Machine Learning: More Review

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-736

1/19/2023

Course Page/Syllabus Up

- Syllabus and policy:
  - https://www.cs.rit.edu/~ago/courses/736/index.html
- Prerequisites:
  - Choose your teams by end of tomorrow (Friday, 1/20/2023, 11:59pm) or else you will be randomly assigned your team

Performance

- There are several factors affecting the performance:
  - **Types of training** provided
  - The form and extent of any initial **background knowledge**
  - The **type of feedback** provided
  - The **learning algorithms** used
- Two important factors:
  - Modeling
  - Optimization
- The success of machine learning system also depends on the algorithms.
- The algorithms control the search to find and build the knowledge structures.
- The learning algorithms should extract useful information from training examples.

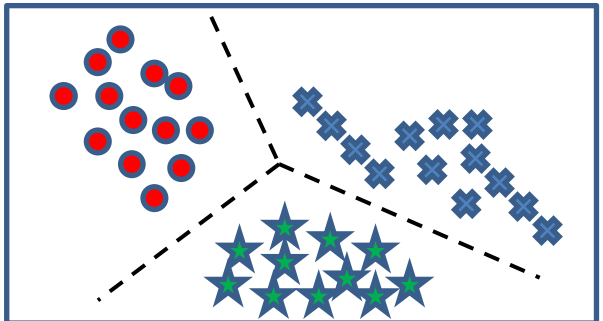
Types of Learning

- **Supervised (inductive) learning** $\{x_n \in R^d, y_n \in R\}_{n=1}^N$
  - Training data includes desired outputs
  - Prediction / Classification (discrete labels), Regression (real values)
- **Unsupervised learning** $\{x_n \in R^d\}_{n=1}^N$
  - Training data does not include desired outputs
  - Clustering / probability distribution estimation
  - Finding association (in features)
  - Dimension reduction
- **Semi-supervised learning**
  - Training data includes a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions
  - Decision making (robot, chess machine)

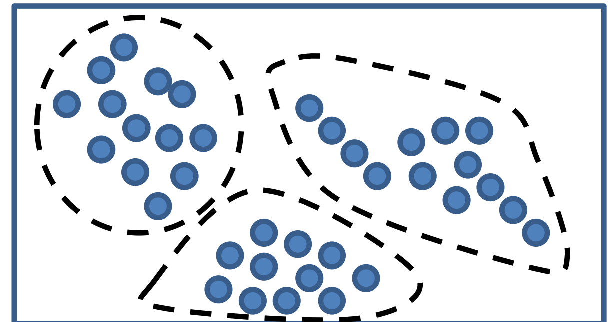
Inductive Learning

- **Given** examples of a function *(X, F(X))*
- **Predict** function *F(X)* for new examples *X*
  - Discrete *F(X)*: Classification
  - Continuous *F(X)*: Regression
  - *F(X)* = Probability(*X*): Probability estimation

