI)?

- AI = intelligence displayed by machines
- Unsolved since 1950s
- Recent resurgence made possible by machine learning and deep learning

What is Pattern Recognition / Machine Learning?

- Will use both terms interchangeably
  - Pattern Recognition: automatically detect patterns from data
  - Machine Learning: Learn from data without being explicitly programmed

What is Pattern Recognition / Machine Learning?

- Will use both terms interchangeably
  - Pattern Recognition: automatically detect patterns from data
  - Machine Learning: Learn from data without being explicitly programmed
- Applications
  - Email SPAM detection
  - Speech recognition (e.g., SIRI)
  - Handwritten digit recognition (e.g., sorting snail mail)

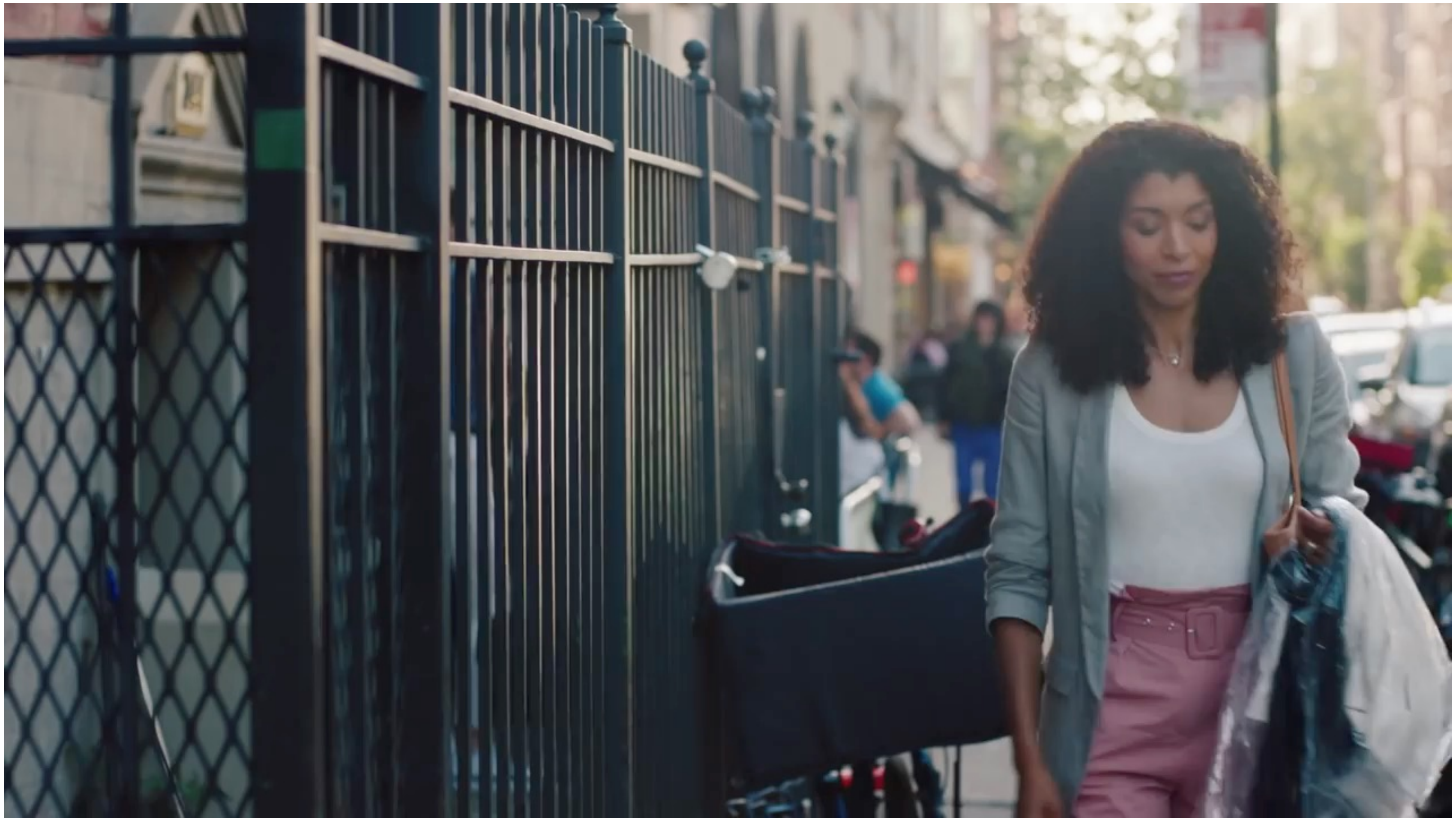
What is Pattern Recognition / Machine Learning?

- Will use both terms interchangeably
  - Pattern Recognition: automatically detect patterns from data
  - Machine Learning: Learn from data without being explicitly programmed
- Applications
  - Email SPAM detection
  - Speech recognition (e.g., SIRI)
  - Handwritten digit recognition (e.g., sorting snail mail)
  - Google search
  - Driverless cars
  - Banking fraud detection

What is Pattern Recognition / Machine Learning?

- Will use both terms interchangeably
  - Pattern Recognition: automatically detect patterns from data
  - Machine Learning: Learn from data without being explicitly programmed
- Applications
  - Email SPAM detection
  - Speech recognition (e.g., SIRI)
  - Handwritten digit recognition (e.g., sorting snail mail)
  - Google search
  - Driverless cars
  - Banking fraud detection
  - Detection and diagnosis of diseases
  - Biometric (e.g., fingerprint security)
  - DNA sequence identification (e.g., Counsyl)
  - Manufacturing (e.g., machine vision)

Google Assistant



Amazon Go

Types of Machine Learning

Supervised Learning

- Given input-output pairs $D = \{x_i, y_i\}_{i=1:N}$, learn mapping $y = f(x)$
  - D = training set

Supervised Learning

- Given input-output pairs $D = \{x_i, y_i\}_{i=1:N}$, learn mapping $y = f(x)$
  - D = training set
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vector (called features)

Supervised Learning

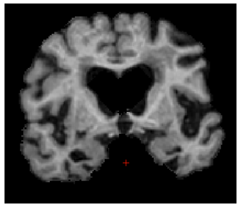
- Given input-output pairs $D = \{x_i, y_i\}_{i=1:N}$, learn mapping $y = f(x)$
  - D = training set
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vector (called features)
  - y also called output or target variable
    - y discrete => problem known as classification
    - y continuous => problem known as regression

Supervised Learning

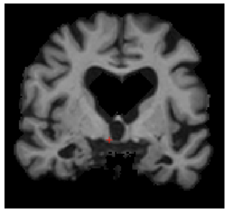
- Given input-output pairs $D = \{x_i, y_i\}_{i=1:N}$, learn mapping $y = f(x)$
  - D = training set
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vector (called features)
  - y also called output or target variable
    - y discrete => problem known as classification
    - y continuous => problem known as regression

Training: Learn relationship between inputs (MRI) & target labels (AD or healthy)

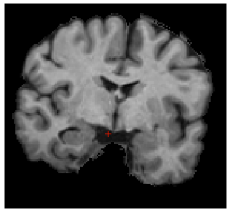
Alzheimer's Disease (AD)



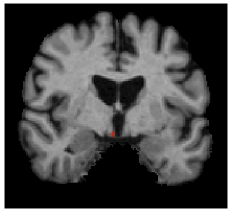
…



Healthy



…



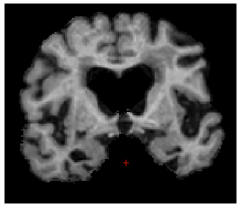
x = MRI images, y = AD or healthy

Supervised Learning

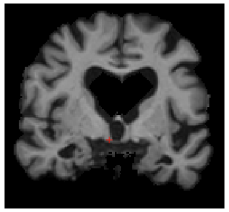
- Given input-output pairs $D = \{x_i, y_i\}_{i=1:N}$, learn mapping $y = f(x)$
  - D = training set
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vector (called features)
  - y also called output or target variable
    - y discrete => problem known as classification
    - y continuous => problem known as regression

Training: Learn relationship between inputs (MRI) & target labels (AD or healthy)

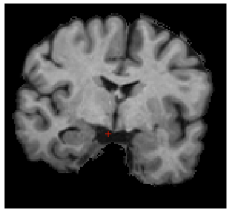
Alzheimer's Disease (AD)



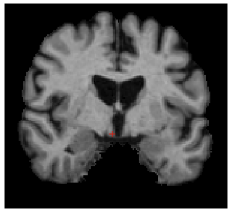
…



Healthy

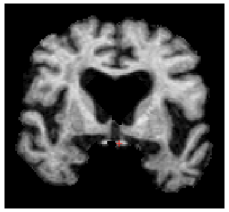


…

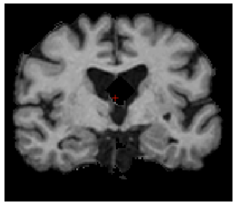


x = MRI images, y = AD or healthy

Testing: Given new MRI, Predict AD or healthy



Healthy or AD?



Healthy or AD?

