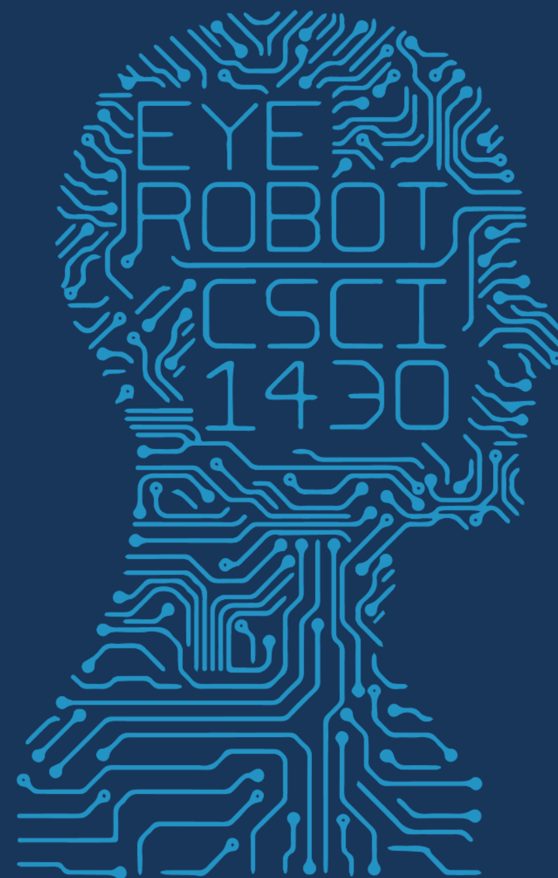
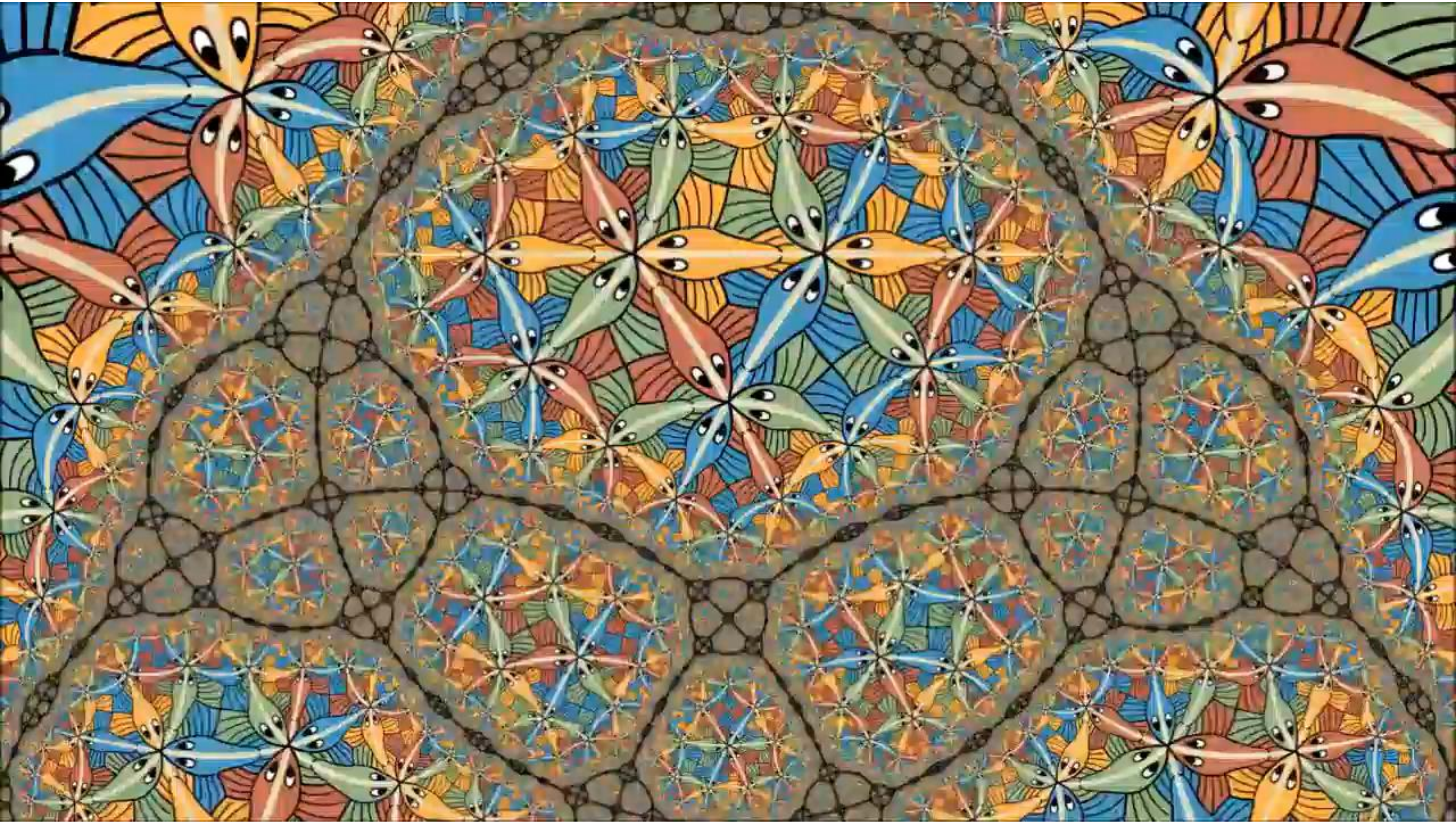
1950