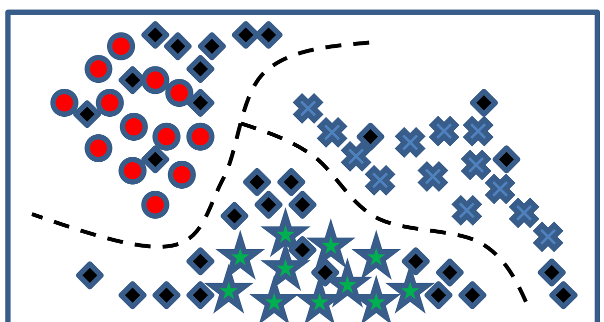
Visualizing Types of Learning



Supervised learning



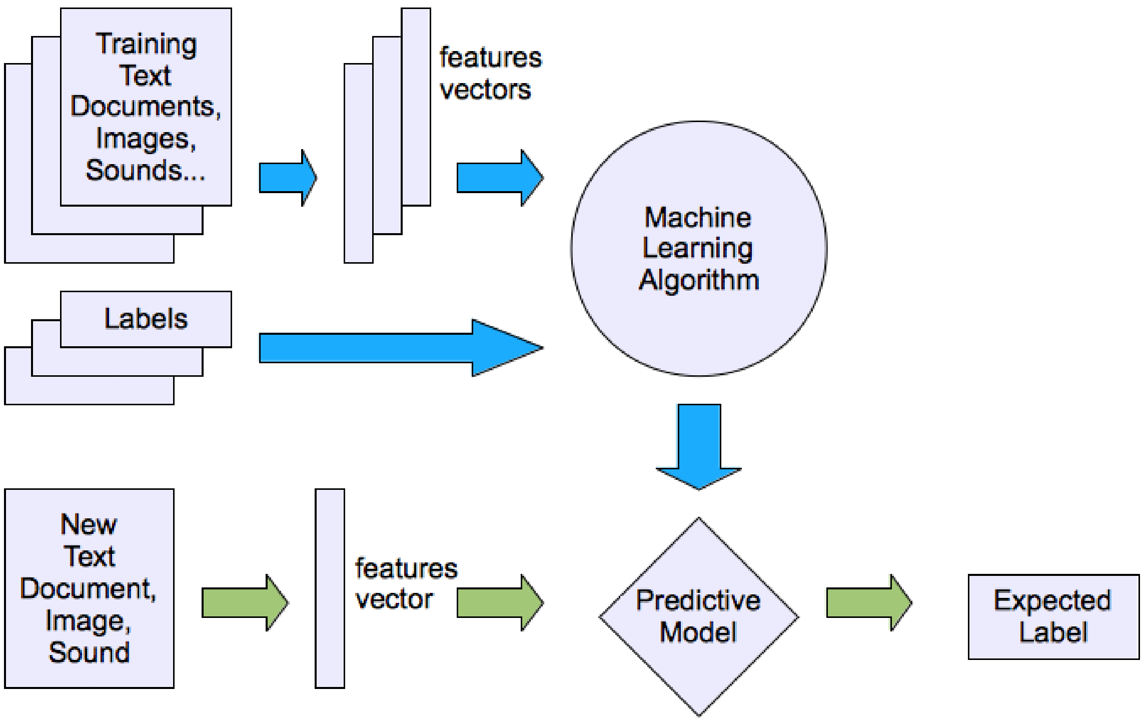
Unsupervised learning



Semi-supervised learning

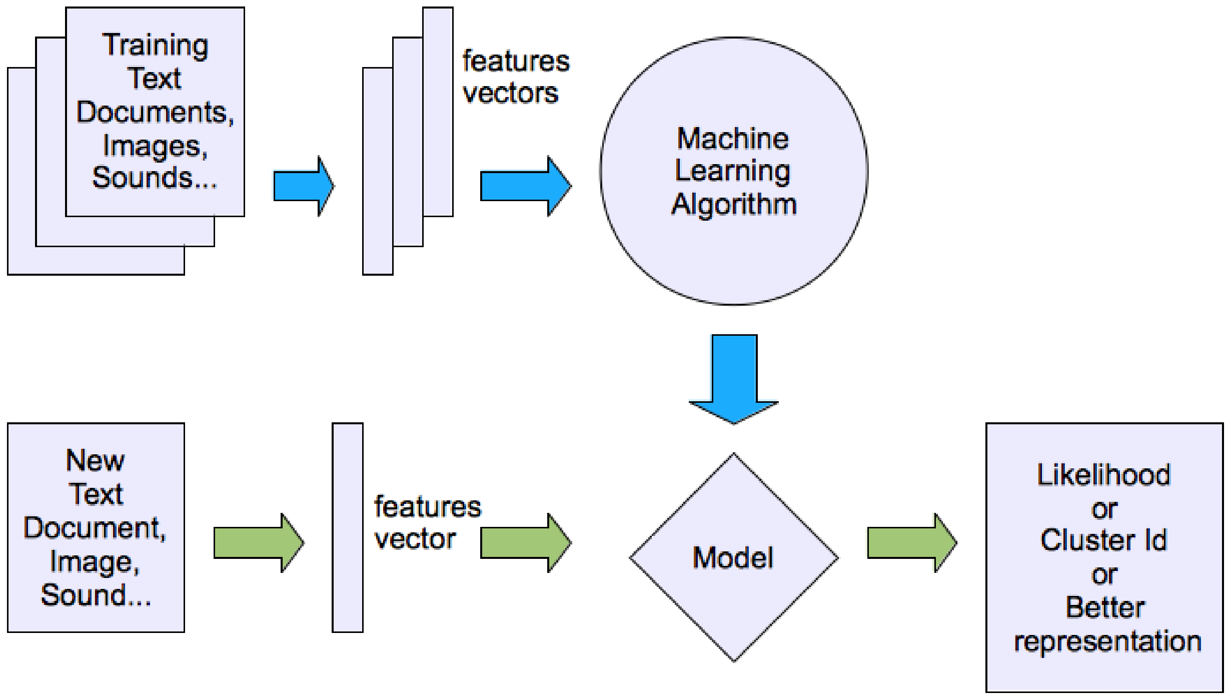
A Typical Supervised Learning Pipeline

- Supervised learning



A Typical Unsupervised Learning Pipeline

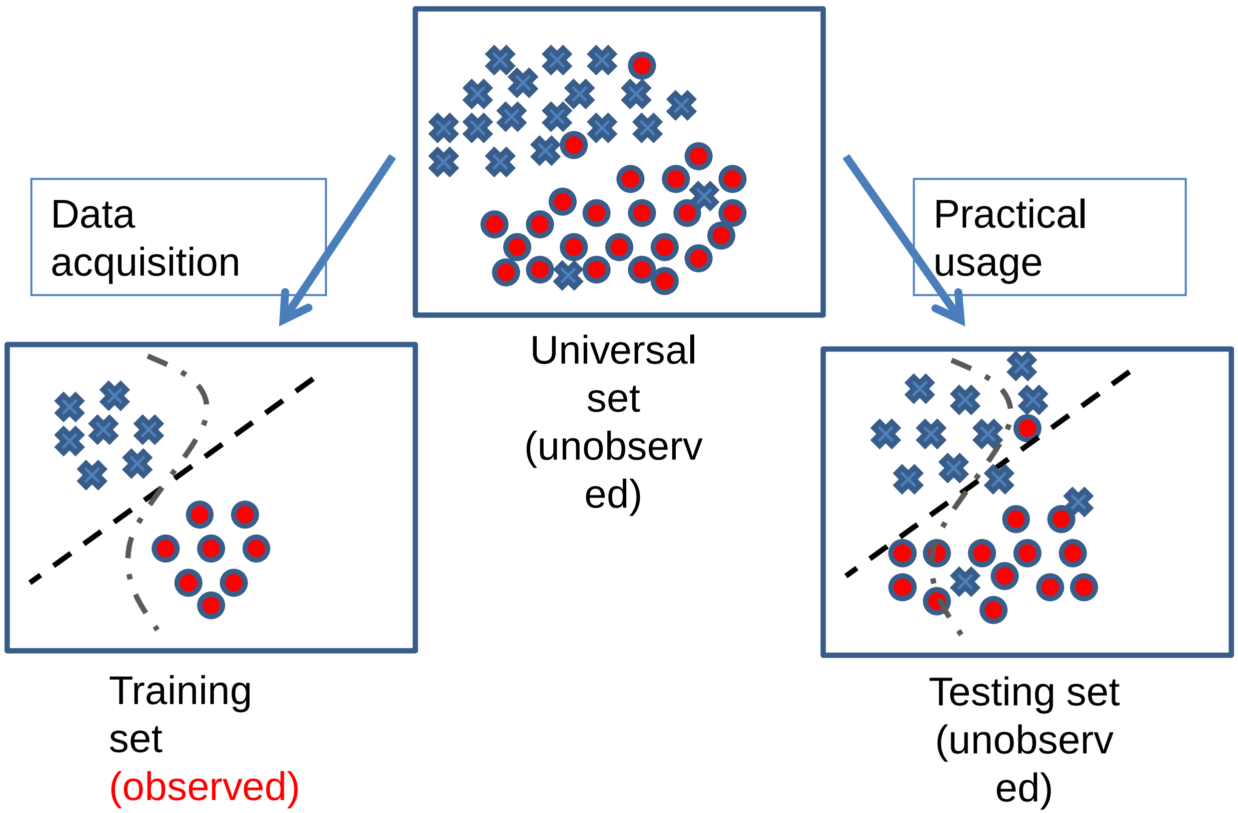
- Unsupervised learning



ML in Practice

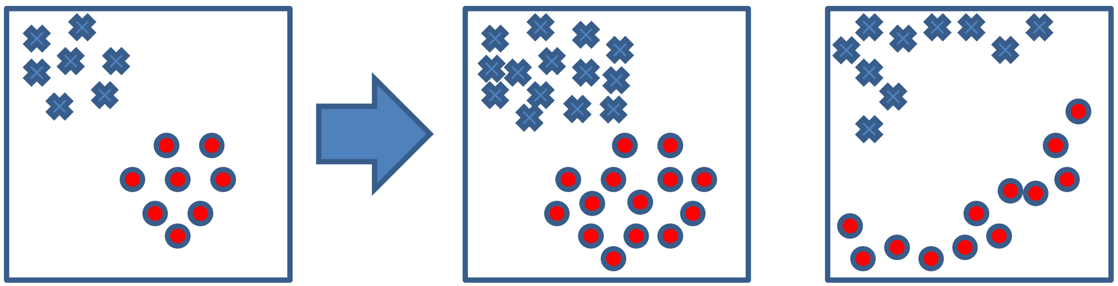
- Understanding domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learning models
- Interpreting results
- Consolidating and deploying discovered knowledge
- Loop

First, let there be data!