Unsupervised Learning

- Given $D = \{x_i\}_{i=1:N}$, discover "interesting structure" in data
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vectors

Unsupervised Learning

- Given $D = \{x_i\}_{i=1:N}$, discover "interesting structure" in data
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vectors
  - "cheap" since labels (i.e., y's) typically expensive to get

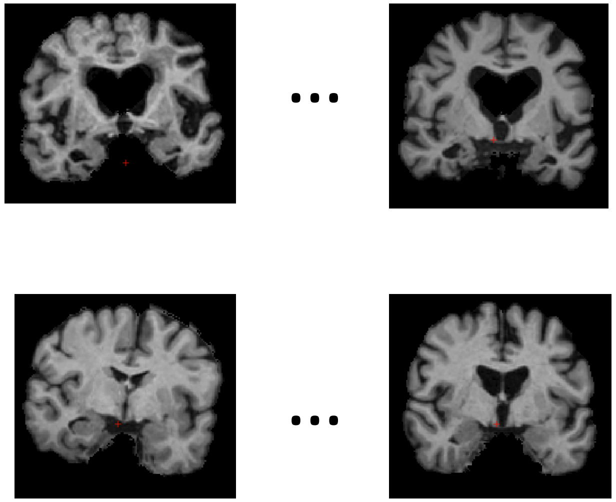
Unsupervised Learning

- Given $D = \{x_i\}_{i=1:N}$, discover "interesting structure" in data
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vectors
  - "cheap" since labels (i.e., y's) typically expensive to get
  - More open-ended, no groundtruth – validation more difficult

Unsupervised Learning

- Given $D = \{x_i\}_{i=1:N}$, discover "interesting structure" in data
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vectors
  - "cheap" since labels (i.e., y's) typically expensive to get
  - More open-ended, no groundtruth – validation more difficult

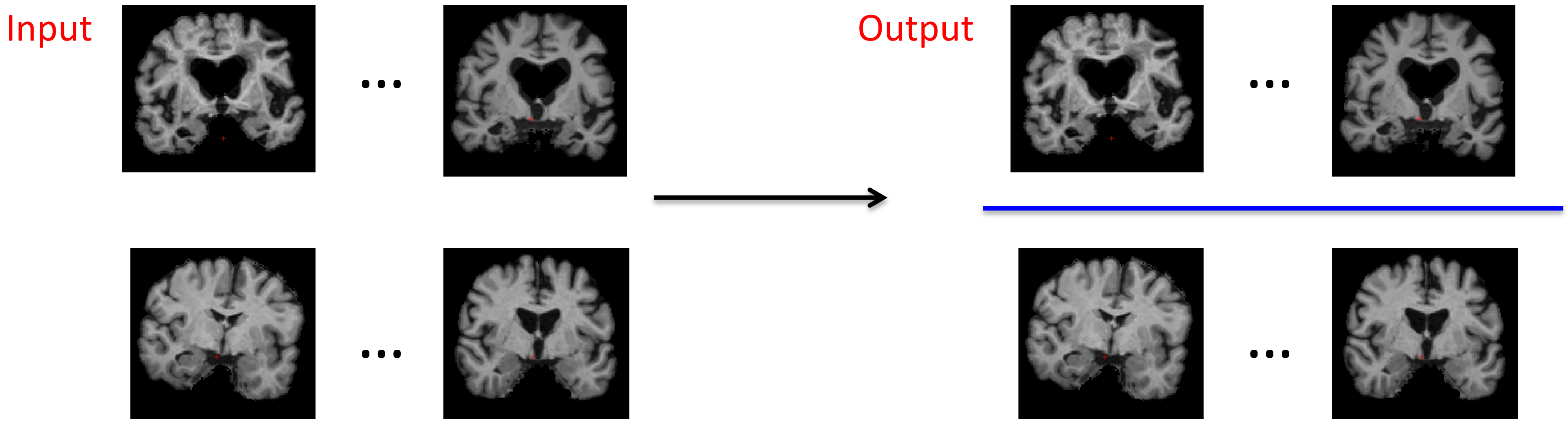
Input



x = MRI images

Unsupervised Learning

- Given $D = \{x_i\}_{i=1:N}$, discover "interesting structure" in data
  - x (images, emails, molecular shapes, etc) typically represented as p-dimensional vectors
  - "cheap" since labels (i.e., y's) typically expensive to get
  - More open-ended, no groundtruth – validation more difficult



x = MRI images

Types of Machine Learning

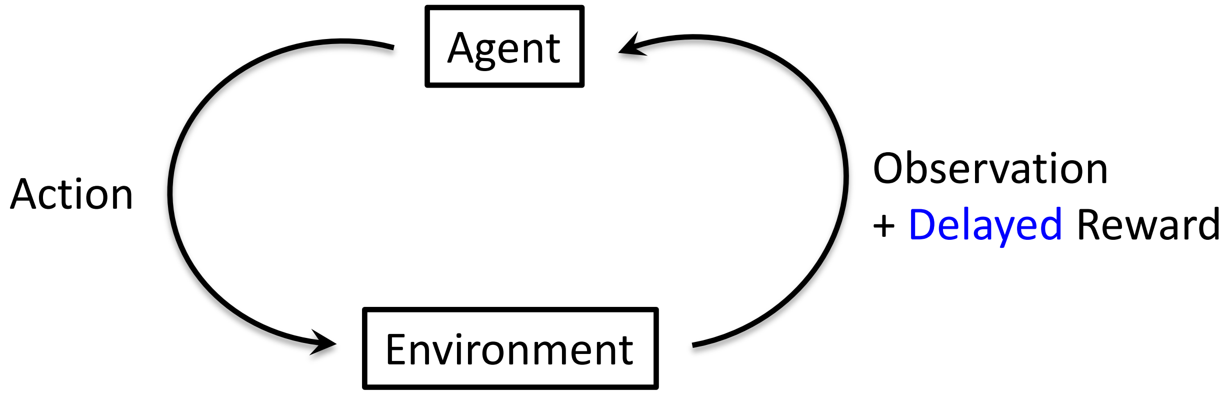
- Summary
  - Supervised learning: Great if we know what we want to predict
  - Unsupervised learning: Great if we want to discover something new

Types of Machine Learning

- Summary
  - Supervised learning: Great if we know what we want to predict
  - Unsupervised learning: Great if we want to discover something new
- Other learning not covered in this class
  - Semi-supervised: some data has target labels, some don’t

Types of Machine Learning

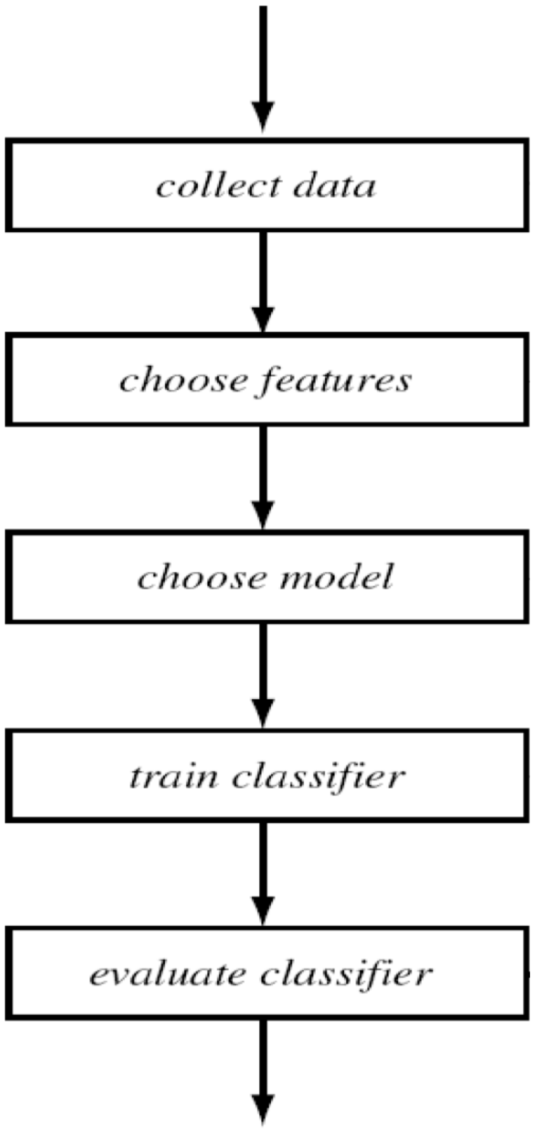
- Summary
  - Supervised learning: Great if we know what we want to predict
  - Unsupervised learning: Great if we want to discover something new
- Other learning not covered in this class
  - Semi-supervised: some data has target labels, some don't
  - Reinforcement learning: learn actions by trial & error



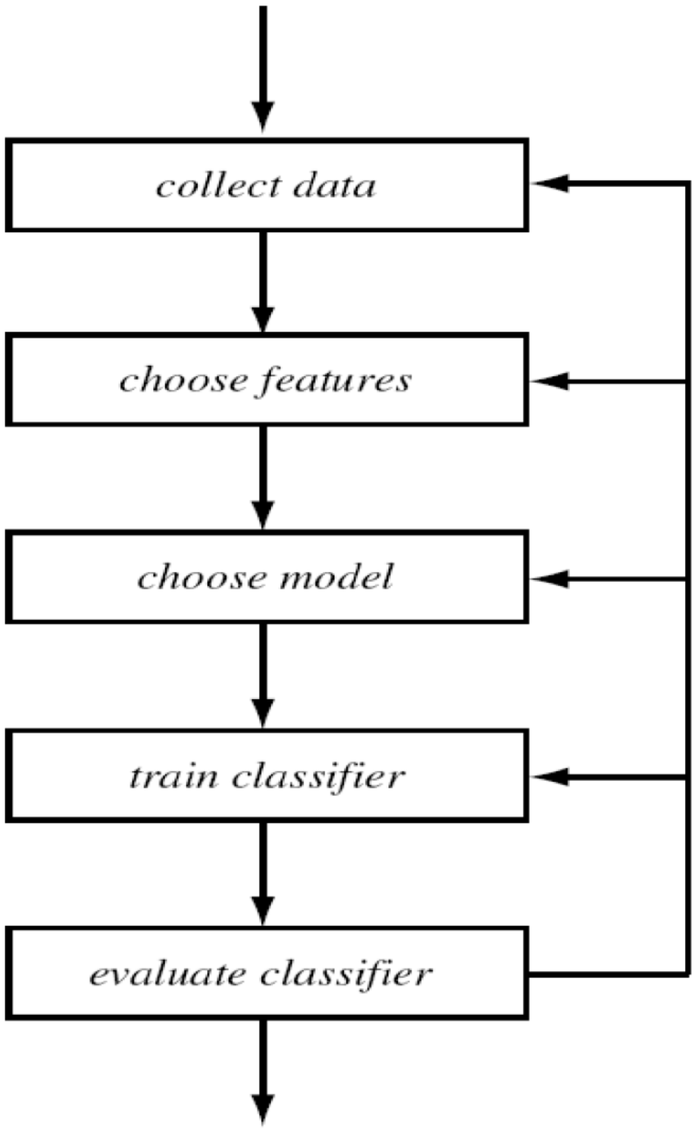
Questions?