Future Vision

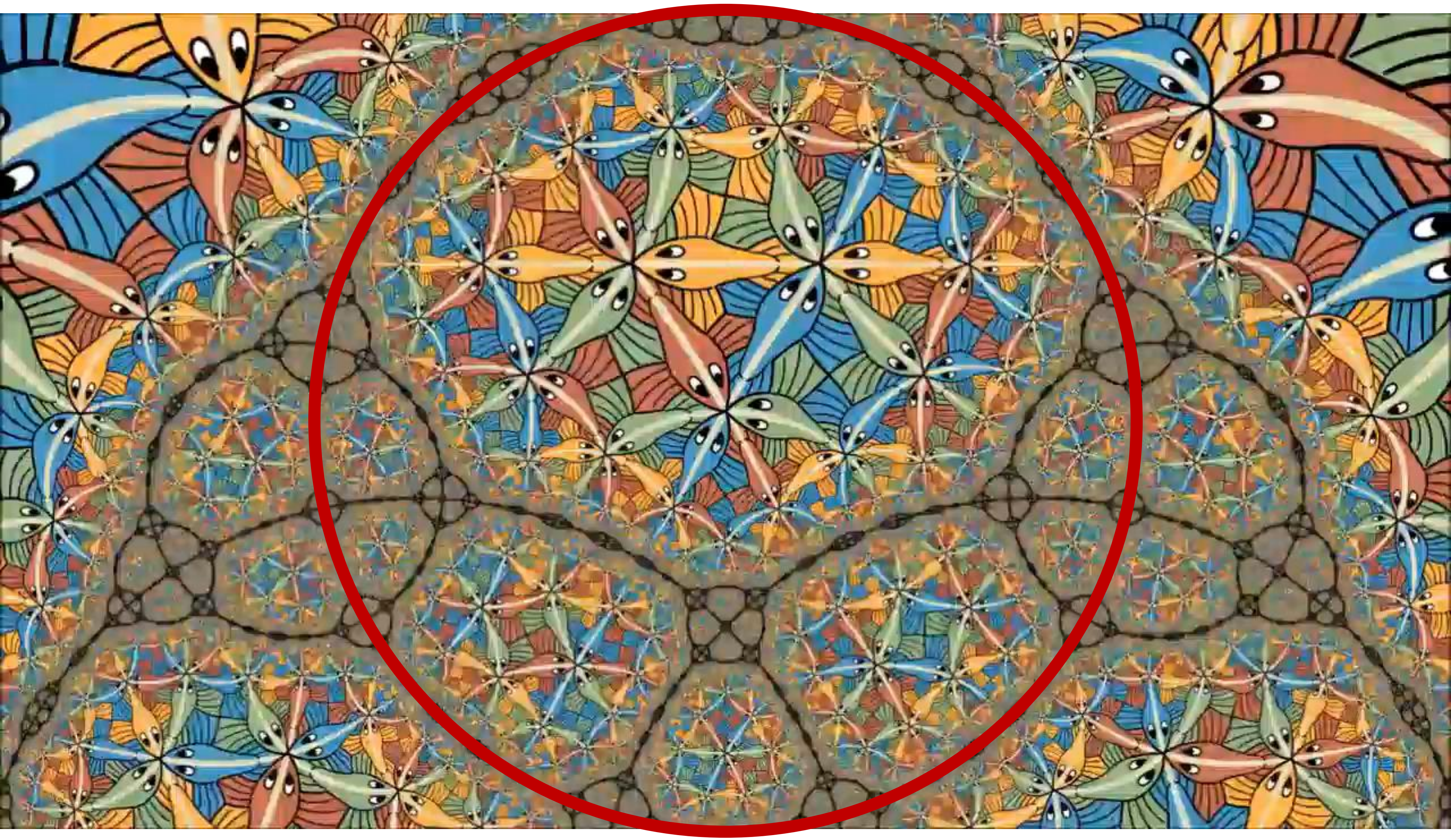


2017 MWF 1pm

Computer Vision



Escher’s Circle Limit III



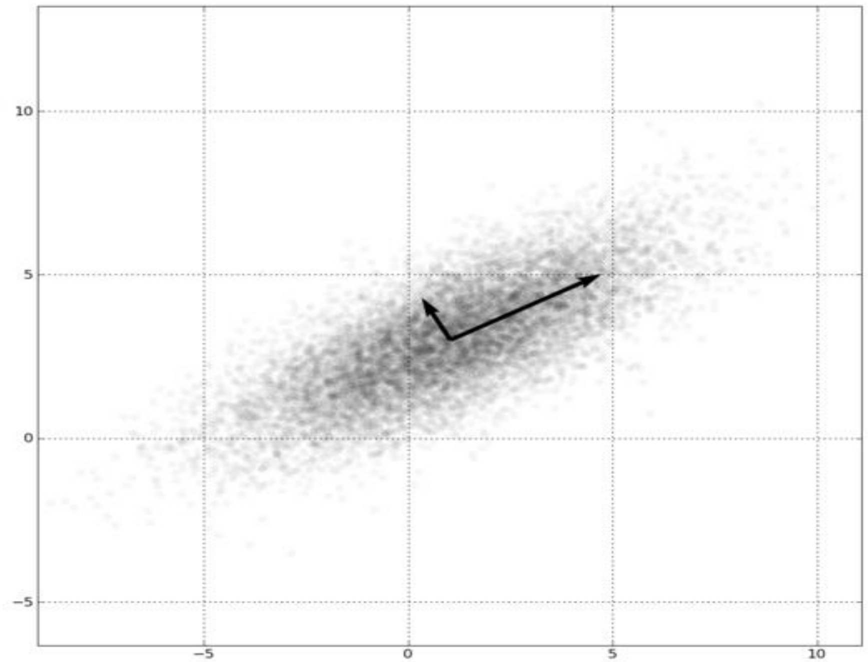
Escher’s Circle Limit III

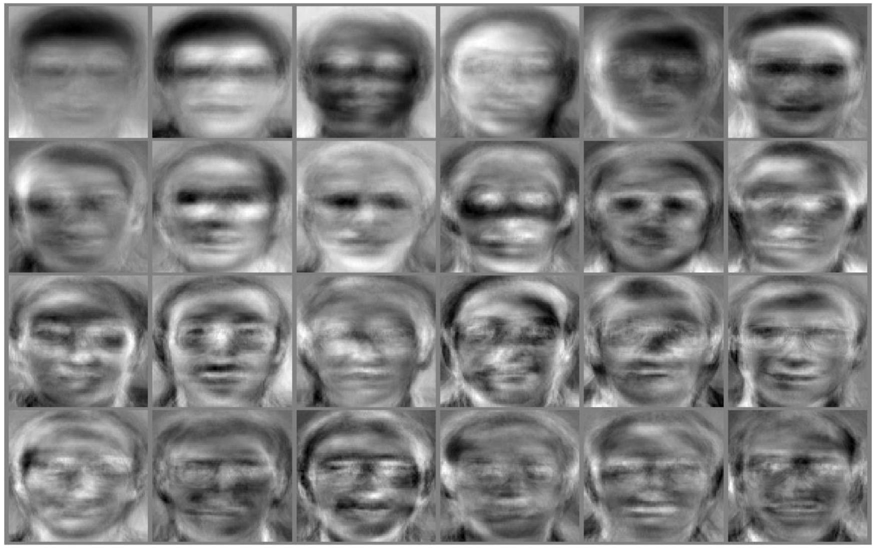
Machine Learning Problems

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

PCA: Principal Component Analysis

- The best possible lower dimensional representation based on linear projections.
- A basis of directions of variance ordered by their significance.
- Throw away least variance dimensions to reduce data representation.





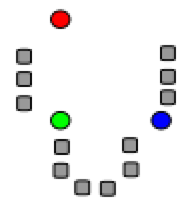
R.P.W. Duin

Machine Learning Problems

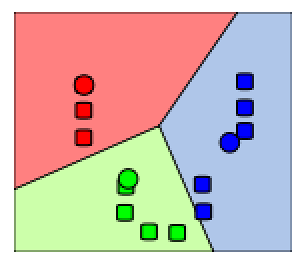
	Supervised Learning	***Unsupervised Learning***
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

K-means algorithm

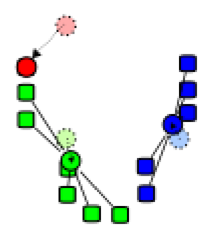
1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster



Back to 2

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

More techniques in notes

- 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 probability density function.
- 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

ImageNet

- Images for each category of WordNet
- 1000 classes
- 1.2mil images
- 100k test

- Top 5 error

Dataset split

Training Images

- Train classifier

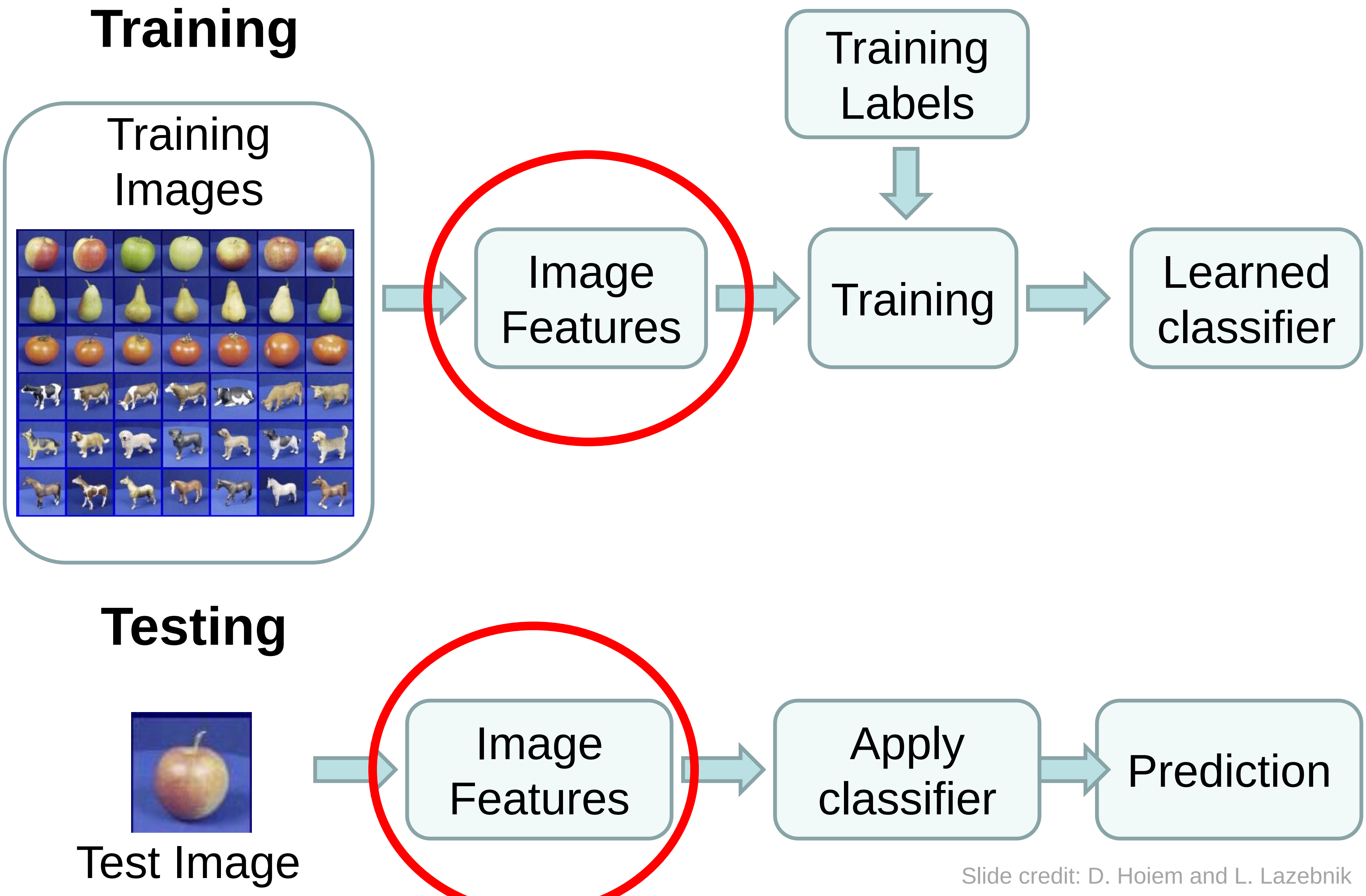
Validation Images

- Measure error
- Tune model hyperparameters

Testing Images

- Secret labels
- Measure error

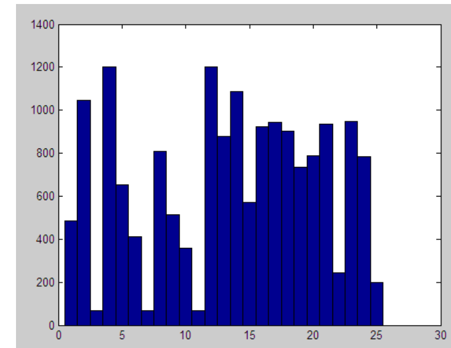
Random train/validate splits = cross validation

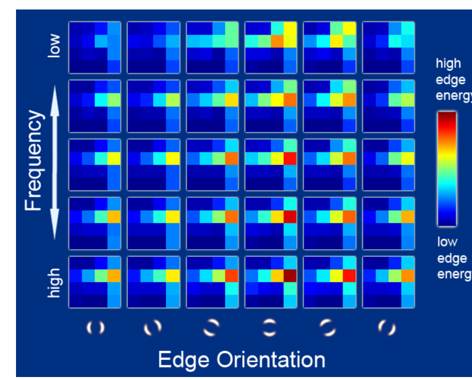


Slide credit: D. Hoiem and L. Lazebnik

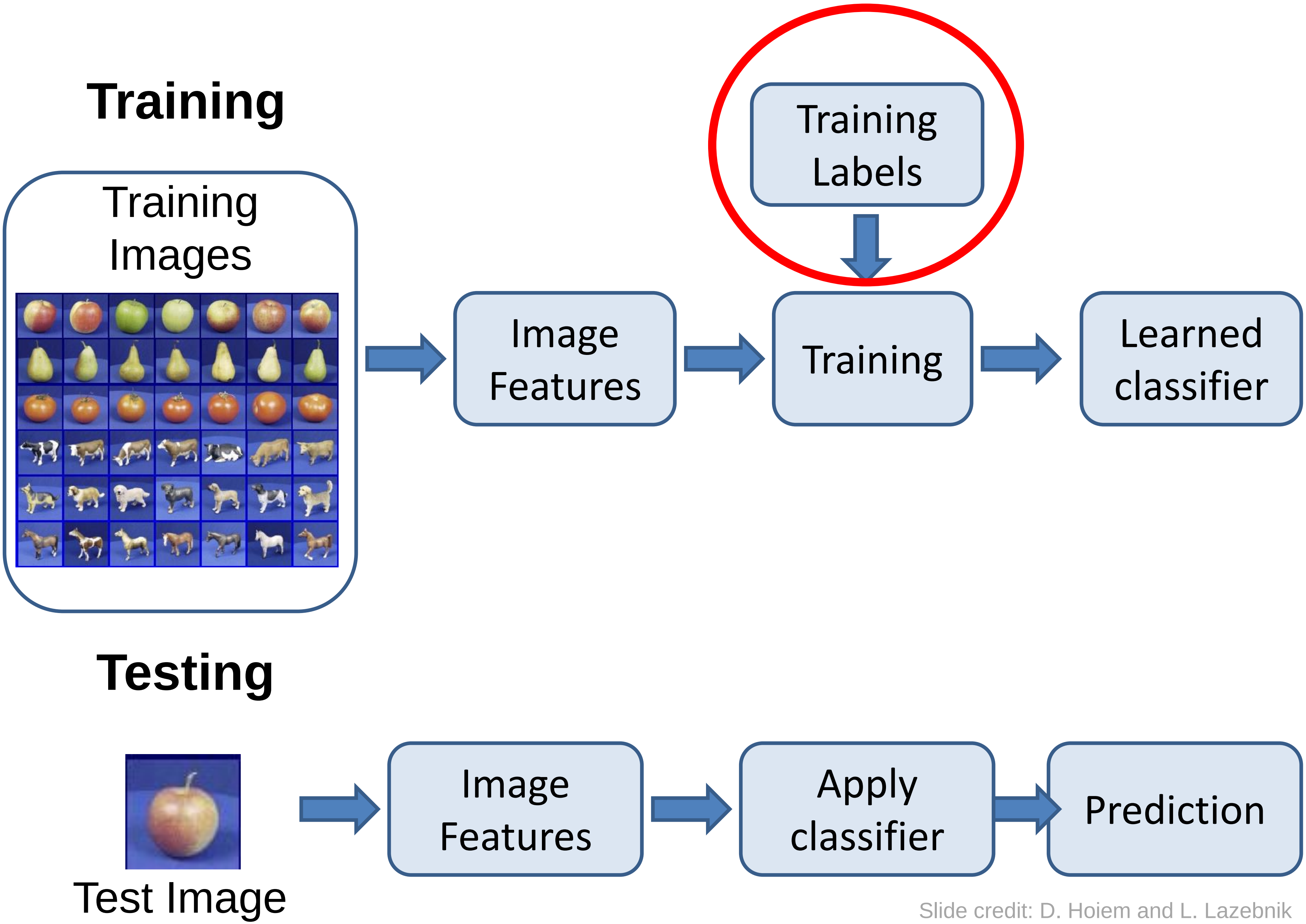
Features

- Raw pixels
- Histograms
- Templates
- SIFT descriptors
  - GIST
  - ORB
  - HOG….