Training and testing

- Training = the process of making the system able to learn
- Testing = the process of seeing how well the system learned
  - Simulates "real world" usage
  - Training set and testing set come from the same distribution
  - Need to make some assumptions or bias
- Deployment = actually using the learned system in practice



Applications Abound

- Face detection
- Object detection and recognition
- Image segmentation
- Multimedia event detection
- Web search / information retrieval
- Computational biology
- Finance
- E-commerce
- Space exploration
- Robotics
- Information extraction
- Social networks
- Debugging
- *[Your favorite area]*

Machine Learning Problems

	Supervised Learning	***Unsupervised Learning***
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Clustering Strategies

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Machine Learning Problems

	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

The machine learning framework

- Apply a prediction function to a feature representation of the image to get the desired output:

f() = “apple”

f() = “tomato”

f() = “cow”

Slide credit: L. Lazebnik

The machine learning framework

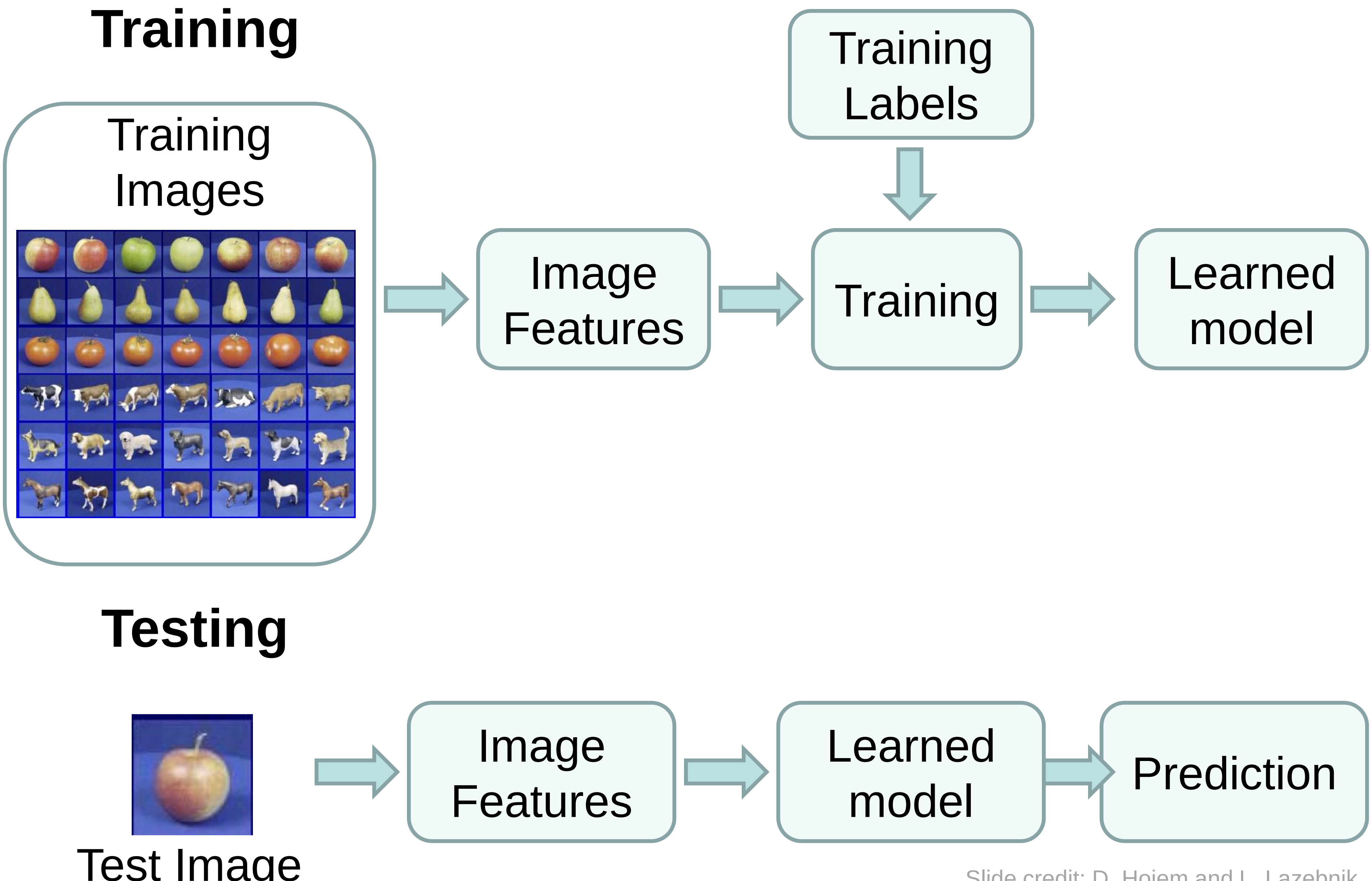
$$\mathrm{y} = \mathrm{f}(\mathbf{x})$$

output   prediction function   Image feature

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply f to a never before seen *test example* **x** and output the predicted value $y = f(\mathbf{x})$

Slide credit: L. Lazebnik

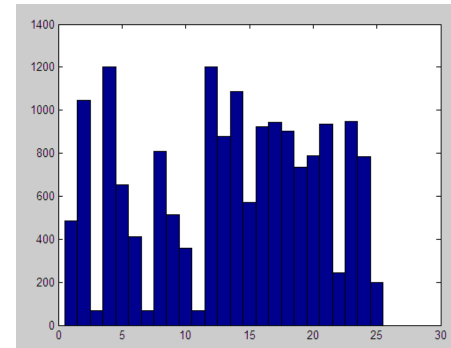
Steps



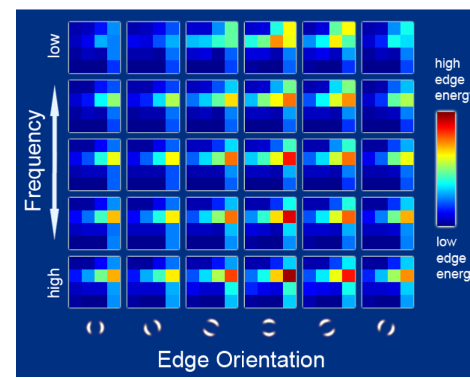
Slide credit: D. Hoiem and L. Lazebnik

Features

- Raw pixels
- Histograms



- GIST descriptors



- ...

Slide credit: L. Lazebnik

Many classifiers to choose from

- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- RBMs
- Etc.