Some Background Knowledge in Machine Learning

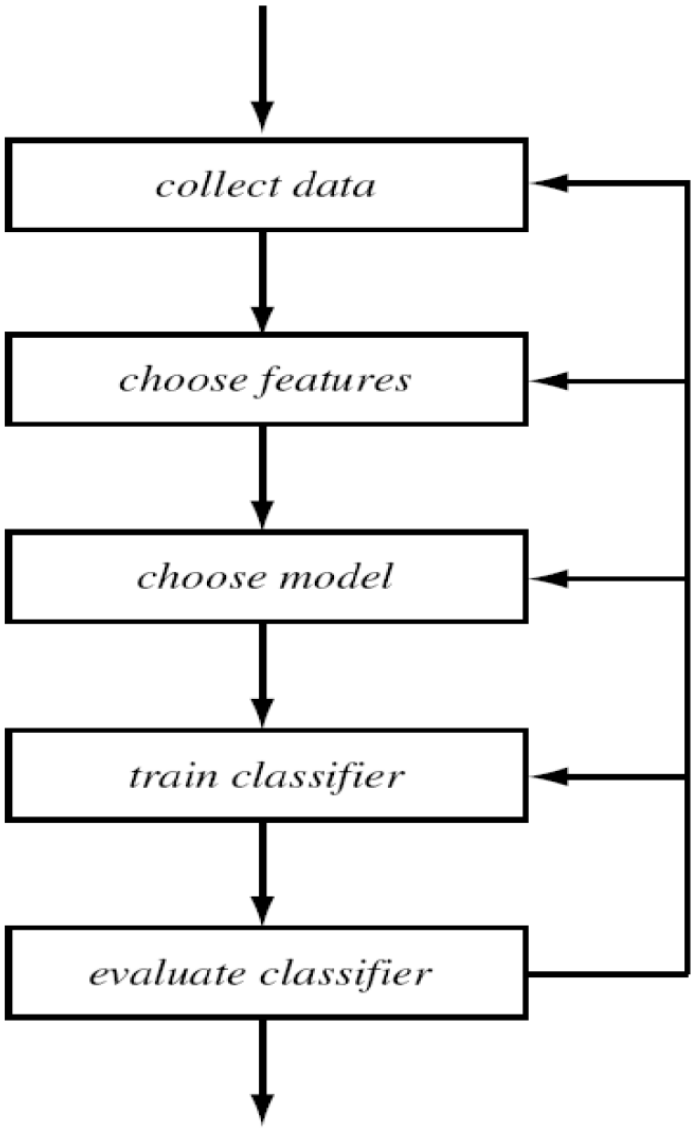
Design Cycle For Supervised Learning

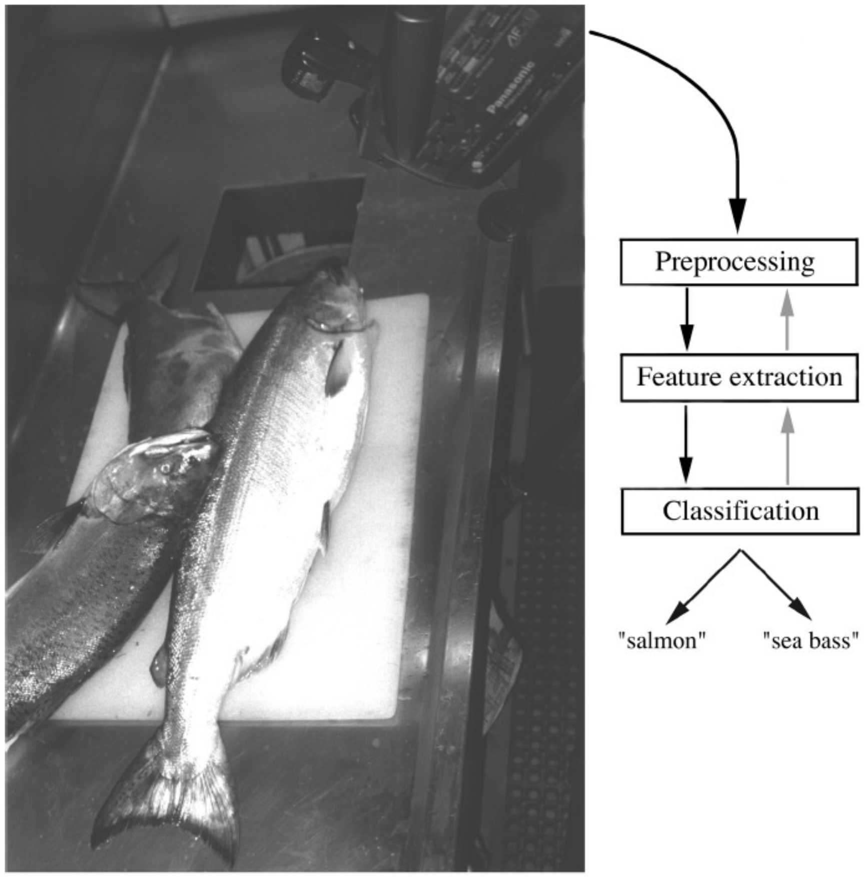


Design Cycle For Supervised Learning



Design Cycle Example: Salmon vs Seabass





Design Cycle Example: Salmon vs Seabass

- Data collection
  - Set up convey belt and cameras, capture images

Design Cycle Example: Salmon vs Seabass

- Data collection
  - Set up convey belt and cameras, capture images
- Preprocessing
  - Image enhancement, remove background, separate occluding fishes, extract single fish (segmentation)

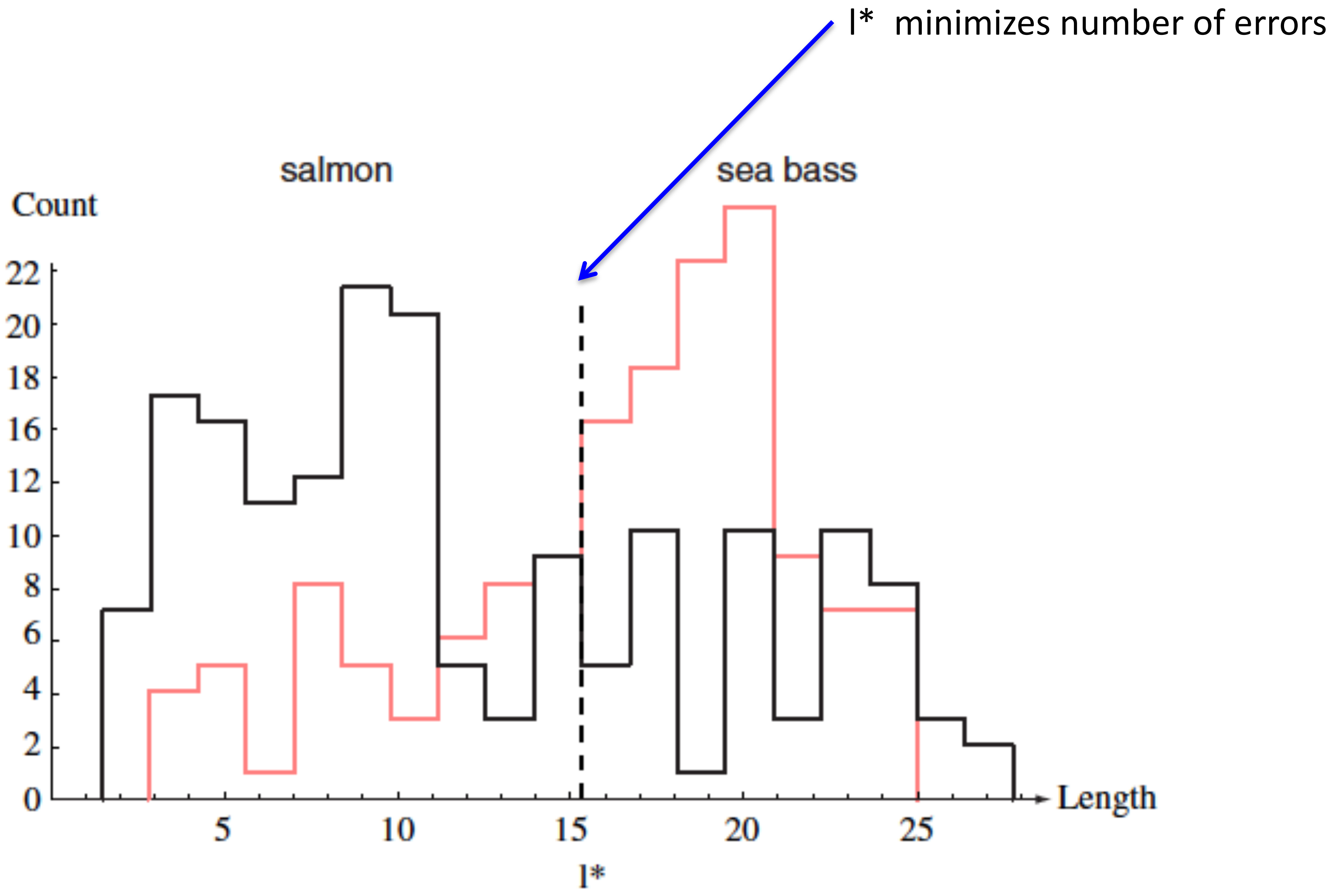
Design Cycle Example: Salmon vs Seabass

- Data collection
  - Set up convey belt and cameras, capture images
- Preprocessing
  - Image enhancement, remove background, separate occluding fishes, extract single fish (segmentation)
- Divide data into training and test sets
  - Feature extraction/engineering, e.g., measure certain features of fish to be used for classifier
  - Train classifier on training set and evaluate on test set