L. Lazebnik

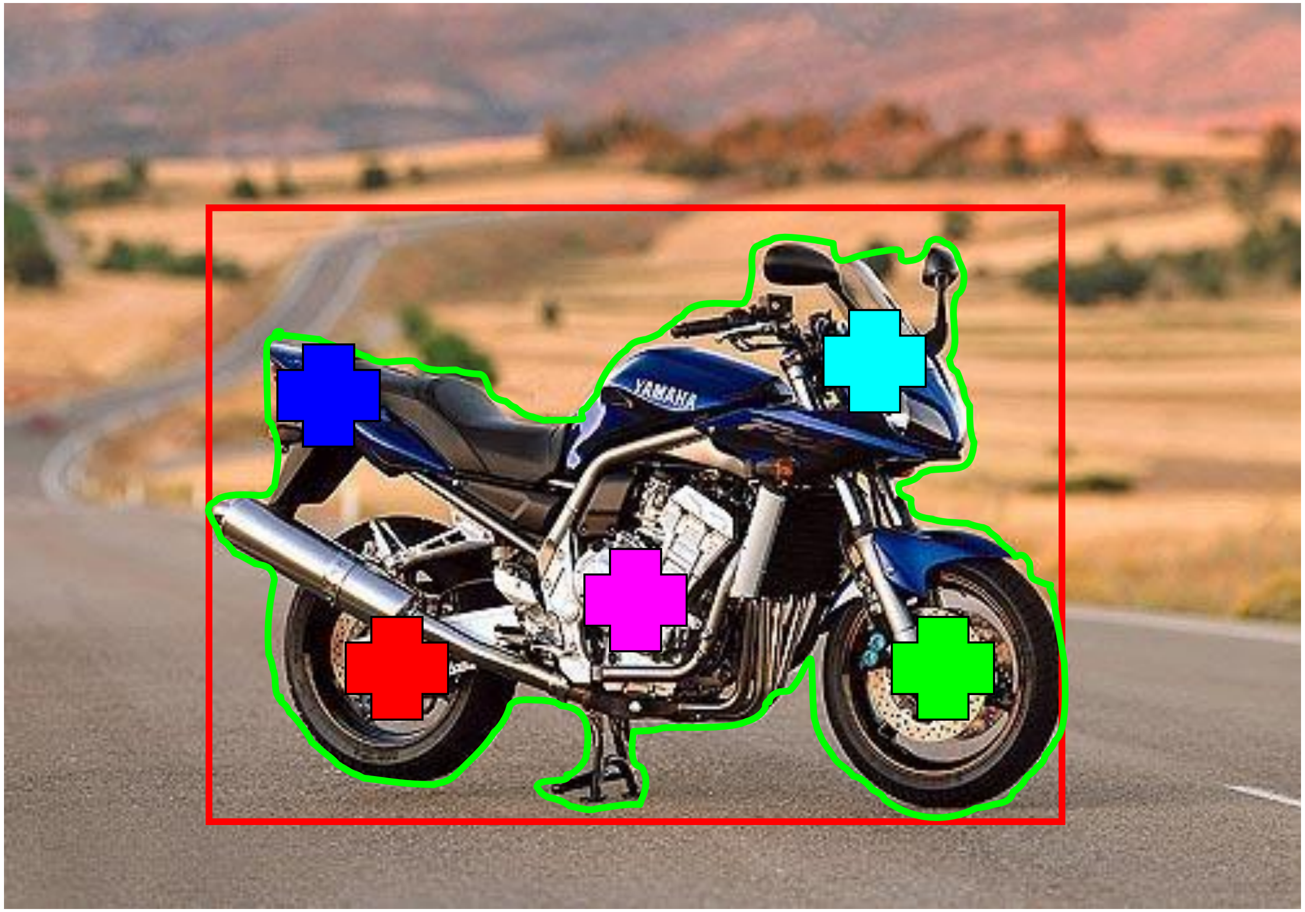


Slide credit: D. Hoiem and L. Lazebnik

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



L. Lazebnik

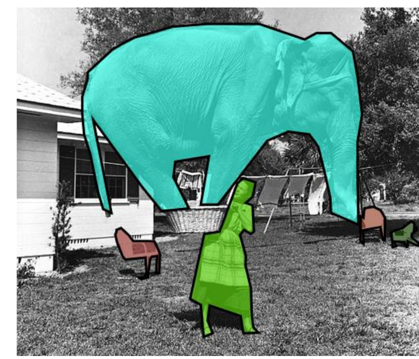
Spectrum of supervision

Less

More

E.G., ImageNet

E.G., MS Coco



Unsupervised

“Weakly” supervised

Fully supervised

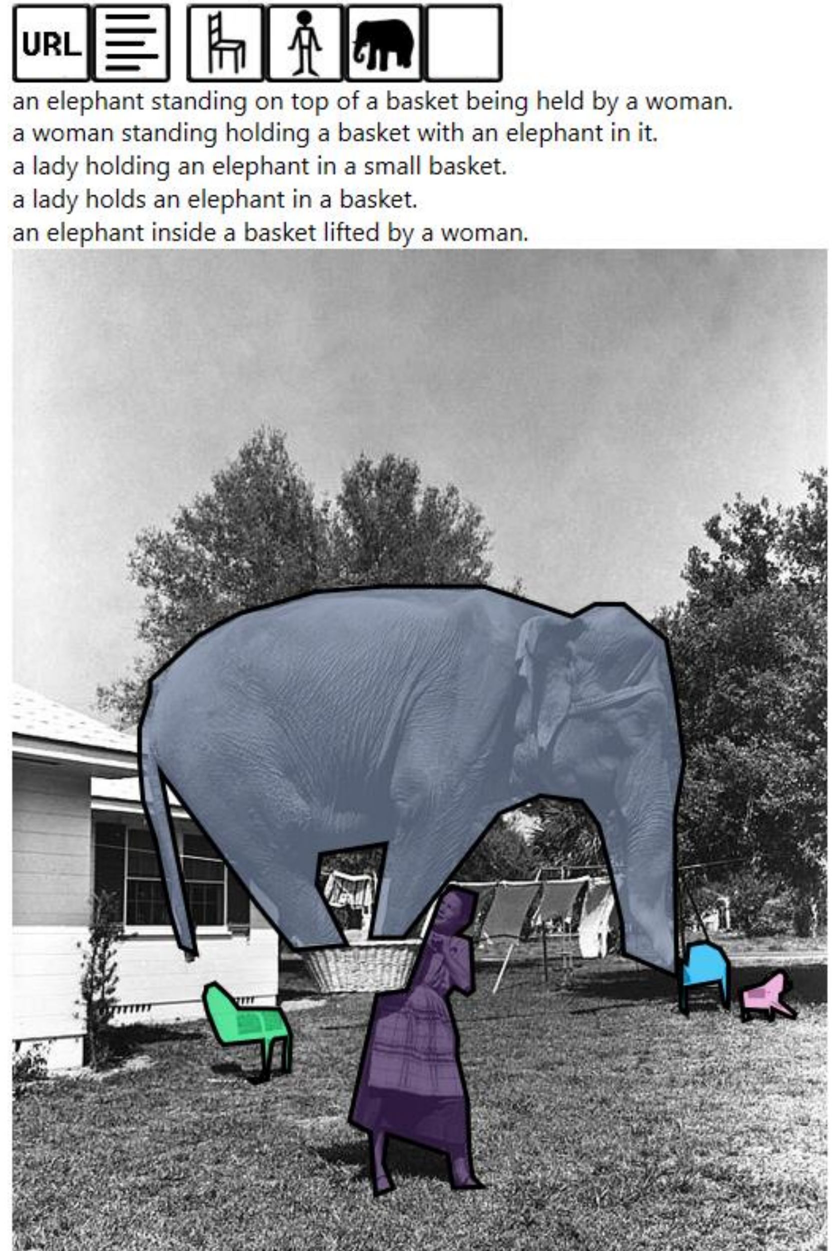
Fuzzy; definition depends on task

‘Semi-supervised’: small partial labeling

Lazebnik

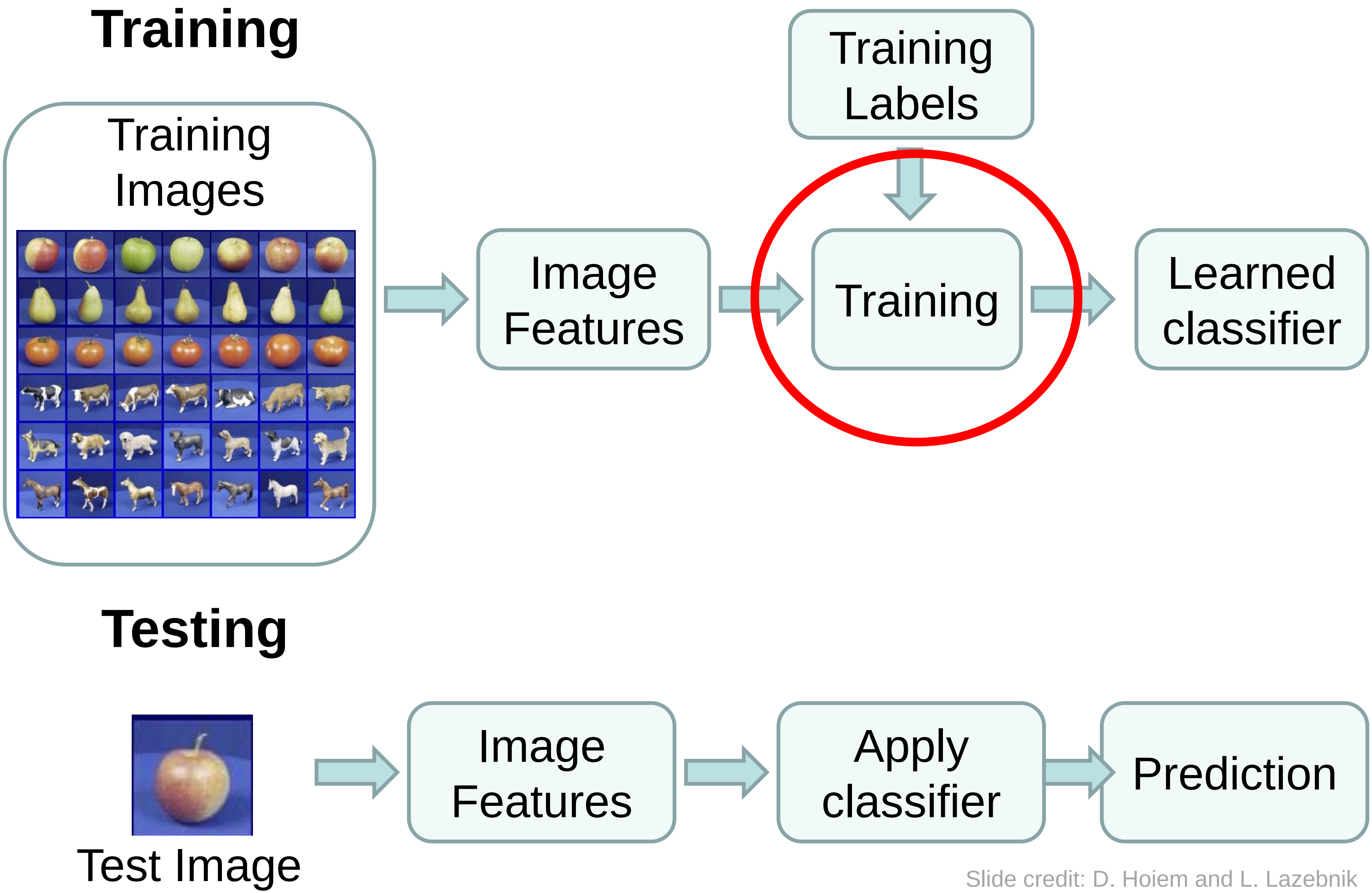
Good training example?