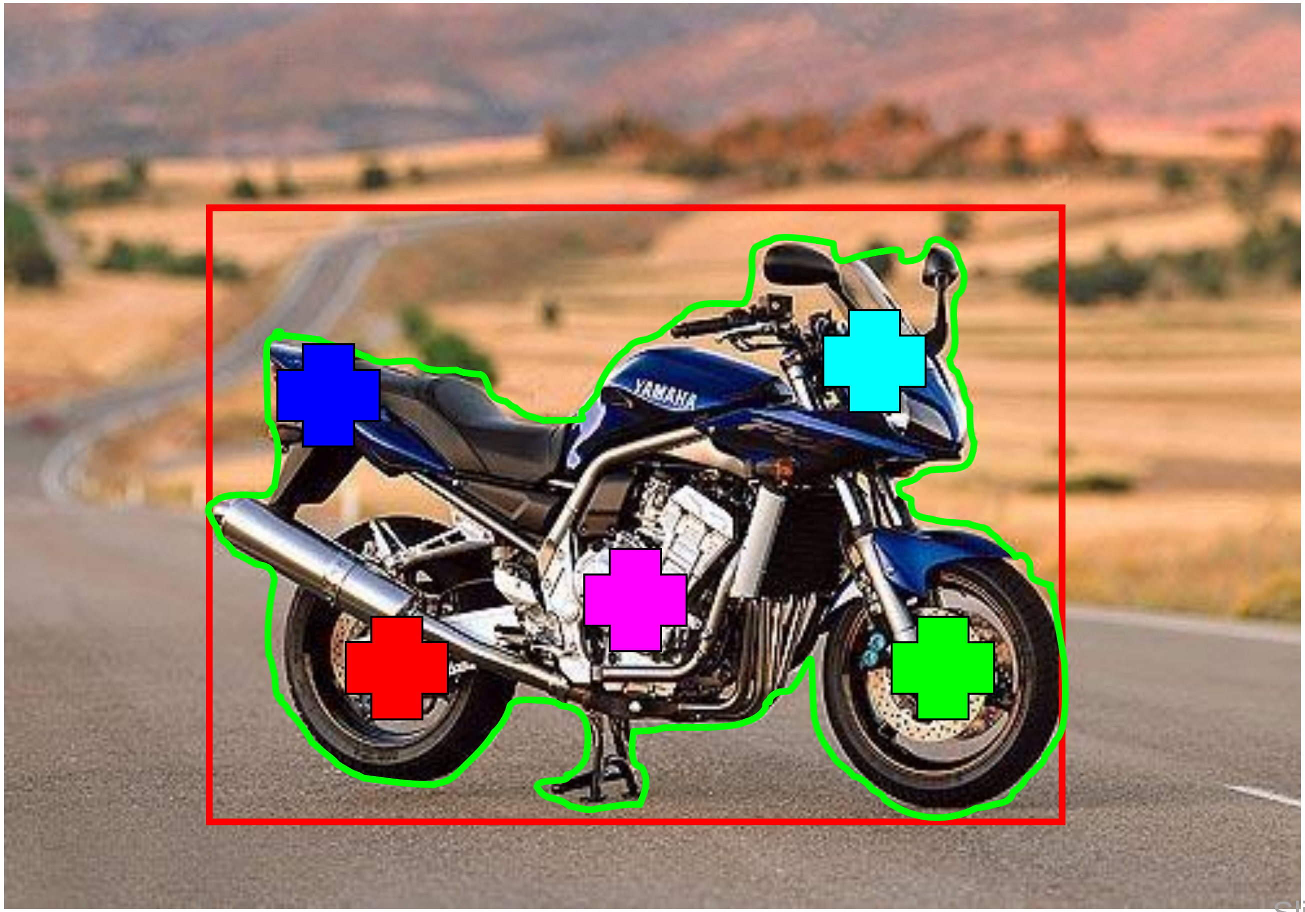
Which is the best one?

Slide credit: D. Hoiem

Recognition task and supervision

- Images in the training set must be annotated with the "correct answer" that the model is expected to produce

Contains a motorbike



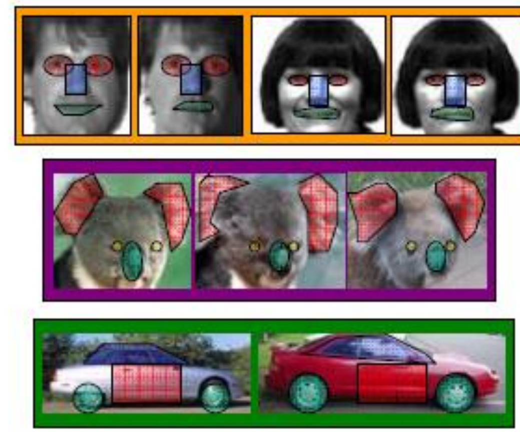
Slide credit: L. Lazebnik

Spectrum of supervision

Less **More**

Unsupervised

"Weakly" supervised



Fully supervised

Definition depends on task

Slide credit: L. Lazebnik

Generalization

Training set (labels known)

Test set (labels unknown)

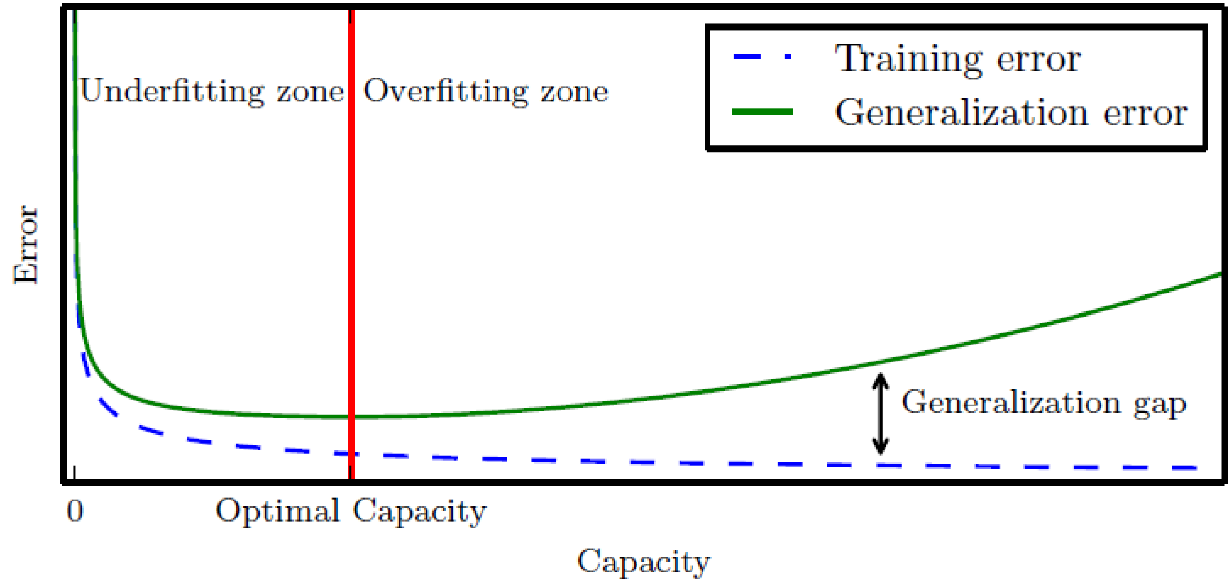
- How well does a learned model generalize from the data it was trained on to a new test set?