Design Cycle Example: Salmon vs Seabass

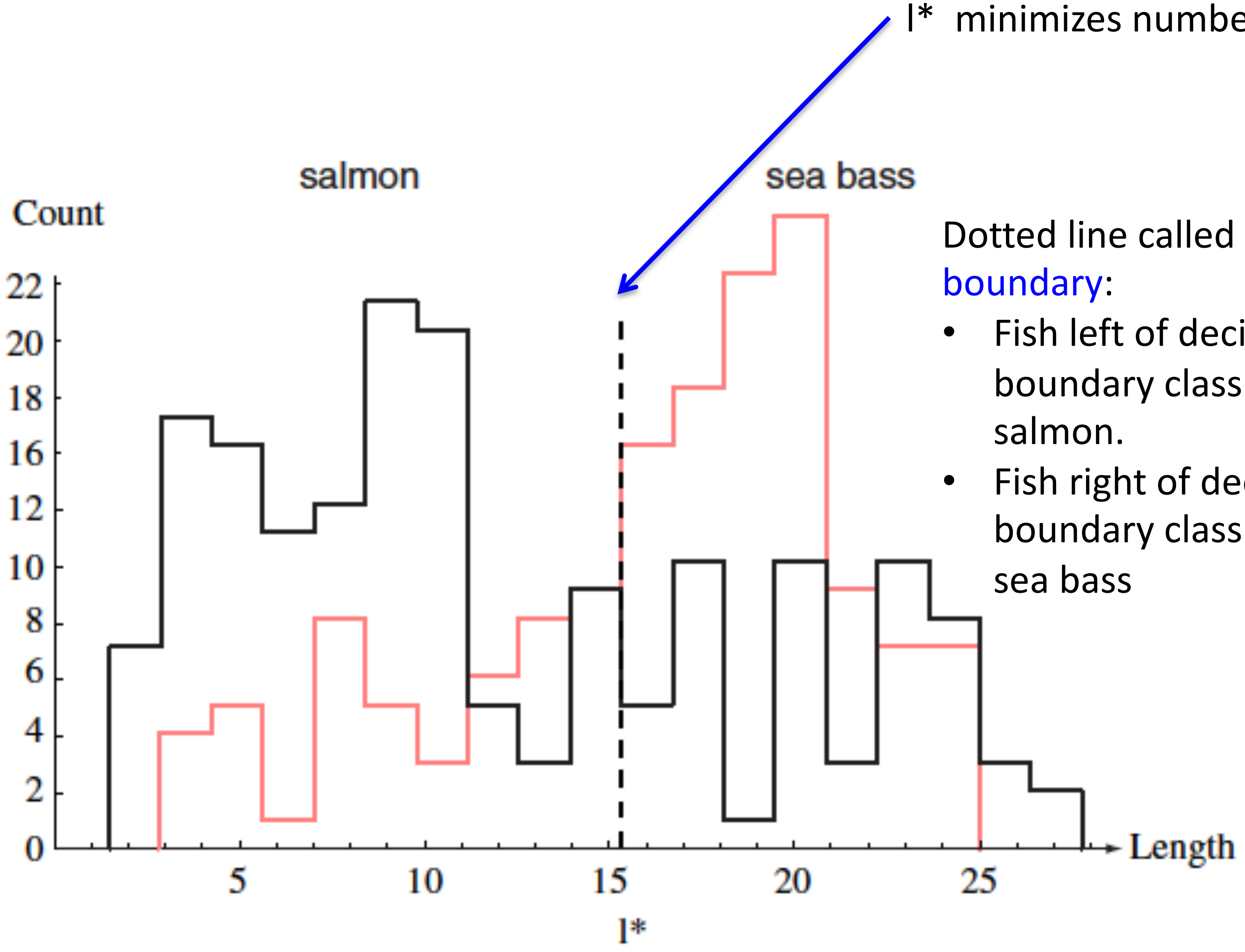
- Data collection
  - Set up convey belt and cameras, capture images
- Preprocessing
  - Image enhancement, remove background, separate occluding fishes, extract single fish (segmentation)
- Divide data into training and test sets
  - Feature extraction/engineering, e.g., measure certain features of fish to be used for classifier
  - Train classifier on training set and evaluate on test set
- Re-visit previous steps if performance unsatisfactory

Feature Engineering is Important!



Feature Engineering is Important!

l* minimizes number of errors

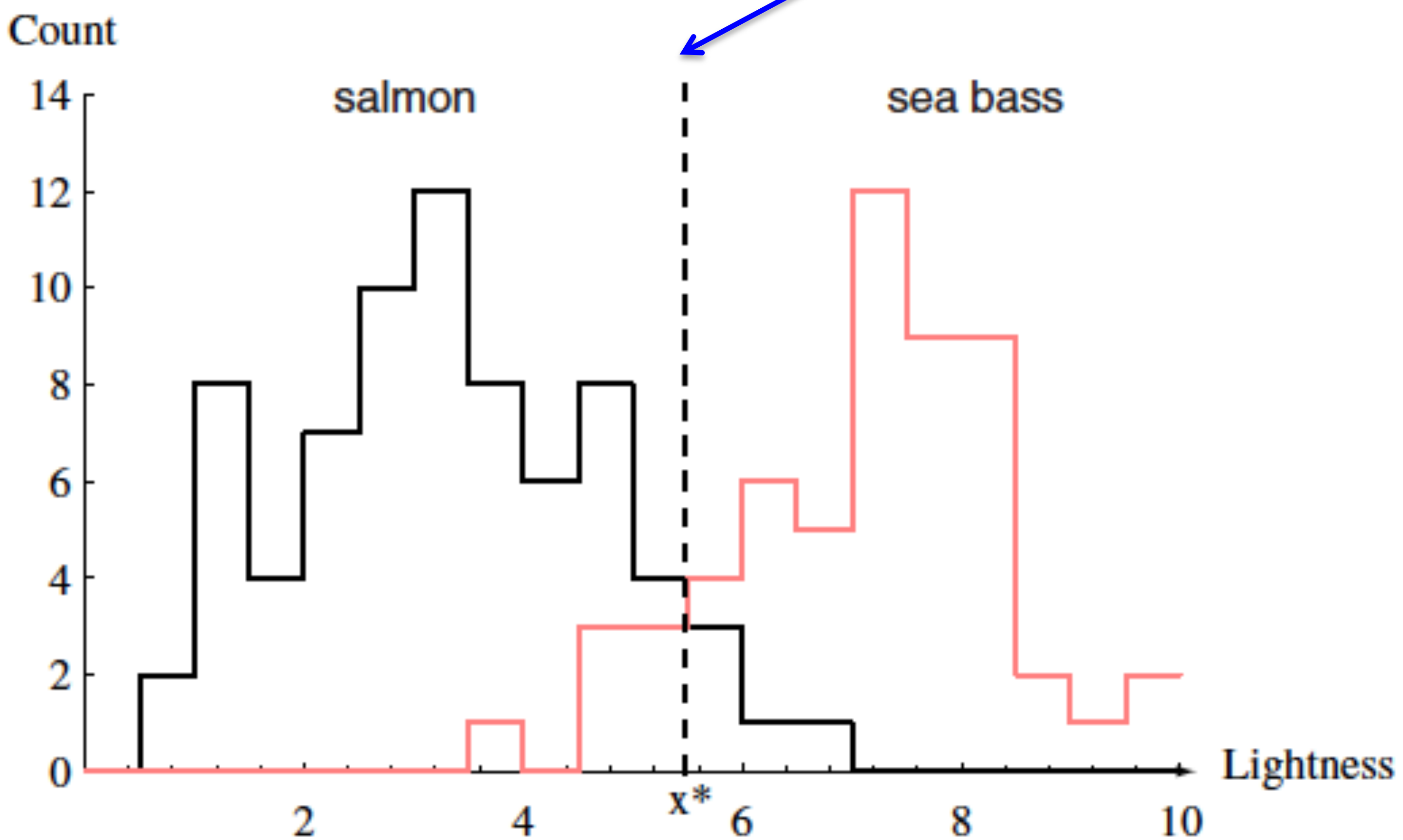


Dotted line called decision boundary:

- Fish left of decision boundary classified as salmon.
- Fish right of decision boundary classified as sea bass

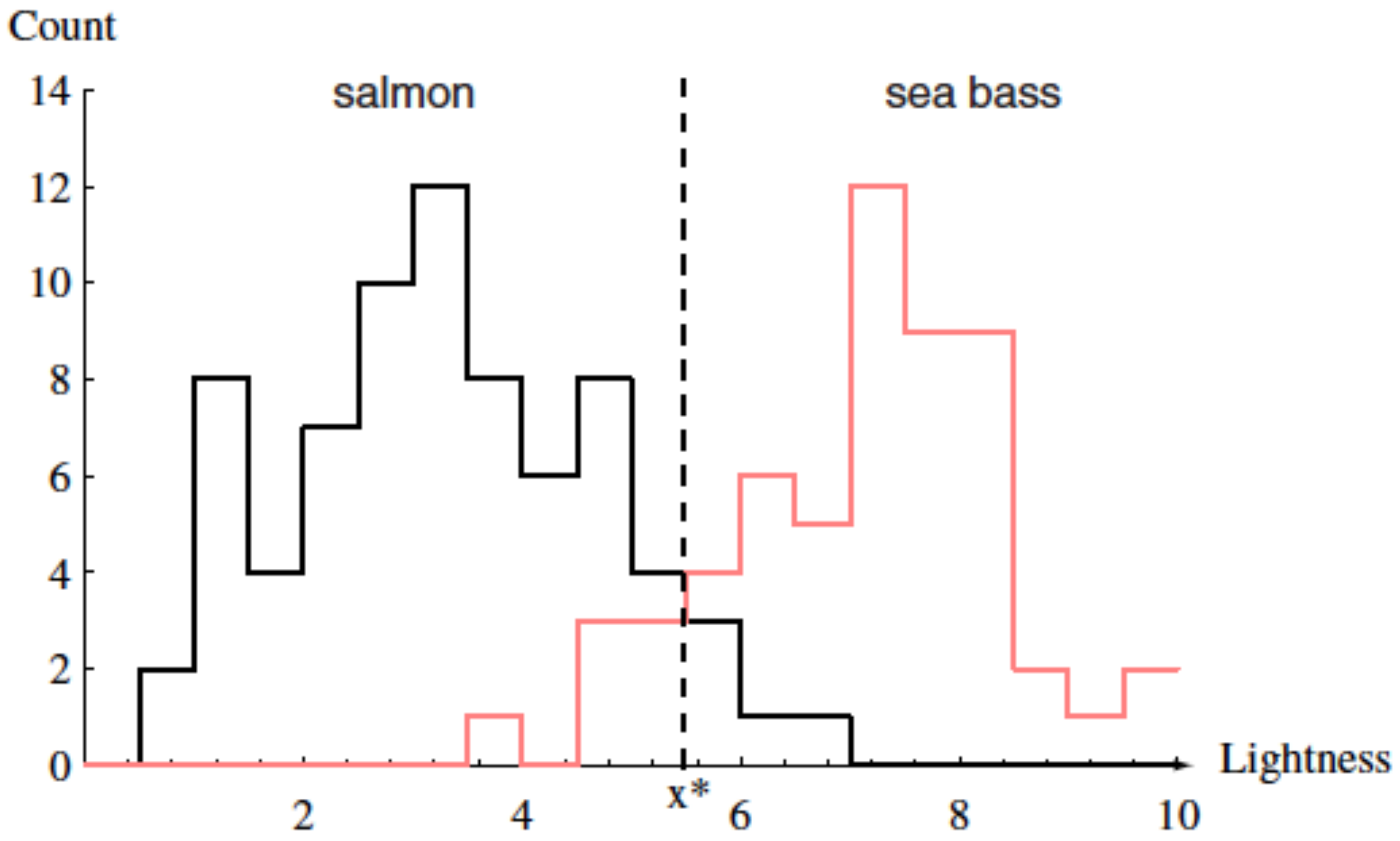
Feature Engineering is Important!