Good labels?



http://mscoco.org/explore/?id=134918

Google guesses from the 1st caption



Slide credit: D. Hoiem and L. Lazebnik

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

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

Prediction function or *classifier*

Image feature

Output (label)

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.

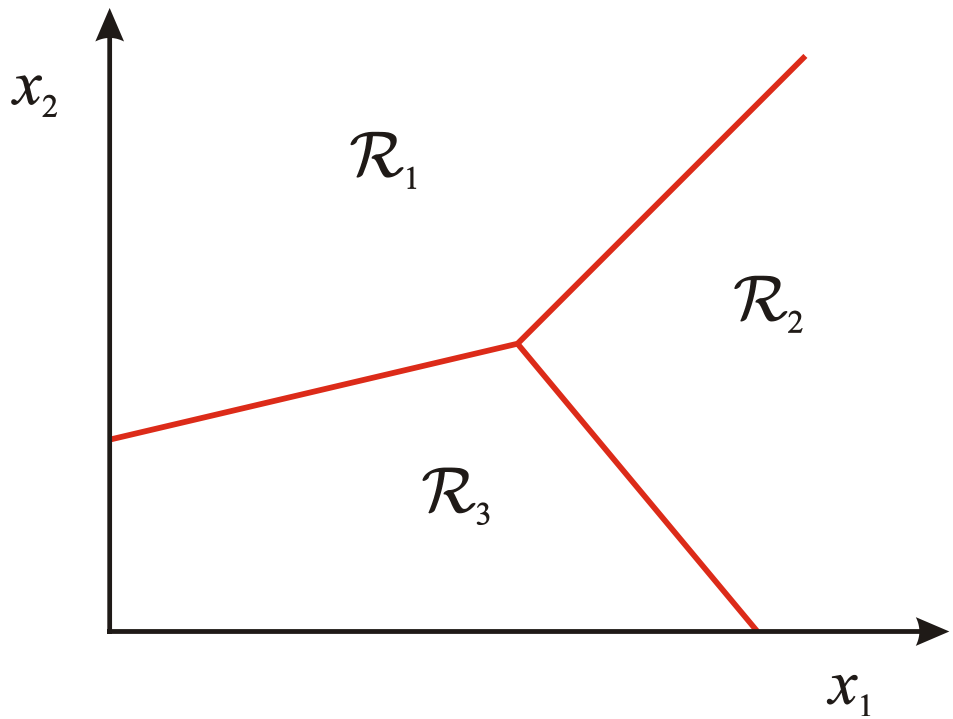
Testing: Apply f to a unseen *test example* $\mathbf{x}_u$ and output the predicted value $y_u = f(\mathbf{x}_u)$ to *classify* $\mathbf{x}_u$.

Slide credit: L. Lazebnik

Classification

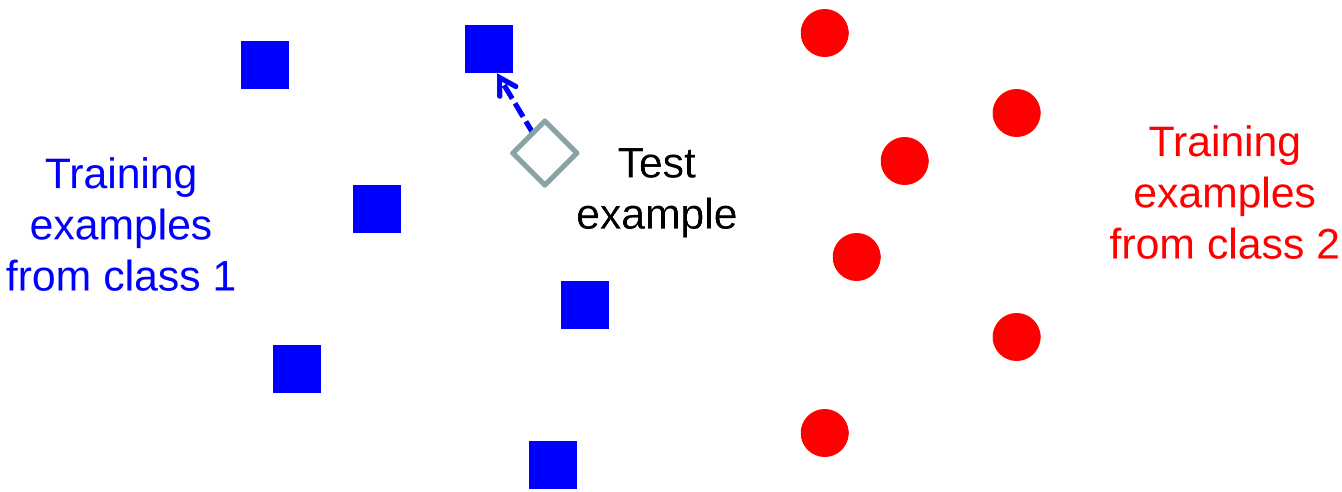
Assign **x** to one of two (or more) classes.

A decision rule divides input space into *decision regions* separated by *decision boundaries.*



L. Lazebnik

Classifiers: Nearest neighbor



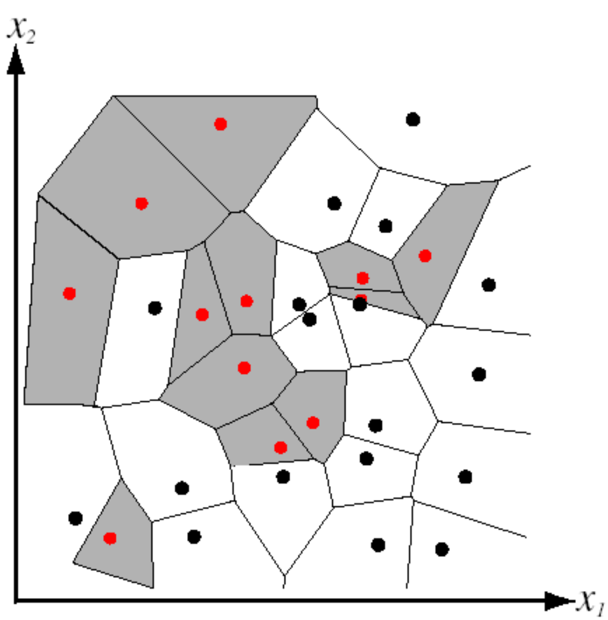
f(**x**) = label of the training example nearest to **x**

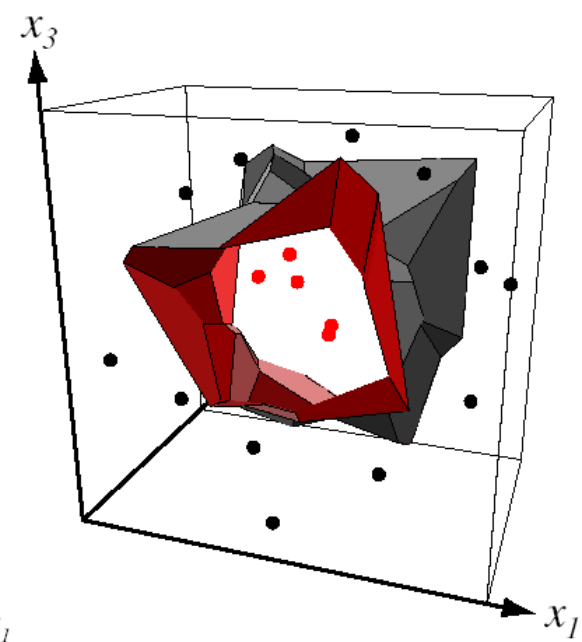
- All we need is a distance function for our inputs
- No training required!
- What does the decision boundary look like?

Slide credit: L. Lazebnik

Decision boundary for Nearest Neighbor Classifier

Divides input space into *decision regions* separated by *decision boundaries – Voronoi.*