Slide credit: L. Lazebnik

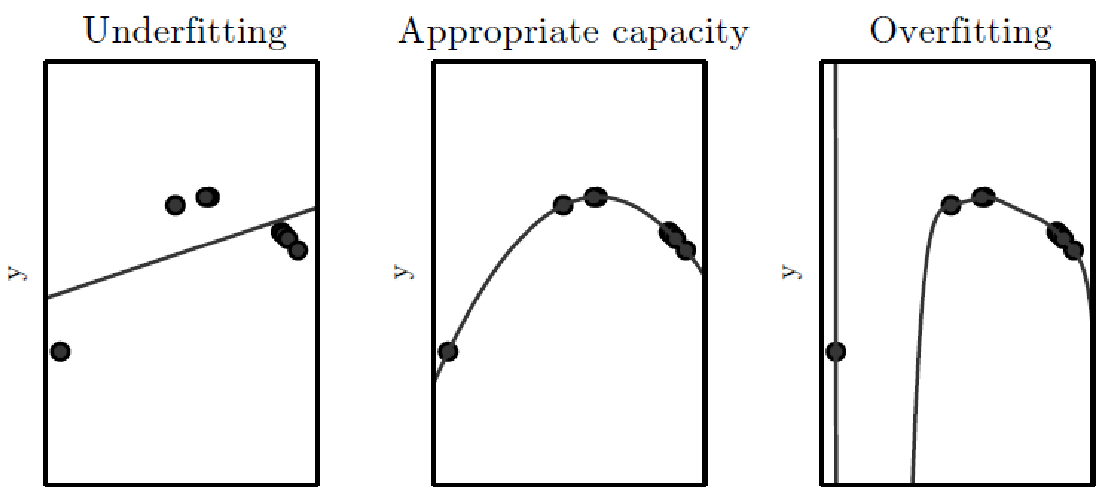
Generalization

- Components of generalization error
  - **Bias:** how much the average model over all training sets differ from the true model?
    - Error due to inaccurate assumptions/simplifications made by the model
  - **Variance:** how much models estimated from different training sets differ from each other
- **Underfitting:** model is too "simple" to represent all the relevant class characteristics
  - High bias and low variance
  - High training error and high test error
- **Overfitting:** model is too "complex" and fits irrelevant characteristics (noise) in the data
  - Low bias and high variance
  - Low training error and high test error

Generalization and Capacity



Underfitting and Overfitting in Polynomial Estimation

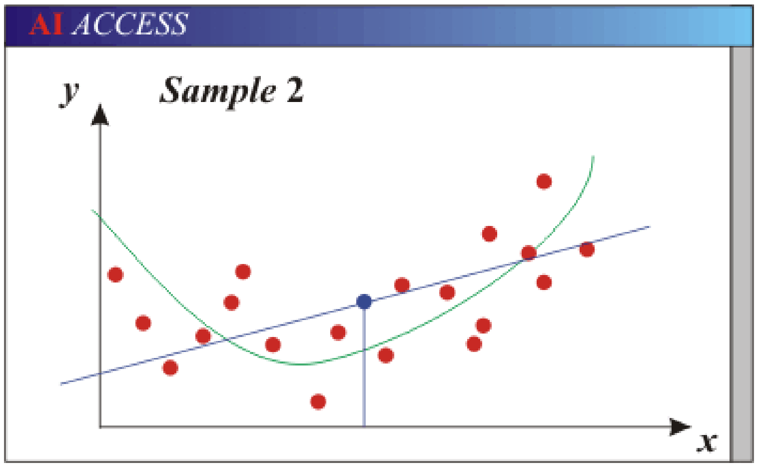


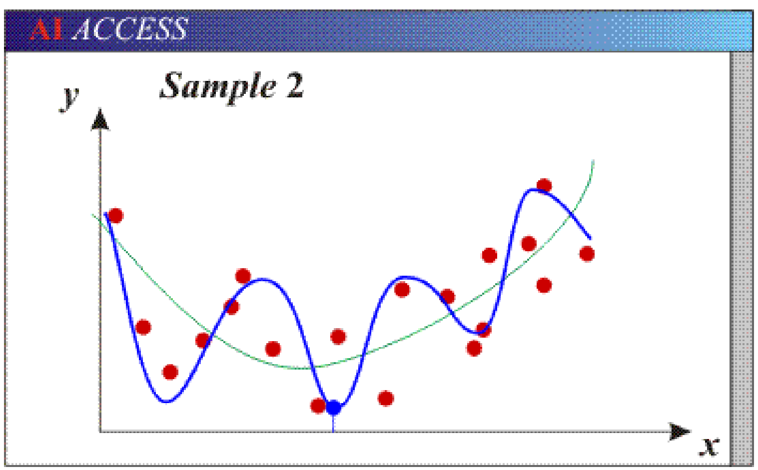
No Free Lunch Theorem

You can only get generalization through assumptions. No one algorithm will solve all problems (some will work better than others in some instances).

Slide credit: D. Hoiem

Bias-Variance Trade-off





- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).
- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