x* minimizes number of errors



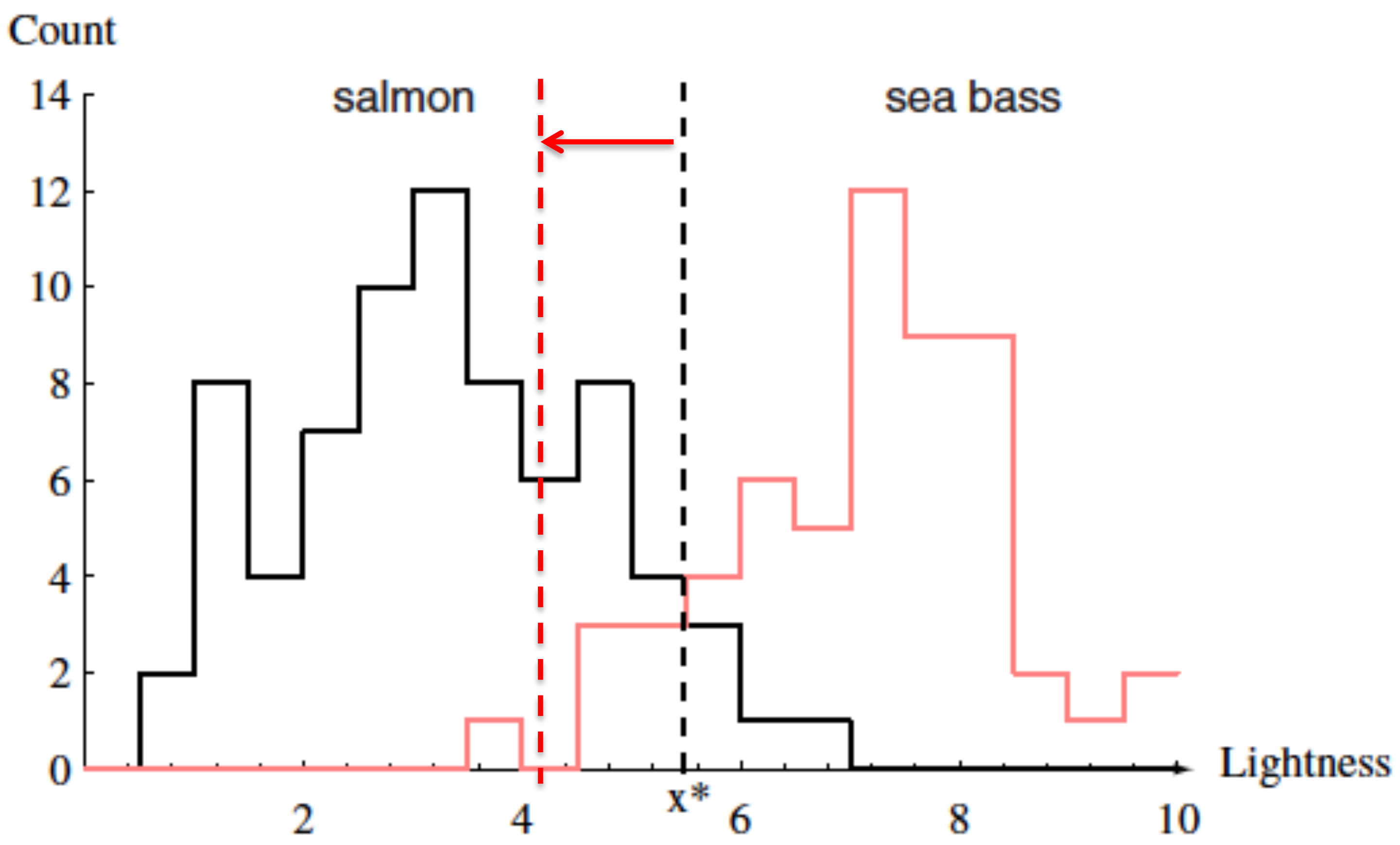
Decision Theory: Not All Errors are Equal

- Example: customers ok with finding salmon in cans marked as sea bass, but very upset to find sea bass in cans marked as salmon
  - Shift decision boundary to minimize angry customers



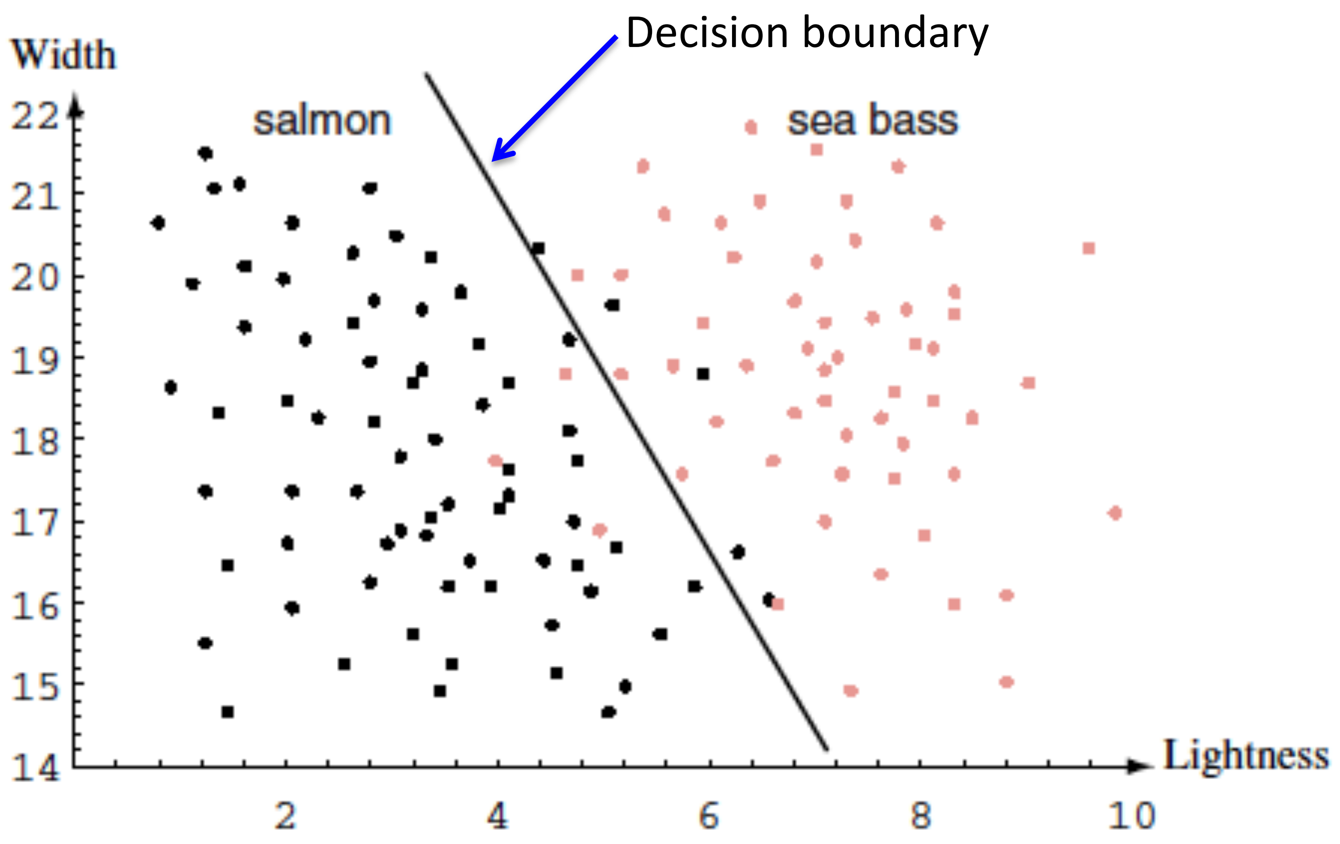
Decision Theory: Not All Errors are Equal

- Example: customers ok with finding salmon in cans marked as sea bass, but very upset to find sea bass in cans marked as salmon
  - Shift decision boundary to minimize angry customers