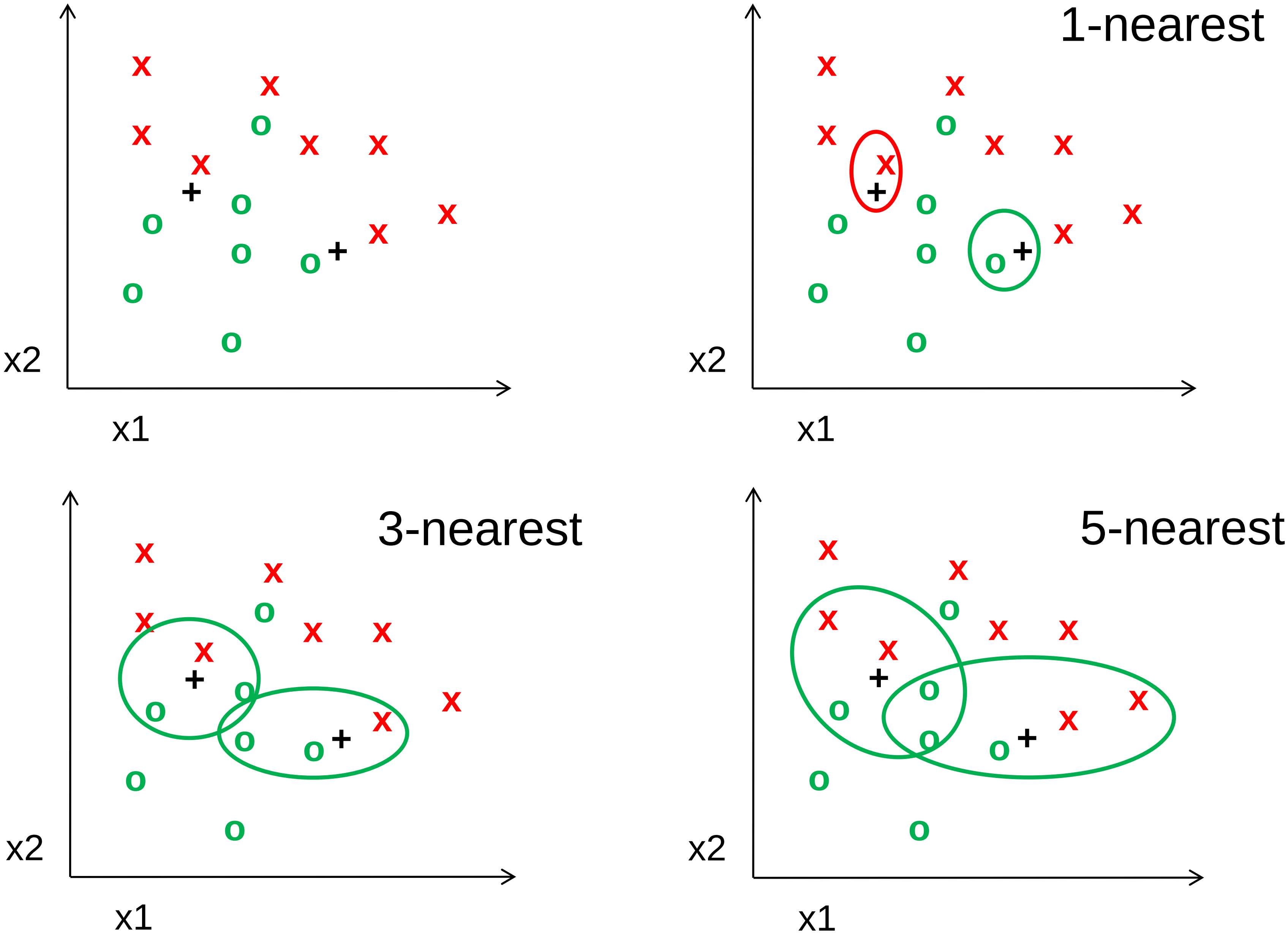




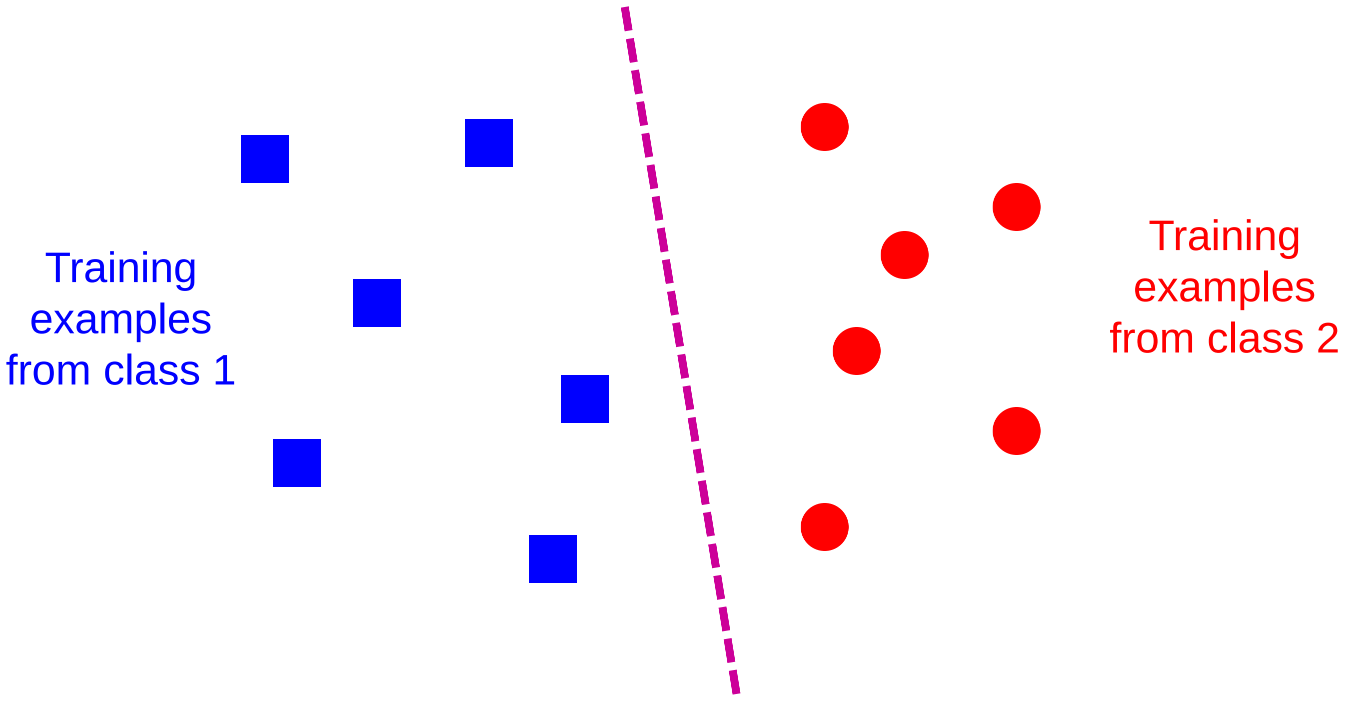
from Duda *et al.*

Voronoi partitioning of feature space for two-category 2D and 3D data

k-nearest neighbor



Classifiers: Linear



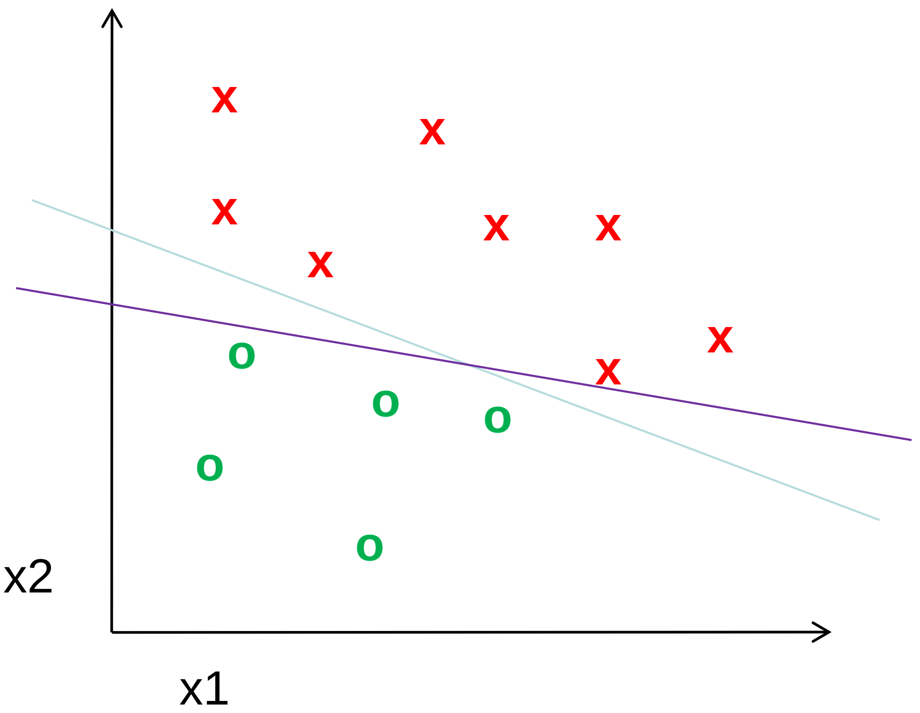
Find a *linear function* to separate the classes

Slide credit: L. Lazebnik

Classifiers: Linear SVM

Find a *linear function* to separate the classes:

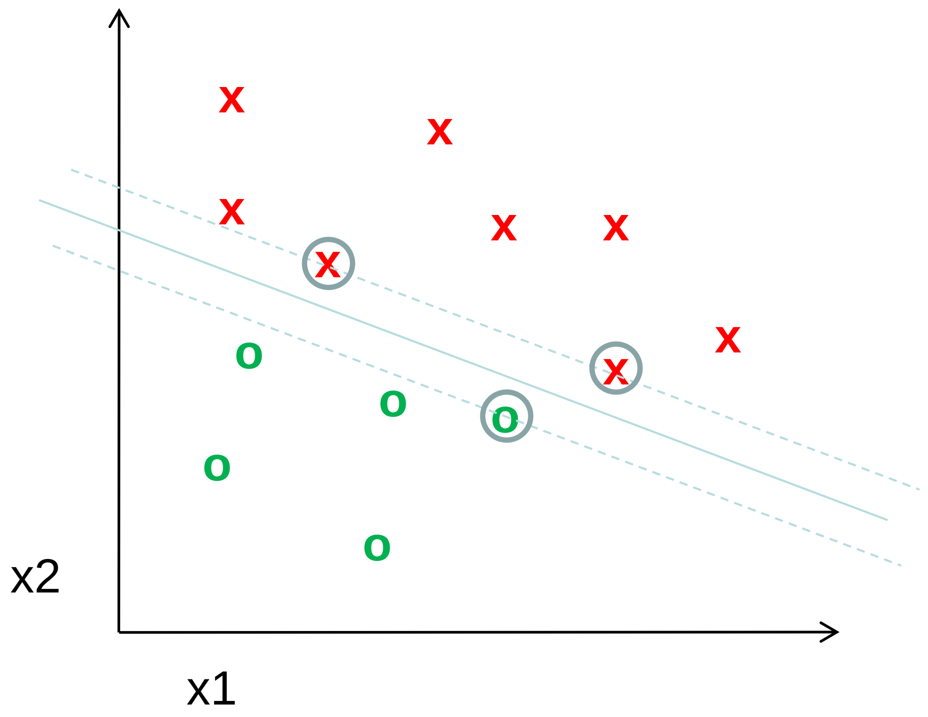
$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$



Classifiers: Linear SVM

Find a *linear function* to separate the classes:

$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$



How?

$\mathbf{X}$ = all data points

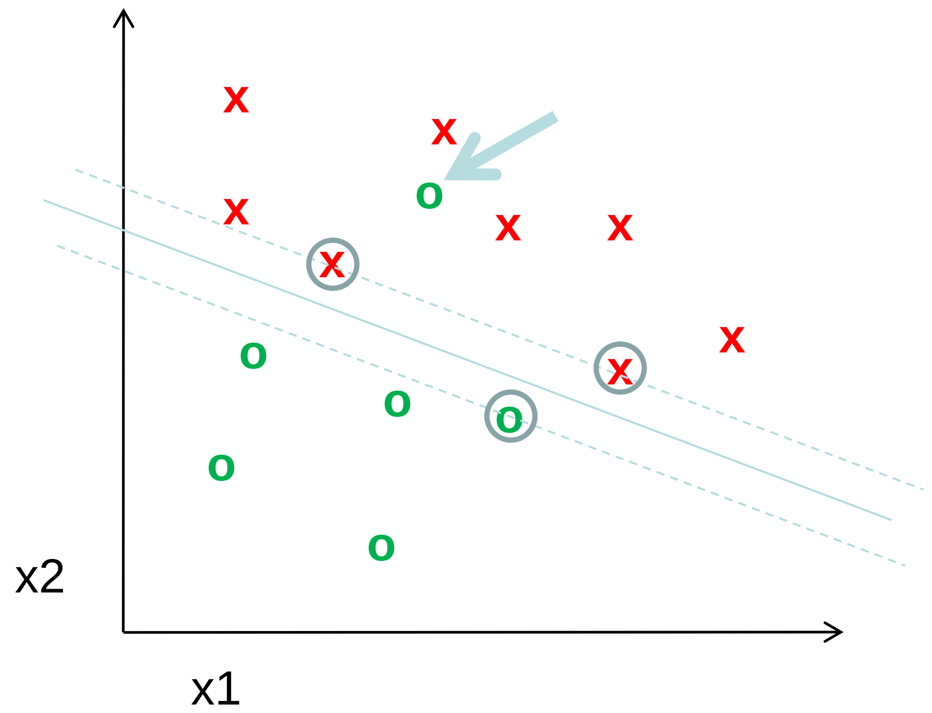
Define *hyperplane* **tX-b = 0**, where **t** is tangent to hyperplane.