Slide credit: D. Hoiem

Bias-Variance Trade-off

$$E(MSE) = noise^2 + bias^2 + variance$$

Unavoidable error

Error due to incorrect assumptions

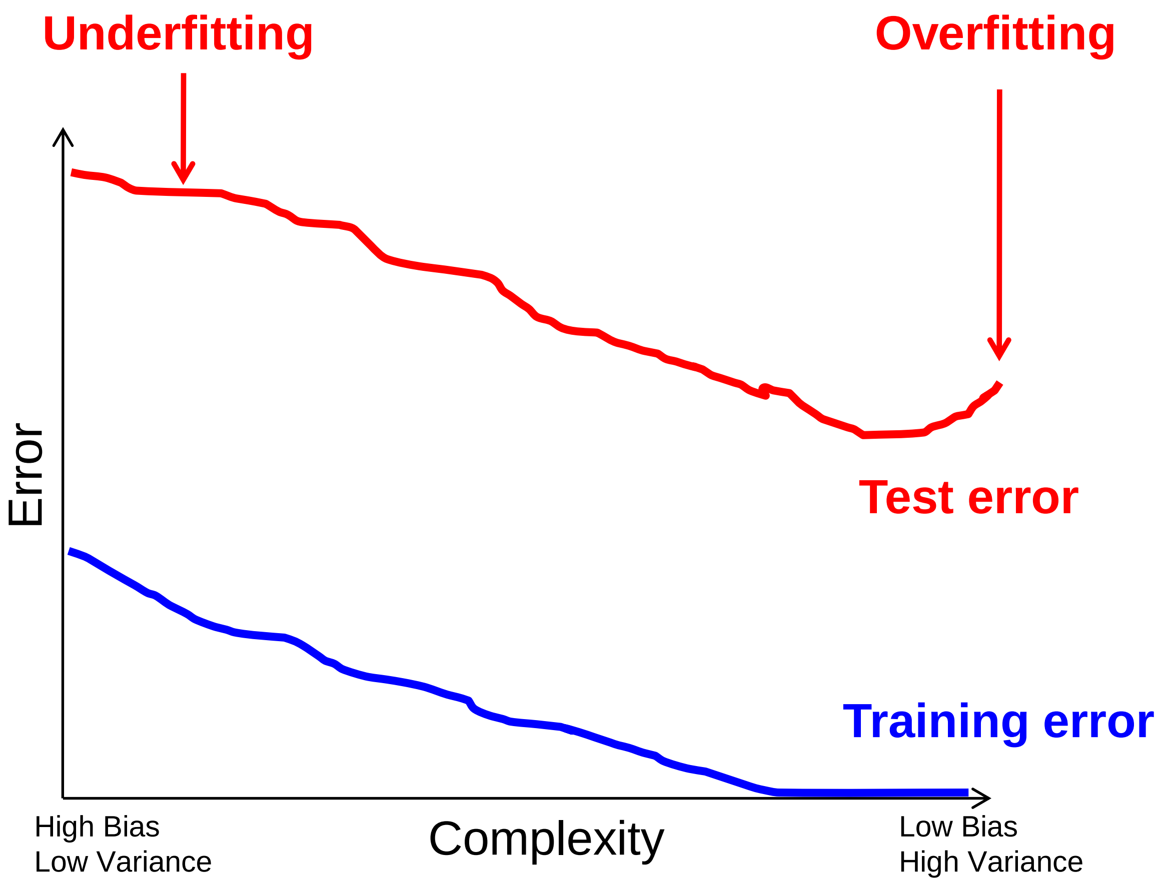
Error due to variance of training samples

See the following for explanations of bias-variance (also Bishop's "Neural Networks" book):