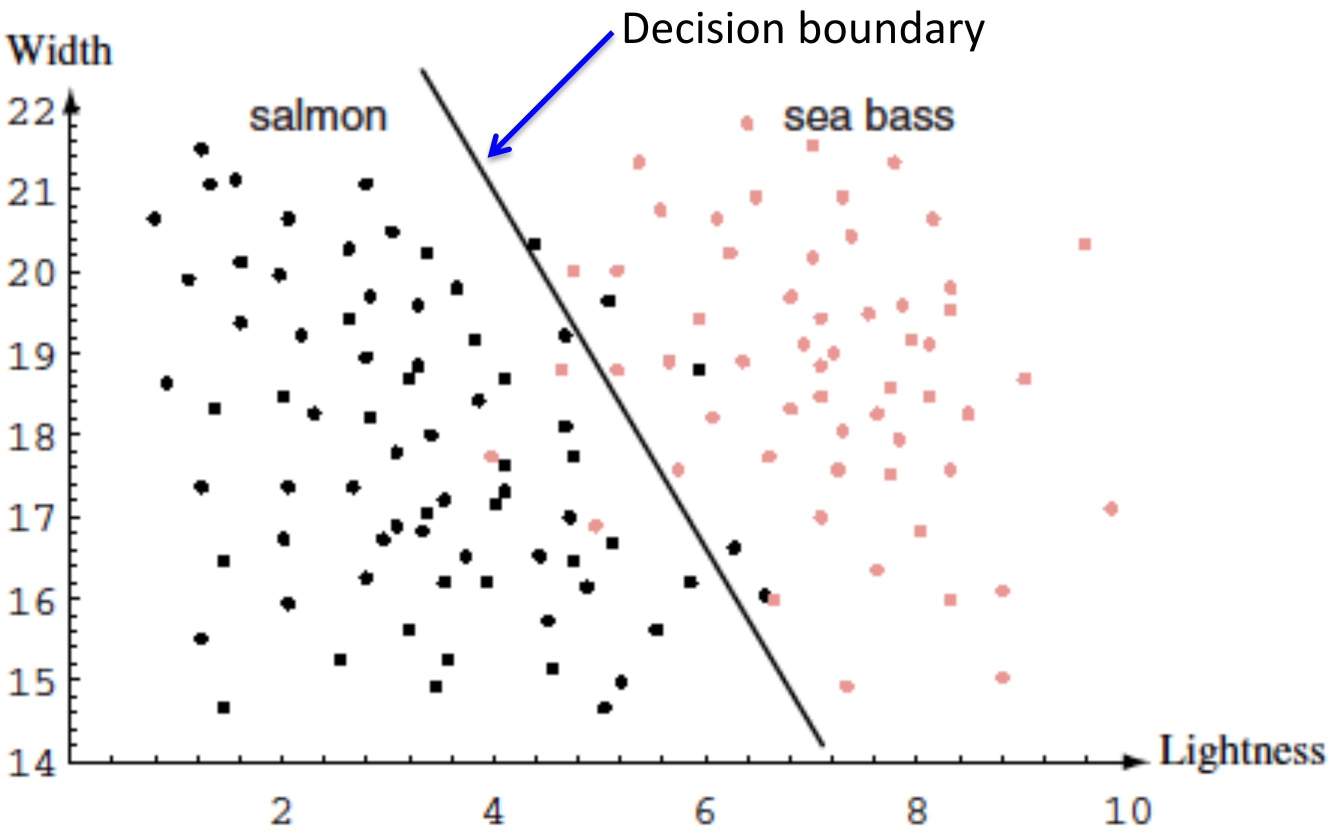
Curse of Dimensionality

- More features might improve performance, but some features (e.g., length) might be useless



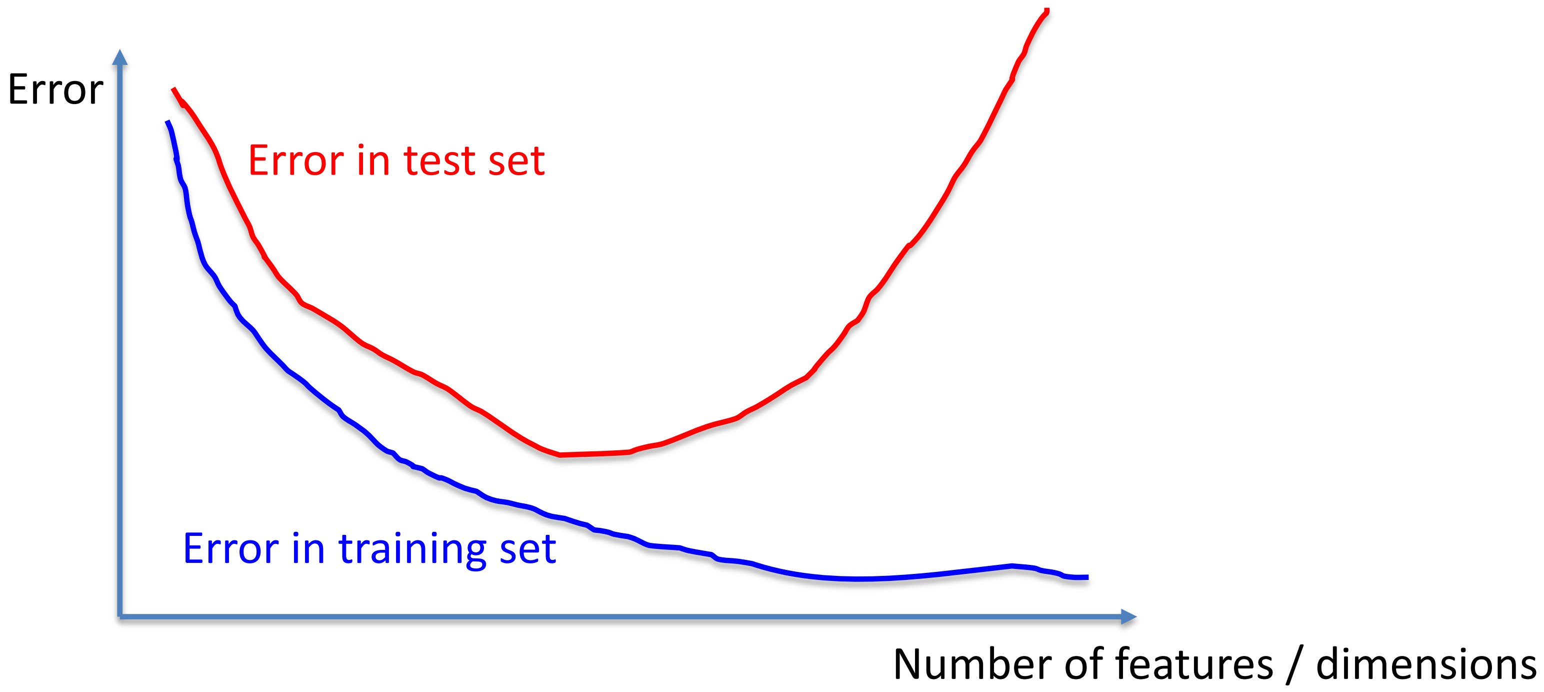
Curse of Dimensionality

- More features might improve performance, but some features (e.g., length) might be useless
- Curse of dimensionality: too many features can lead to worse performance



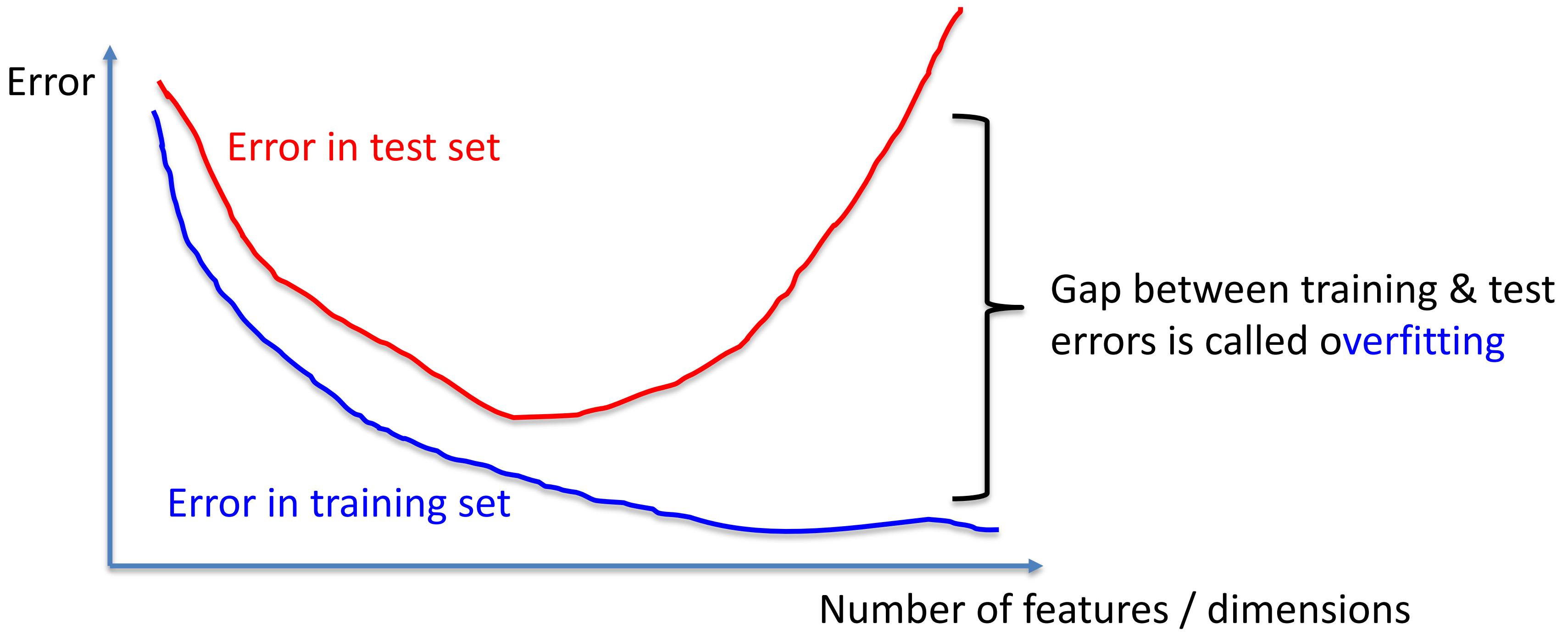
Curse of Dimensionality

- For fixed amount of data samples N (e.g., number of fish photos), as we increase number of features
  - Training error might keep decreasing
  - Test error might decrease and then increase