Minimize **||t||** s.t. **tX-b** produces correct label for all **X**

Classifiers: Linear SVM

Find a *linear function* to separate the classes:

$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$

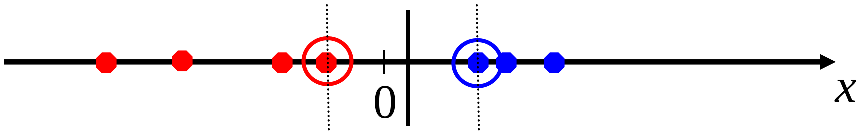


What if my data are not linearly separable?

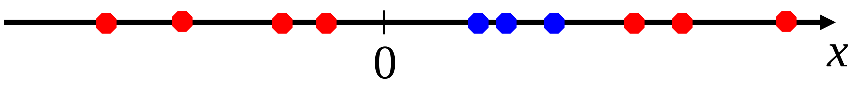
Introduce flexible ‘hinge’ loss (or ‘soft-margin’)

Nonlinear SVMs

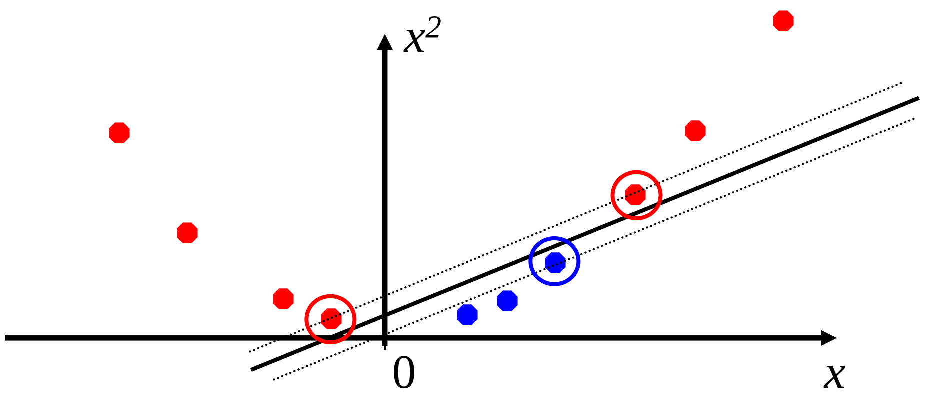
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?

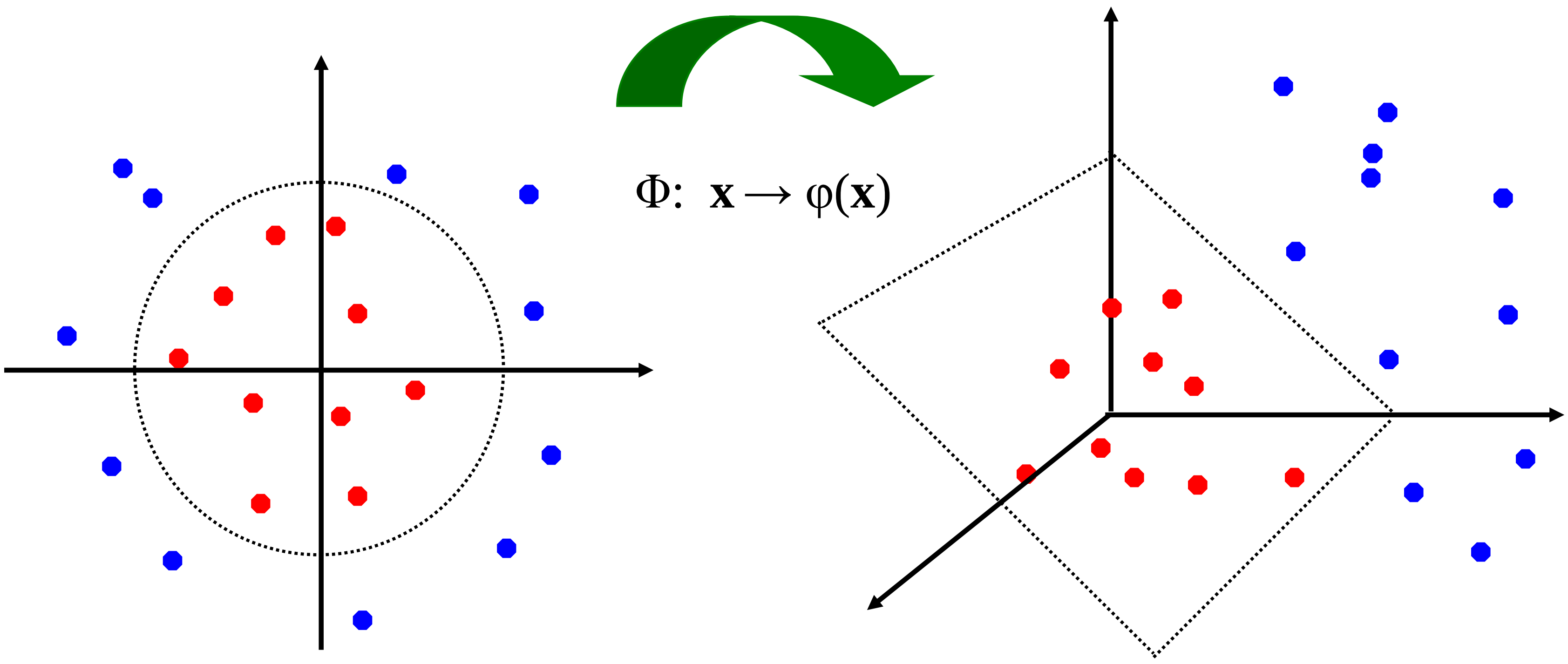


- We can map it to a higher-dimensional space:



Nonlinear SVMs

Map the original input space to some higher-dimensional feature space where the training set is separable:



Slide credit: Andrew Moore

What about multi-class SVMs?

- Unfortunately, there is no “definitive” multi-class SVM.
- In practice, we combine multiple two-class SVMs
- One vs. others
  - Training: learn an SVM for each class vs. the others
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- One vs. one
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM “votes” for a class to assign to the test example

Slide credit: L. Lazebnik

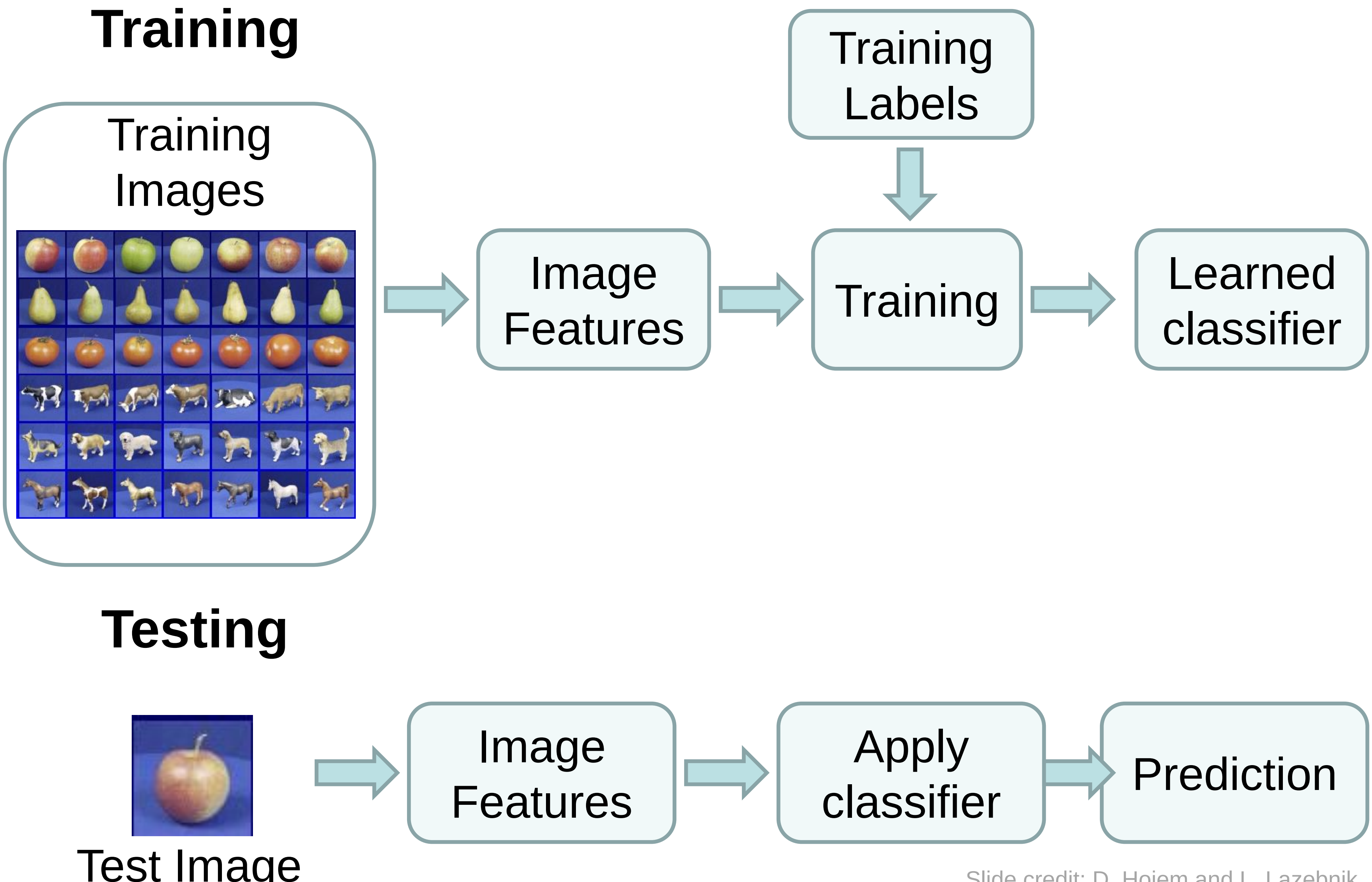
SVMs: Pros and cons

- Pros
  - Many publicly available SVM packages: http://www.kernel-machines.org/software
  - Kernel-based framework is very powerful, flexible
  - SVMs work very well in practice, even with very small training sample sizes
- Cons
  - No “direct” multi-class SVM, must combine two-class SVMs
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

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



Slide credit: D. Hoiem and L. Lazebnik

Features and **distance measures**

define visual similarity.

Training labels

dictate that examples are the same or different.

Classifiers

learn weights (or parameters) of features and distance measures…
so that visual similarity predicts label similarity.

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?