• http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf

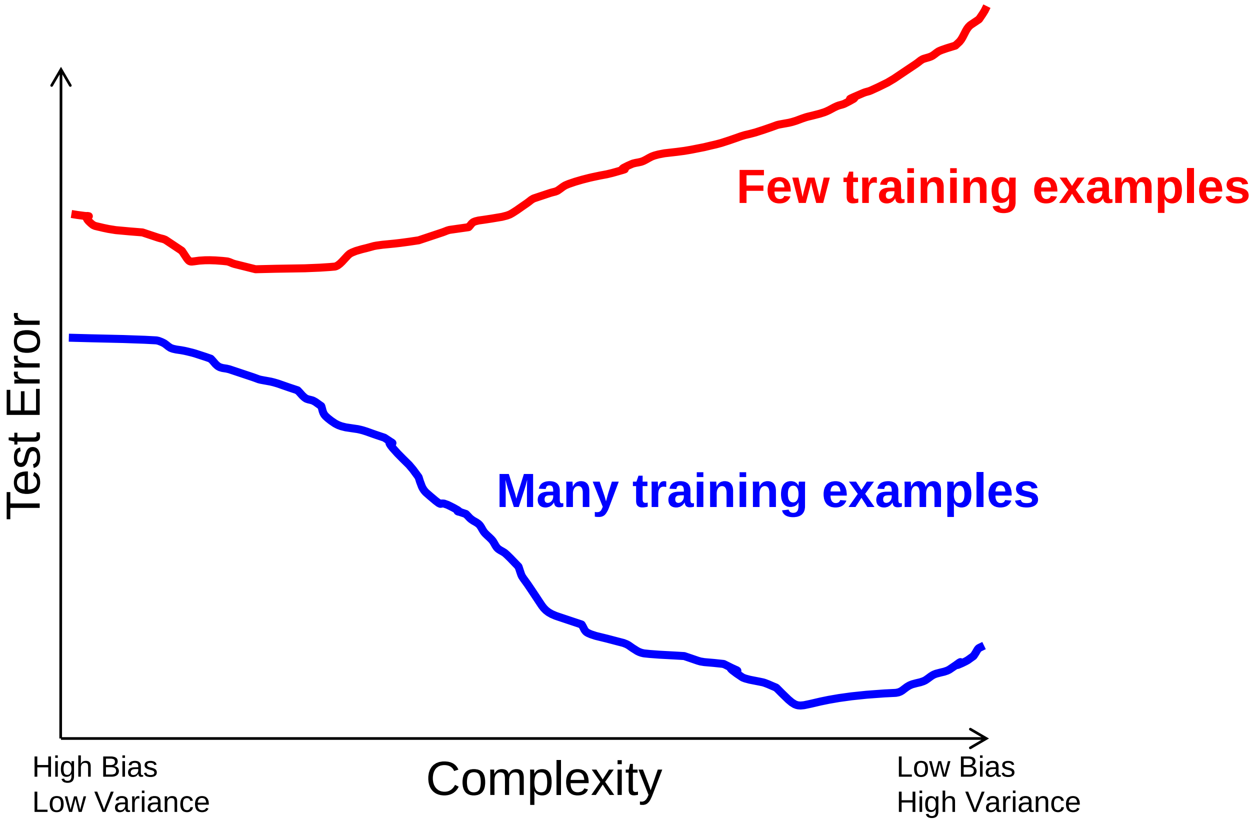
Slide credit: D. Hoiem

Bias-Variance Trade-off: Underfitting & Overfitting



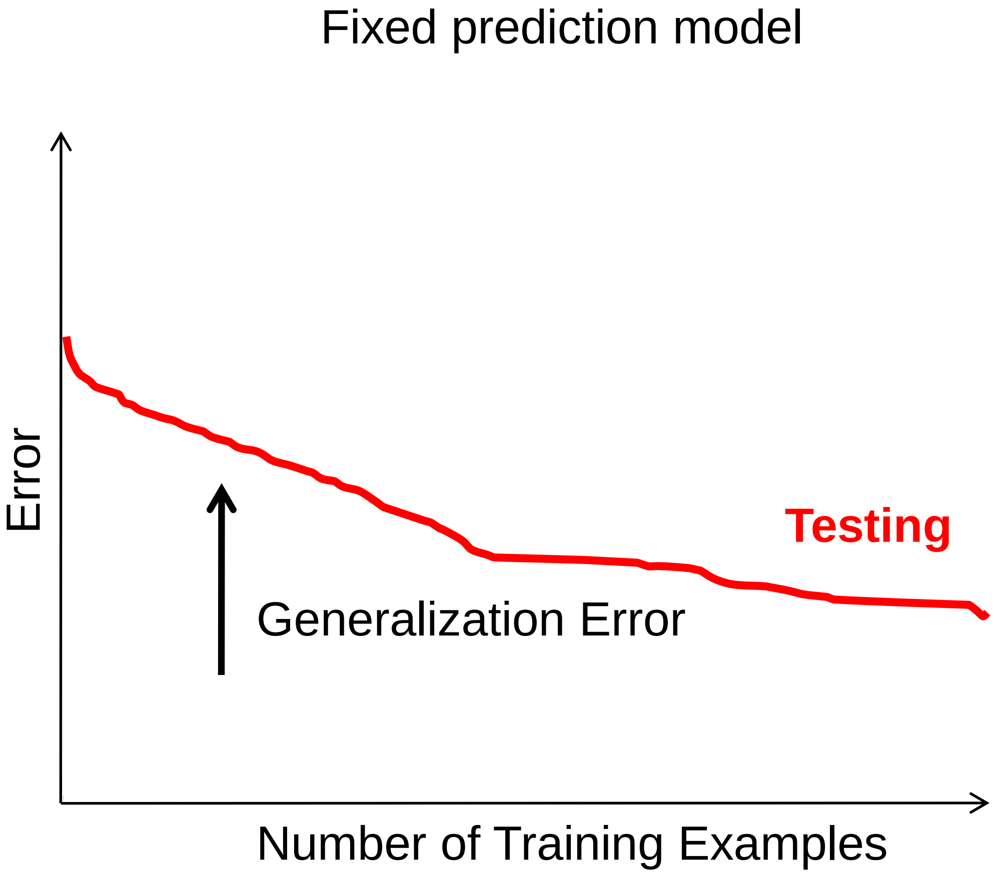
Slide credit: D. Hoiem

Bias-Variance Trade-off: Effect of Sample Size



Slide credit: D. Hoiem

Effect of Training Size on Generalization Error



Slide credit: D. Hoiem

The perfect classification algorithm

- Objective function: encodes the right loss for the problem
- Parameterization: makes assumptions that fit the problem
- Regularization: right level of regularization for amount of training data
- Training algorithm: can find parameters that maximize objective on training set
- Inference algorithm: can solve for the objective function in evaluation

Slide credit: D. Hoiem

Remember…

- No classifier is inherently better than any other: you need to make assumptions to generalize

- Three kinds of error
  - Inherent: unavoidable
  - Bias: due to over-simplifications
  - Variance: due to inability to perfectly estimate parameters from limited data

How to reduce variance?

- Choose a simpler classifier
  - Occam's Razor: *Among competing hypotheses, the one with the fewest assumptions should be selected.*
- Regularize the parameters
  - Think of L1/L2 penalties in regression
  - Think of Laplacian/Gaussian priors from a Bayesian probabilistic perspective
- Get more training data
  - ***BIG*** data

Slide credit: D. Hoiem

Many Models / Classifiers!

- Supervised learning categories and techniques
  - **Linear classifier** (numerical functions)
  - **Parametric** (Probabilistic functions)
    - Naïve Bayes, Gaussian discriminant analysis (GDA), Hidden Markov models (HMM), Probabilistic graphical models
  - **Non-parametric** (Instance-based functions)
    - *K*-nearest neighbors, Kernel regression, Kernel density estimation, Local regression
  - **Non-metric** (Symbolic functions)
    - Classification and regression tree (CART), decision tree
  - **Aggregation**
    - Bagging (bootstrap + aggregation), Adaboost, Random forest
- Unsupervised learning categories and techniques
  - **Clustering**
    - K-means clustering / Spectral clustering
  - **Density Estimation**
    - Gaussian mixture model (GMM)
    - Graphical models
  - **Dimensionality reduction**
    - Principal component analysis (PCA)
    - Factor analysis

Generative vs. Discriminative Classifiers

Generative Models

- Represent both the data and the labels
- Often, makes use of conditional independence and priors
- Examples
  - Naïve Bayes classifier
  - Bayesian network

- Models of data may apply to future prediction problems

Discriminative Models

- Learn to directly predict the labels from the data
- Often, assume a simple boundary (e.g., linear)
- Examples
  - Logistic regression
  - SVM
  - Boosted decision trees

- Often easier to predict a label from the data than to model the data

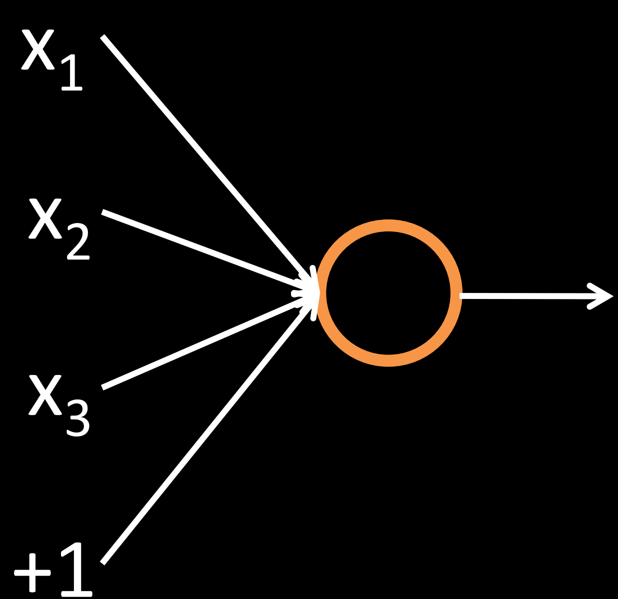
Slide credit: D. Hoiem

Logistic regression has a learned parameter vector θ.
On input x, it outputs:

$$\begin{aligned} h_\theta(x) &= \sigma(\theta^T x) \\ &= \frac{1}{1+\exp(-\theta^T x)} \end{aligned}$$

where $\sigma(z) = 1/(1+\exp(-z))$

Draw a logistic regression unit as:



$$h_\theta(x) = \frac{1}{1+\exp(-\theta^T x)}$$

Comparison

assuming x in {0 1}

	Learning Objective	Training	Inference
Naïve Bayes	$\text{maximize} \sum_i \left[\sum_j \log P(x_{ij} \mid y_i; \theta_j) + \log P(y_i; \theta_0) \right]$	$\theta_{kj} = \dfrac{\sum_i \delta(x_{ij} = 1 \wedge y_i = k) + r}{\sum_i \delta(y_i = k) + Kr}$	$\boldsymbol{\theta}_1^T \mathbf{x} + \boldsymbol{\theta}_0^T (1 - \mathbf{x}) > 0$ where $\theta_{1j} = \log \dfrac{P(x_j = 1 \mid y = 1)}{P(x_j = 1 \mid y = 0)}$, $\theta_{0j} = \log \dfrac{P(x_j = 0 \mid y = 1)}{P(x_j = 0 \mid y = 0)}$
Logistic Regression	$\text{maximize} \sum_i \log(P(y_i \mid \mathbf{x}, \boldsymbol{\theta})) + \lambda \lVert \boldsymbol{\theta} \rVert$ where $P(y_i \mid \mathbf{x}, \boldsymbol{\theta}) = 1/(1 + \exp(-y_i \boldsymbol{\theta}^T \mathbf{x}))$	Gradient ascent	$\boldsymbol{\theta}^T \mathbf{x} > 0$
Linear SVM	$\text{minimize } \lambda \sum_i \xi_i + \frac{1}{2} \lVert \boldsymbol{\theta} \rVert$ such that $y_i \boldsymbol{\theta}^T \mathbf{x} \geq 1 - \xi_i \quad \forall i$	Linear programming	$\boldsymbol{\theta}^T \mathbf{x} > 0$
Kernelized SVM	complicated to write	Quadratic programming	$\sum_i y_i \alpha_i K(\hat{\mathbf{x}}_i, \mathbf{x}) > 0$
Nearest Neighbor	most similar features → same label	Record data	y_i where $i = \underset{i}{\operatorname{argmin}} \; K(\hat{\mathbf{x}}_i, \mathbf{x})$