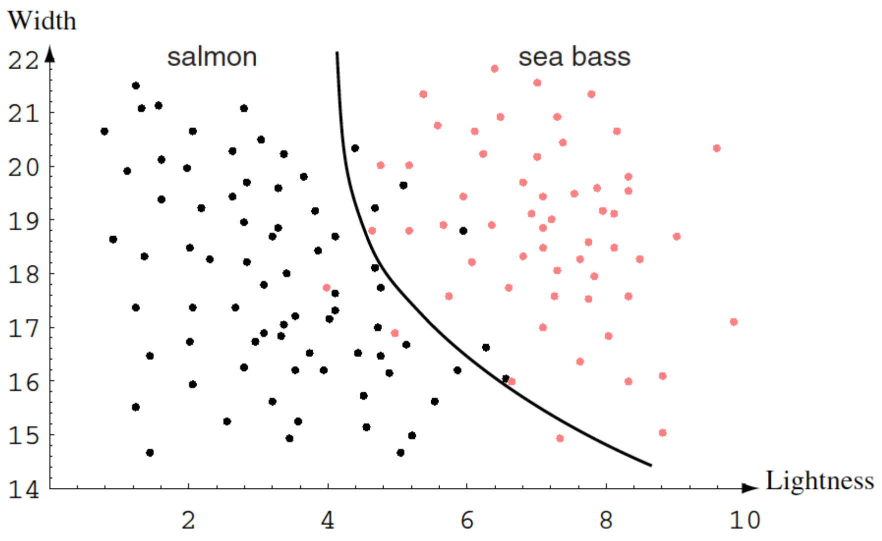
Curse of Dimensionality

- For fixed amount of data samples N (e.g., number of fish photos), as we increase number of features
  - Training error might keep decreasing
  - Test error might decrease and then increase



Model Complexity

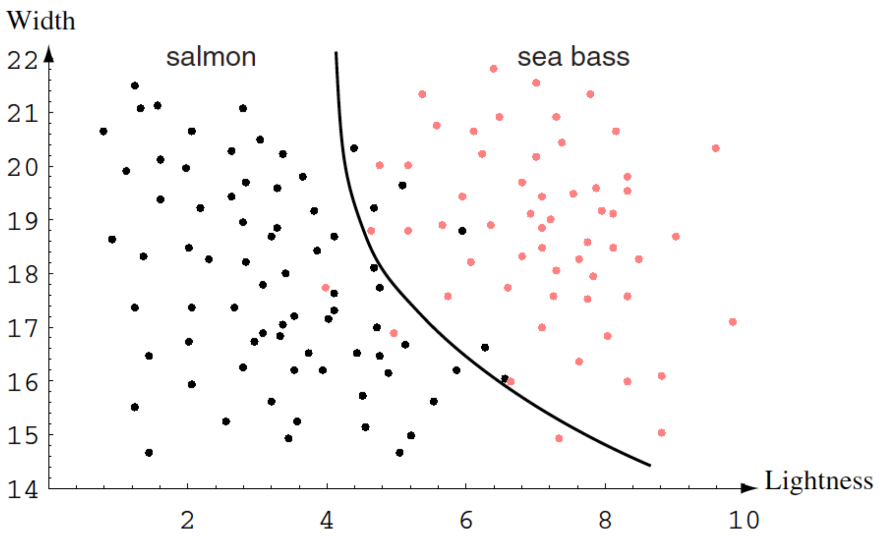
- Instead of increasing the number of features, can also use more complex decision boundaries (models)
- This can potentially reduce errors, but might also increase errors



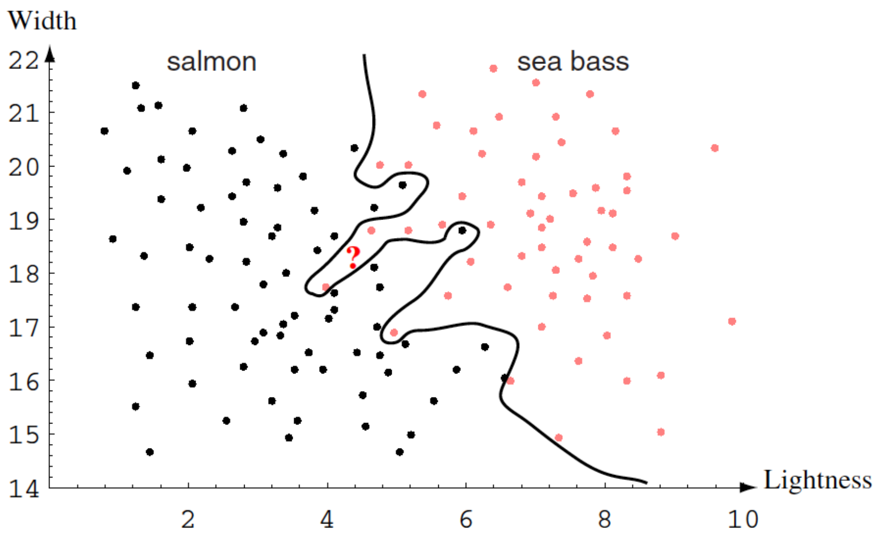
Shifting from linear straight line to quadratic curve reduces errors (in this example)

Model Complexity

- Instead of increasing the number of features, can also use more complex decision boundaries (models)
- This can potentially reduce errors, but might also increase errors



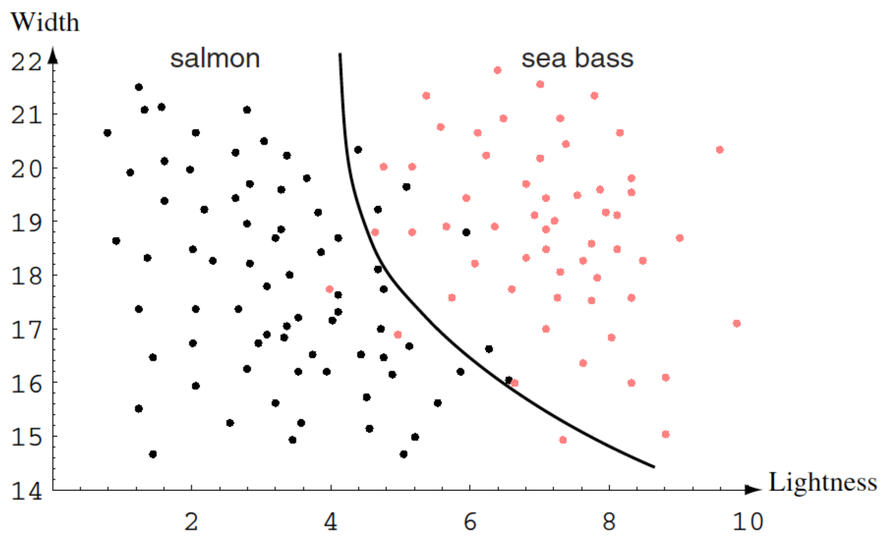
Shifting from linear straight line to quadratic curve reduces errors (in this example)



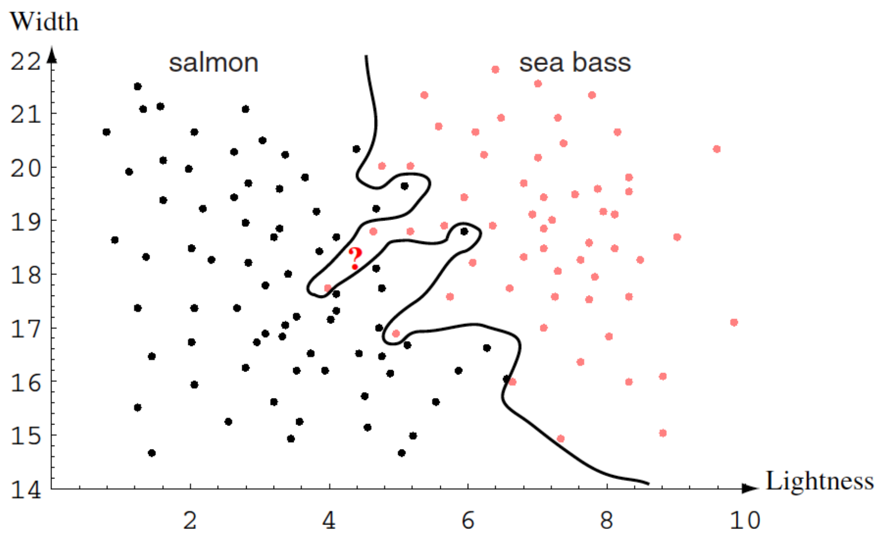
"?" likely to be salmon, but classified as sea bass

Model Complexity

- Instead of increasing the number of features, can also use more complex decision boundaries (models)
- This can potentially reduce errors, but might also increase errors
- Model complexity roughly related to # model parameters (e.g., quadratic decision boundary requires more parameters to specify than linear decision boundary)



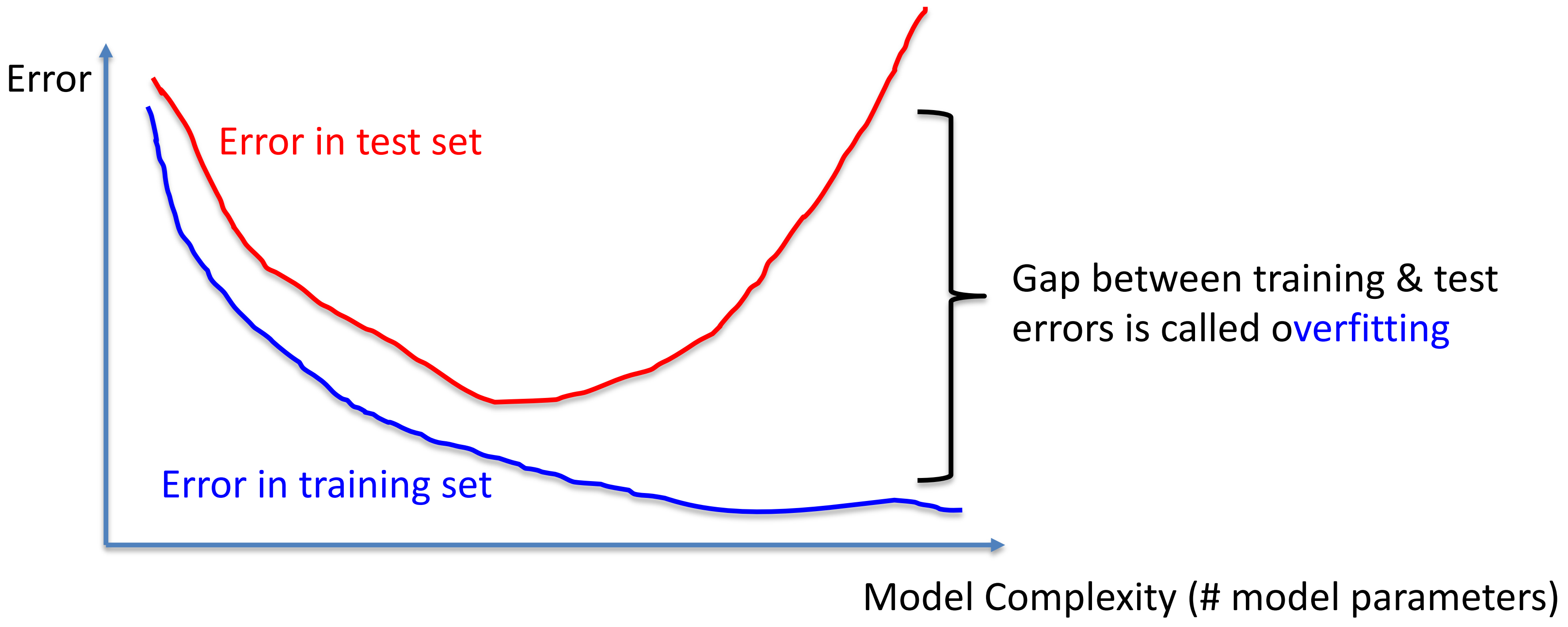
Shifting from linear straight line to quadratic curve reduces errors (in this example)