Generalization Error

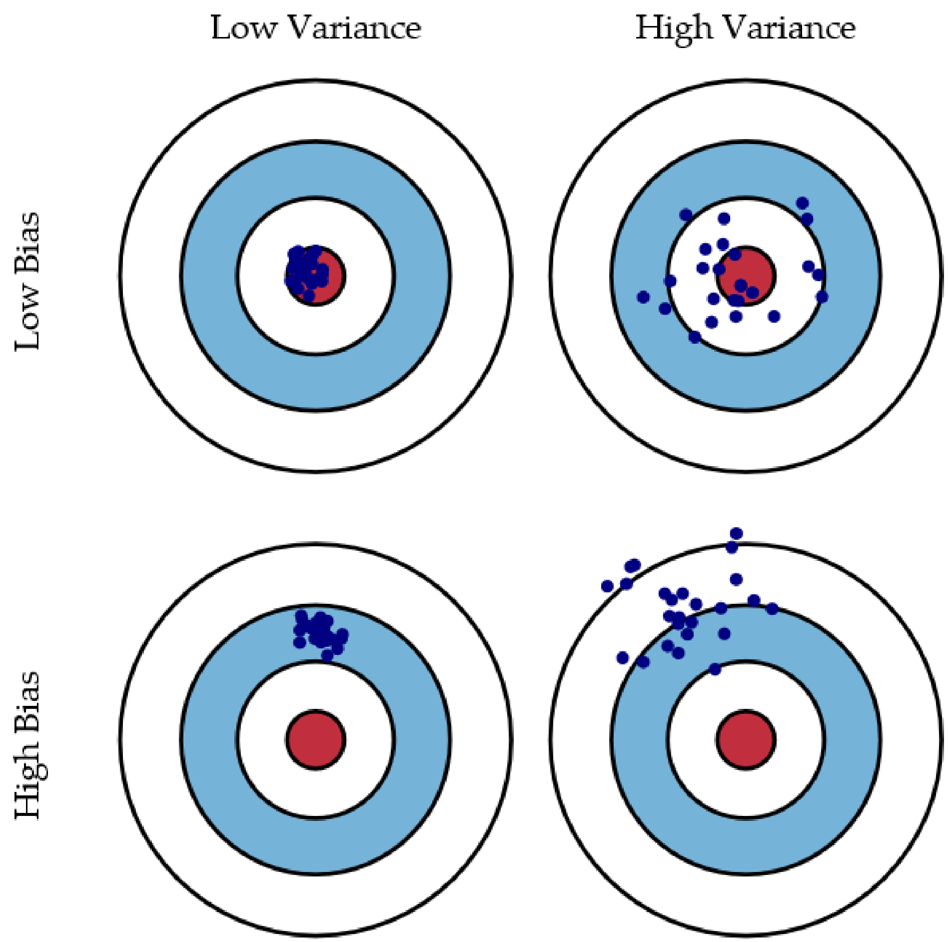
Bias:

- Difference between the expected (or average) prediction of our model and the correct value.
- Error due to inaccurate assumptions/simplifications.

Variance:

- Amount that the estimate of the target function will change if different training data was used.

Bias/variance trade-off



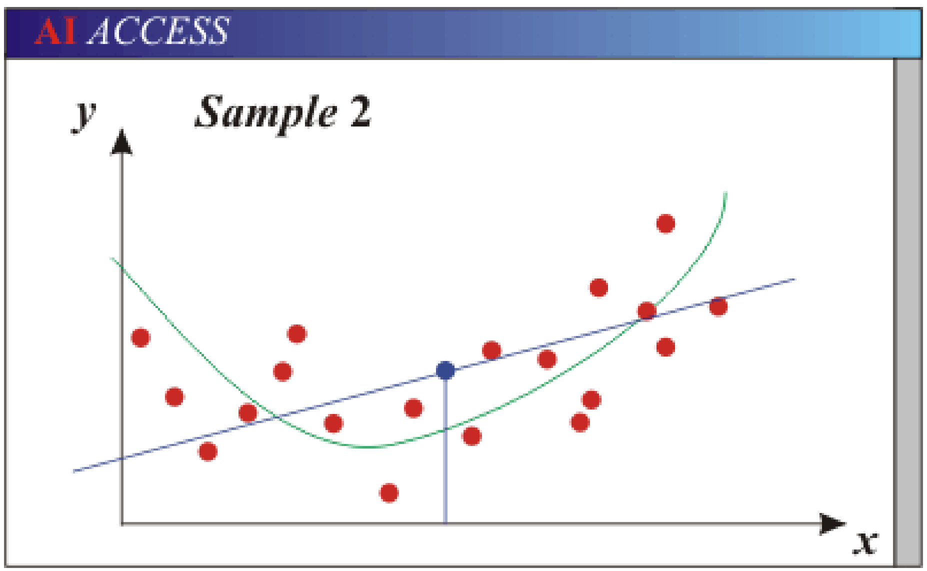
Bias = accuracy

Variance = precision

Scott Fortmann-

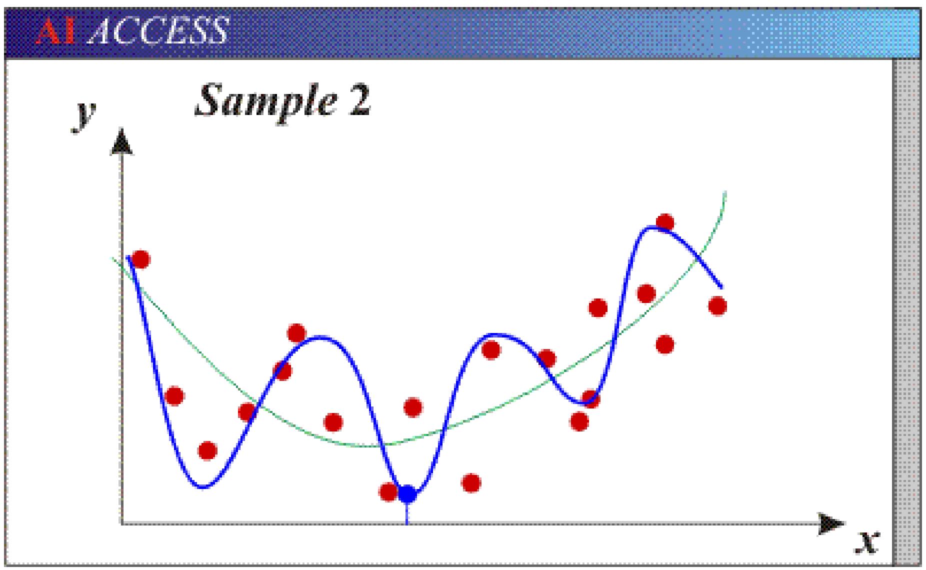
Generalization Error Effects

- **Underfitting:** model is too “simple” to represent all the relevant class characteristics
  - High bias (few degrees of freedom) and low variance
  - High training error and high test error

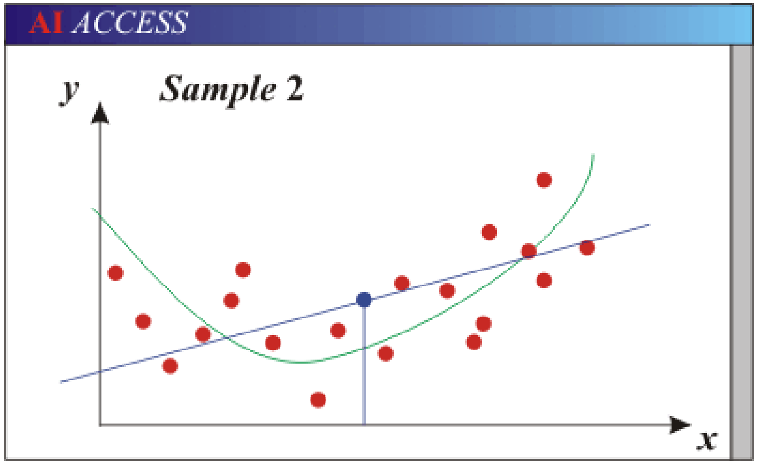


Generalization Error Effects

- **Overfitting:** model is too “complex” and fits irrelevant characteristics (noise) in the data
  - Low bias (many degrees of freedom) and high variance
  - Low training error and high test error

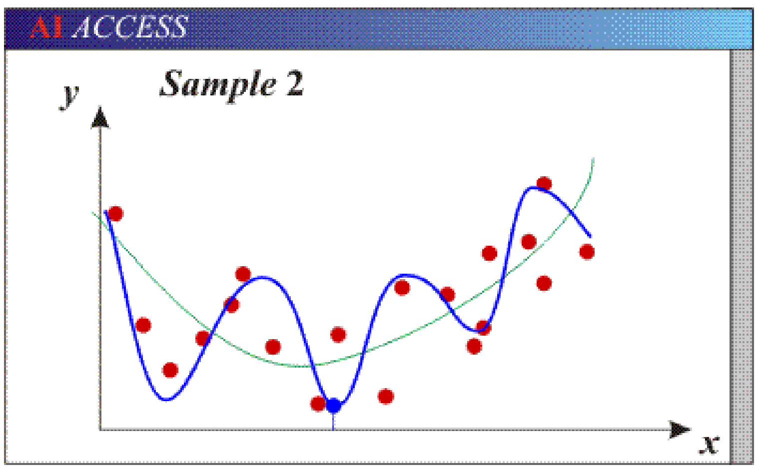


Bias-Variance Trade-off



Models with too few parameters are inaccurate because of a large bias.

- Not enough flexibility!
- Too many assumptions

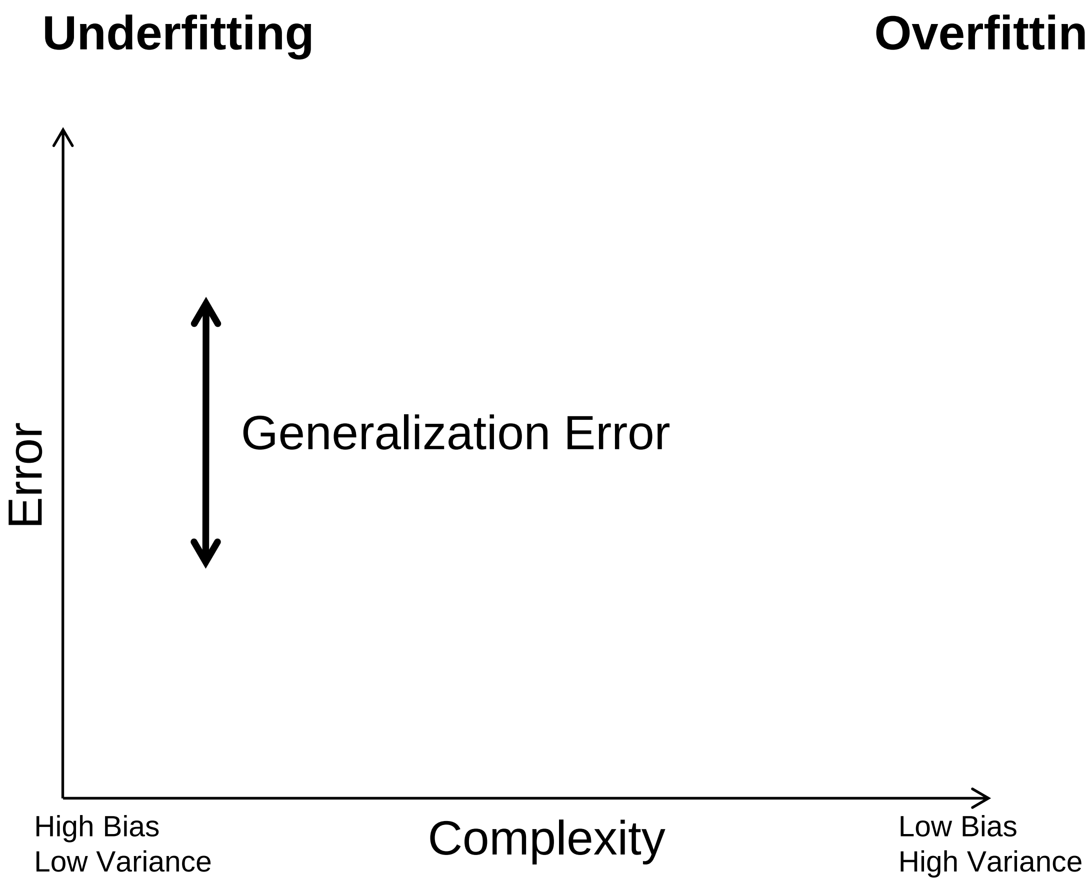


Models with too many parameters are inaccurate because of a large variance.