Slide credit: D. Hoiem

What to remember about classifiers

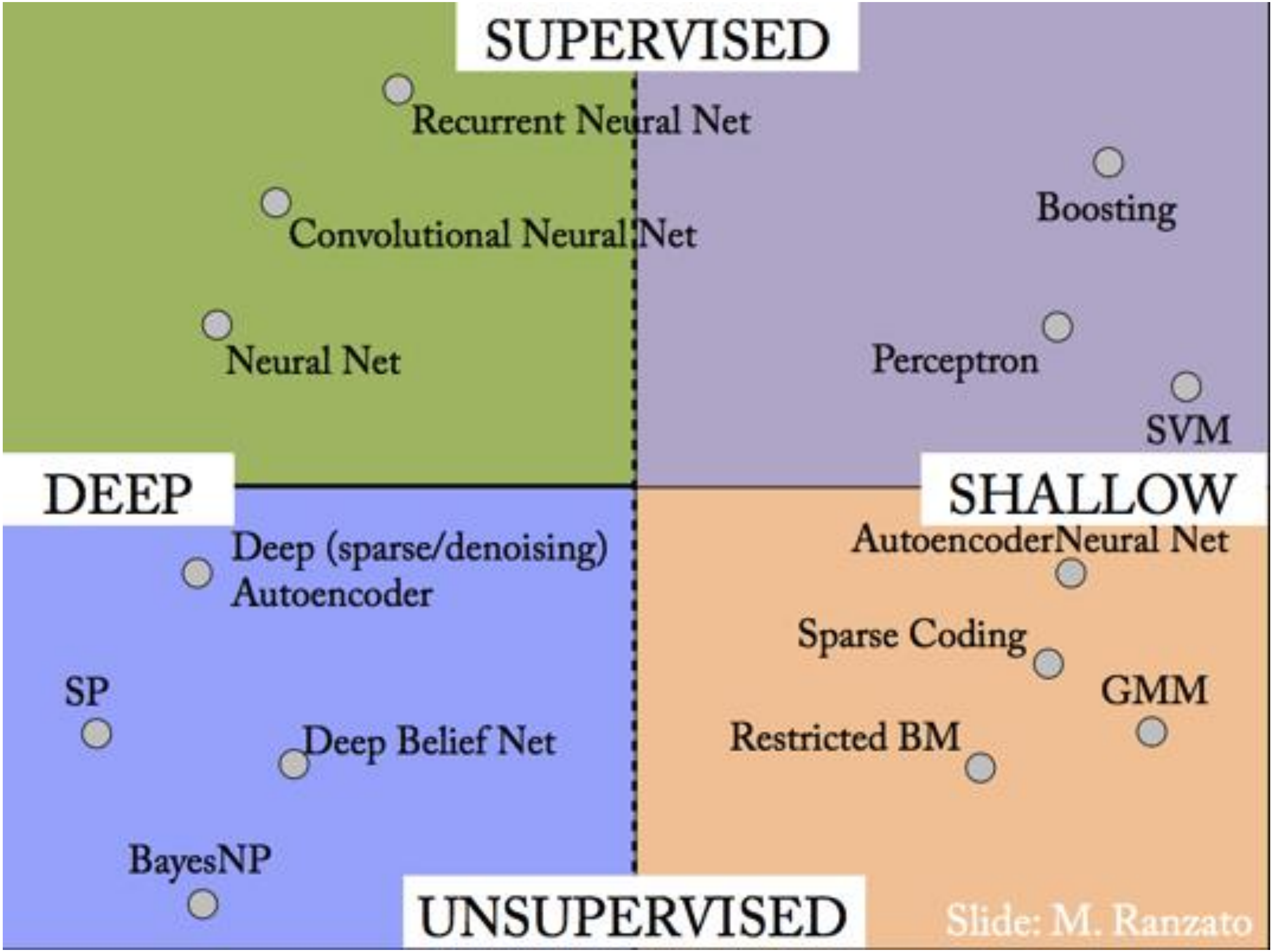
- No free lunch: machine learning algorithms are tools, not dogmas

- Try simple classifiers first

- Better to have smart features and simple classifiers than simple features and smart classifiers

- Use increasingly powerful classifiers with more training data (bias-variance tradeoff)

Slide credit: D. Hoiem


SUPERVISED
Recurrent Neural Net
Convolutional Neural Net
Neural Net
Boosting
Perceptron
SVM
DEEP
SHALLOW
Deep (sparse/denoising) Autoencoder
SP
Deep Belief Net
BayesNP
AutoencoderNeural Net
Sparse Coding
GMM
Restricted BM
UNSUPERVISED
Slide: M. Ranzato

Areas We Will Study Together

- Representation learning
- Generative models
- Variational inference
- Reinforcement learning
- Recurrence and temporal learning
- Neurobiological learning / brain-inspired computing
- Uncertainty
- Graph neural networks

References

- Slides/content were adapted from:
  - "Deep Learning" (Machine Learning Basics, Chapter 5, Goodfellow et al., 2016)
  - "An Overview of Machine Learning" (Yi-Fan Chang, 2011)
  - "CSE 446 Machine Learning" (intro) (Pedro Domingos)
  - "Feature learning for image classification" (Kai Yu and Andrew Ng)
  - "Machine Learning" (Computer Vision) James Hays, Brown
- Andrew Ng's Machine Learning course/lectures:
  - http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=MachineLearning
- Data Mining textbook : "Data Mining: Concepts and Techniques, Third Edition (The Morgan Kaufmann Series in Data Management Systems)" Han et al. 2011

Some Machine Learning References

- General
  - Tom Mitchell, *Machine Learning*, McGraw Hill, 1997
  - Christopher Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995
- Adaboost (to learn about Boosting)
  - Friedman, Hastie, and Tibshirani, “Additive logistic regression: a statistical view of boosting”, Annals of Statistics, 2000
- SVMs
  - http://www.support-vector.net/icml-tutorial.pdf

Slide credit: D. Hoiem