"?" likely to be salmon, but classified as sea bass

Model Complexity

- For fixed amount of data samples N (e.g., number of fish photos), as we increase number of model parameters
  - Training error might keep decreasing
  - Test error might decrease and then increase



Training, Validation, Test Sets

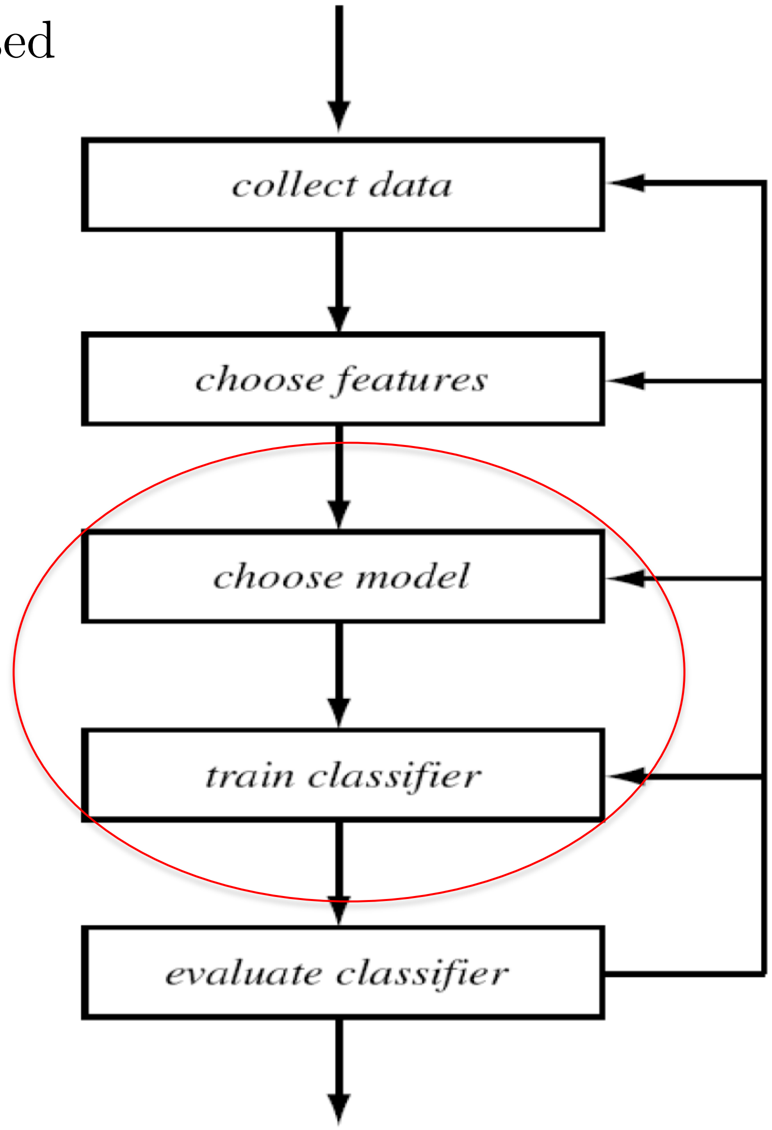
- To really test the quality of our algorithm, need to evaluate generalization error on test set
- However, splitting data into training-test set not enough
  - Imagine we train an algorithm on training set and error is terrible in test set
  - We then train new algorithm on training set and error is now better in test set
  - But we will have used the test set twice. If you repeat this many times, you will overfit to the test set

Training, Validation, Test Sets

- To really test the quality of our algorithm, need to evaluate generalization error on test set
- However, splitting data into training-test set not enough
  - Imagine we train an algorithm on training set and error is terrible in test set
  - We then train new algorithm on training set and error is now better in test set
  - But we will have used the test set twice. If you repeat this many times, you will overfit to the test set
- Best practice: split data into training, validation and test sets
  - Train on training set & evaluate error on validation set
  - Repeat as many times as we like
  - Once satisfied with results, then apply to test set to get realistic error quantification

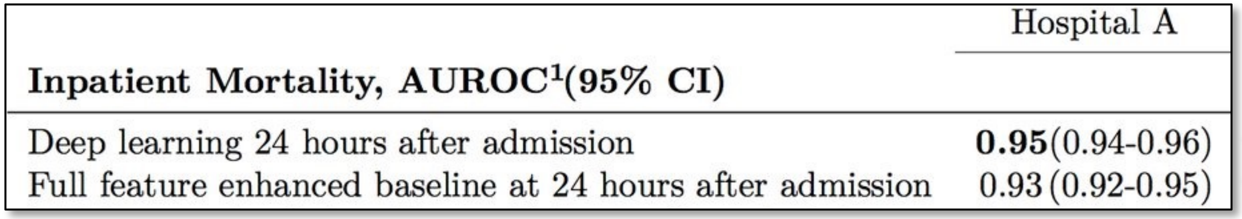
What will be covered?

- Focus on classification/regression/unsupervised learning
  - Assume features already extracted
  - Strong bias on probabilistic approaches



Why Not Just Teach Deep Learning?

- In certain settings, deep learning do not beat classical machine learning, e.g., logistic regression
  - These classical approaches can be significantly faster and easier to implement



	Hospital A
Inpatient Mortality, AUROC[1](95% CI)	
Deep learning 24 hours after admission	**0.95**(0.94-0.96)
Full feature enhanced baseline at 24 hours after admission	0.93 (0.92-0.95)

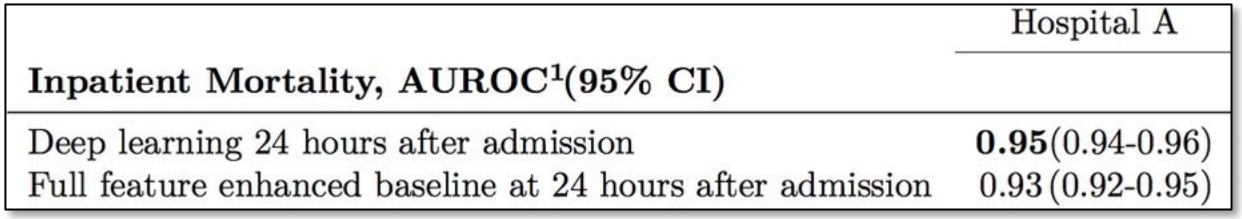
Deep learning

Logistic Regression

Google Research, Scalable and accurate deep learning with electronic health records, NPJ Digital Medicine, 2018

Why Not Just Teach Deep Learning?

- In certain settings, deep learning do not beat classical machine learning, e.g., logistic regression
  - These classical approaches can be significantly faster and easier to implement



Inpatient Mortality, AUROC[1](95% CI)	Hospital A
Deep learning 24 hours after admission	**0.95**(0.94-0.96)
Full feature enhanced baseline at 24 hours after admission	0.93(0.92-0.95)

Deep learning

Logistic Regression

Google Research, Scalable and accurate deep learning with electronic health records, NPJ Digital Medicine, 2018

- Machine learning is cyclical
  - Our goal not to teach you only the popular stuff, but foundational knowledge useful regardless of what is popular because that will change

Summary

- Different types of learning: supervised (classification vs regression), unsupervised, semi-supervised, reinforcement learning

Summary

- Different types of learning: supervised (classification vs regression), unsupervised, semi-supervised, reinforcement learning
- Iterative nature: collect data, choose features, choose models, train, test, rinse and repeat

Summary

- Different types of learning: supervised (classification vs regression), unsupervised, semi-supervised, reinforcement learning
- Iterative nature: collect data, choose features, choose models, train, test, rinse and repeat
- Feature engineering (i.e., representation) is important

Summary

- Different types of learning: supervised (classification vs regression), unsupervised, semi-supervised, reinforcement learning
- Iterative nature: collect data, choose features, choose models, train, test, rinse and repeat
- Feature engineering (i.e., representation) is important
- Decision theory: not all errors are equal

Summary

- Different types of learning: supervised (classification vs regression), unsupervised, semi-supervised, reinforcement learning
- Iterative nature: collect data, choose features, choose models, train, test, rinse and repeat
- Feature engineering (i.e., representation) is important
- Decision theory: not all errors are equal
- Curse of dimensionality, model complexity, overfitting

Summary

- Different types of learning: supervised (classification vs regression), unsupervised, semi-supervised, reinforcement learning
- Iterative nature: collect data, choose features, choose models, train, test, rinse and repeat
- Feature engineering (i.e., representation) is important
- Decision theory: not all errors are equal
- Curse of dimensionality, model complexity, overfitting
- Training-validation-testing

Questions?