- Too much sensitivity to the sample.
- Slightly different data -> very different function.

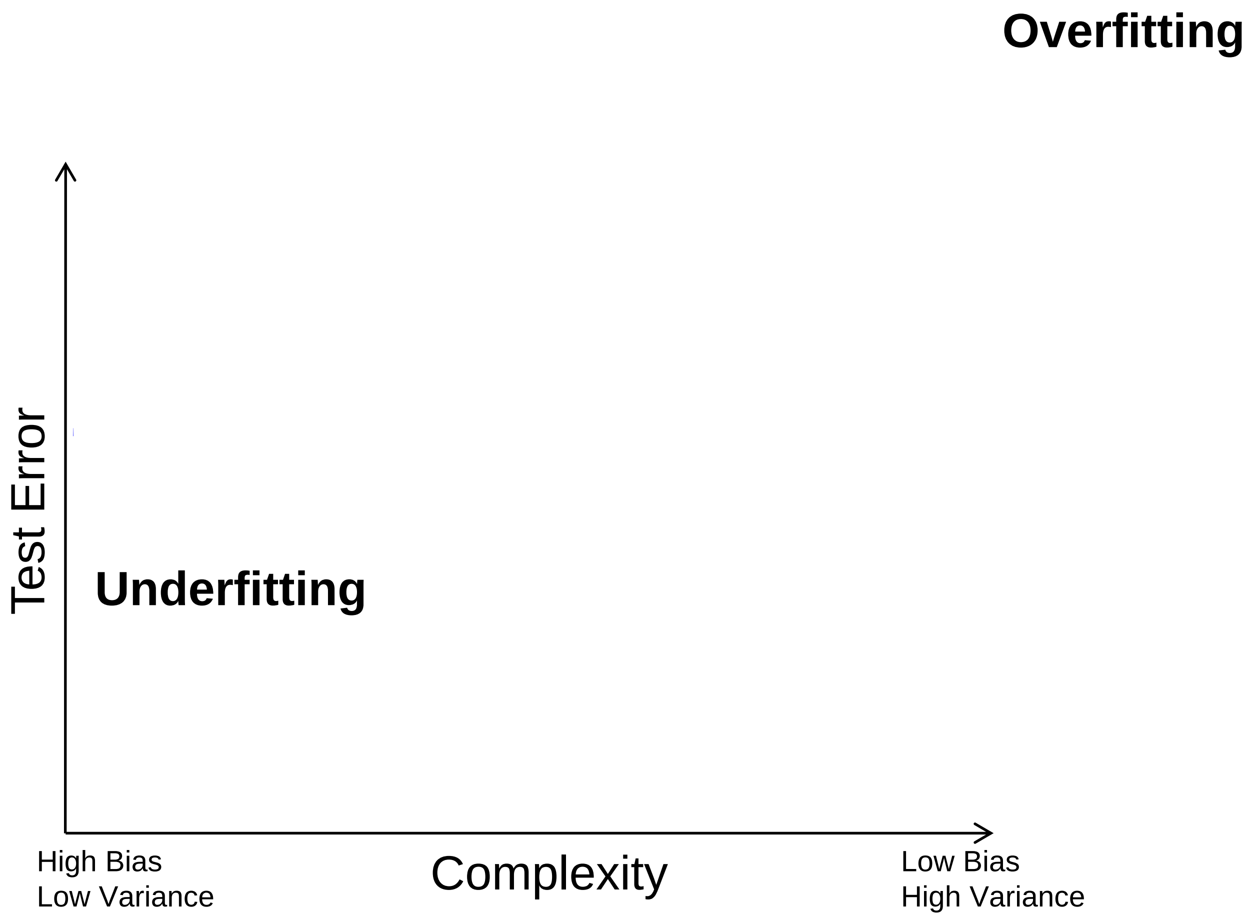
Slide credit: D. Hoiem

Bias-variance tradeoff



Slide credit: D. Hoiem

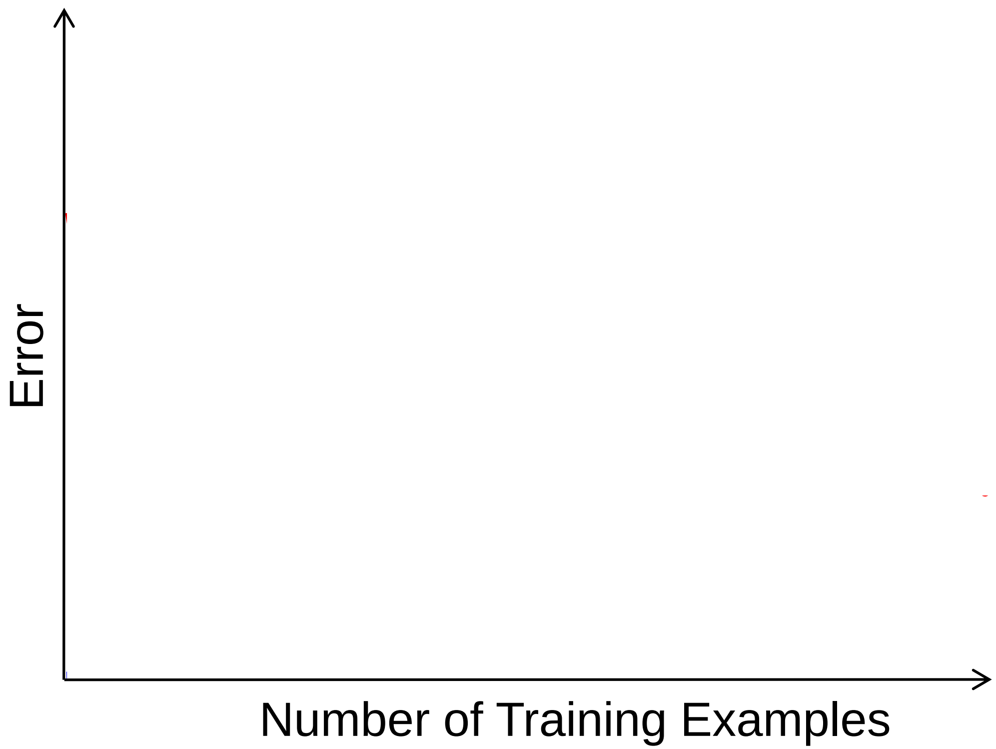
Bias-variance tradeoff



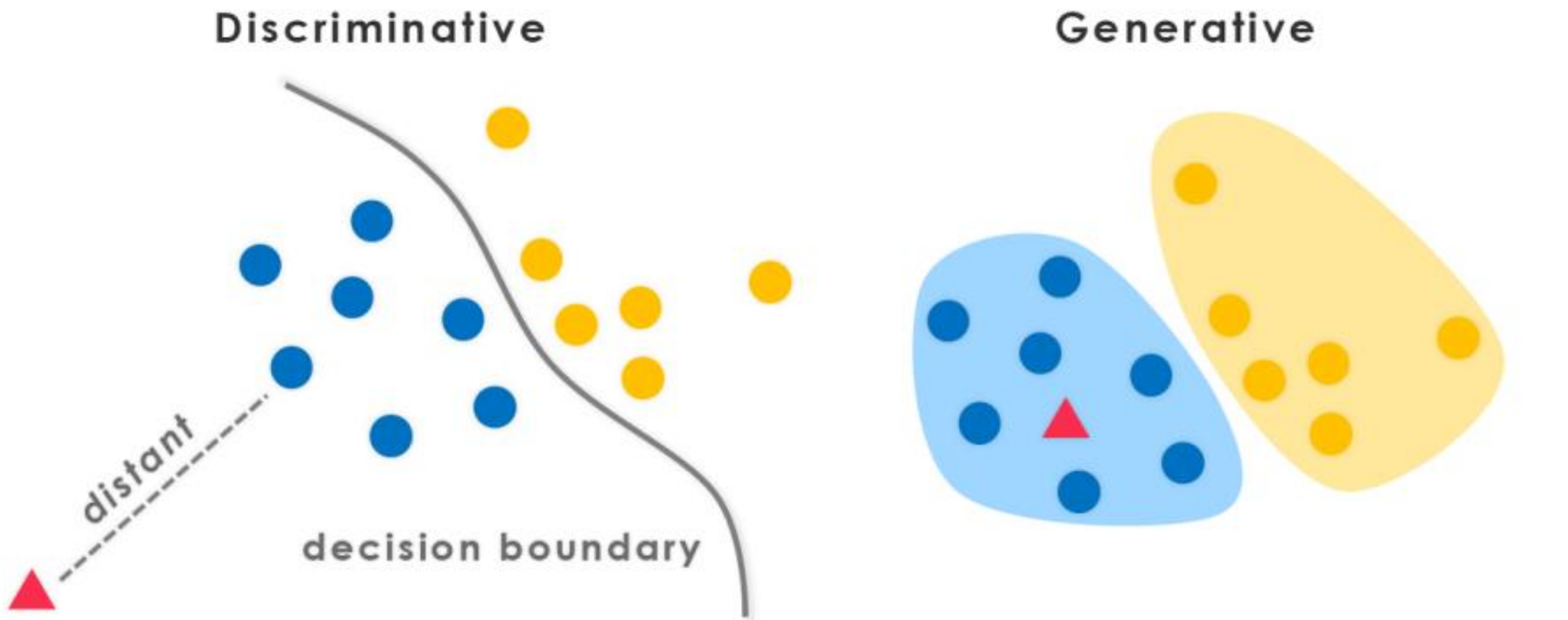
Slide credit: D. Hoiem

Effect of Training Size

Fixed prediction model



Slide credit: D. Hoiem



“Learn the data boundary” “Represent the data + boundary”

Bayesian methods:
Condition model on
data probabilistically

evolvingai.org

Photo: CMU Machine Learning Department Protests G20

Slides: James Hays, Isabelle Guyon, Erik Sudderth, Mark Johnson, Derek Hoiem

Many classifiers to choose from…

- K-nearest neighbor
- SVM
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- Restricted Boltzmann Machines
- Neural networks
- Deep Convolutional Network
- …

Which is the best?

Claim:

The decision to use machine learning is more important than the choice of a particular learning method.

*Deep learning seems to be an exception to this, currently, because it learns the feature representation.

Claim:

It is more important to have more or better labeled data than to use a different supervised learning technique.

*Again, deep learning may be an exception here for the same reason, but deep learning _needs_ a lot of labeled data in the first place.

“The Unreasonable Effectiveness of Data” - Norvig