CS 1674: Visual Recognition

PhD. Nils Murrugarra-Llerena
nem177@pitt.edu

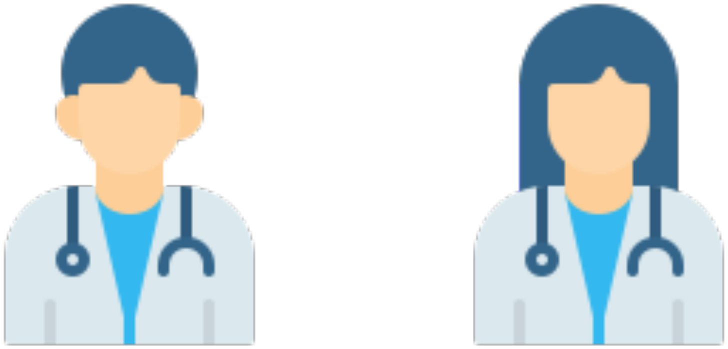
University of Pittsburgh

[Motivation] Visual Recognition

Medical Diagnostics: Aiding Pathologists

Imagine a pathologist examining a biopsy slide to detect cancer. This is a highly skilled, time-consuming, and mentally taxing task. They have to scan the entire slide under a microscope, looking for subtle cellular abnormalities that might indicate a malignant tumor. A single misdiagnosis can have devastating consequences.

Can we use CV to help pathologists in this task? How?

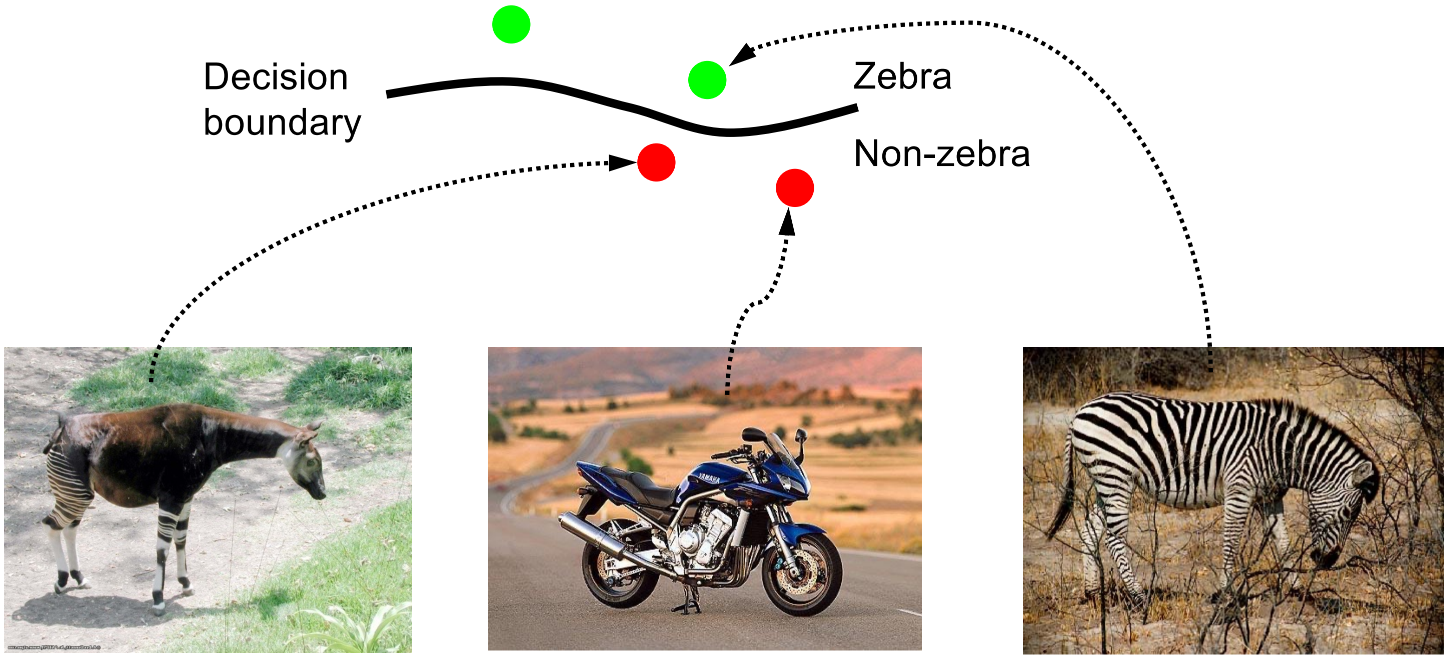


Plan for this lecture

- What is recognition?
  - a.k.a. classification, categorization
- Support vector machines
  - Separable case / non-separable case
  - Linear / non-linear (kernels)
- Example approach for scene classification
- Evaluation Metrics

Classification

- Given a feature representation for images, how do we learn a model for distinguishing features from different classes?



Decision boundary

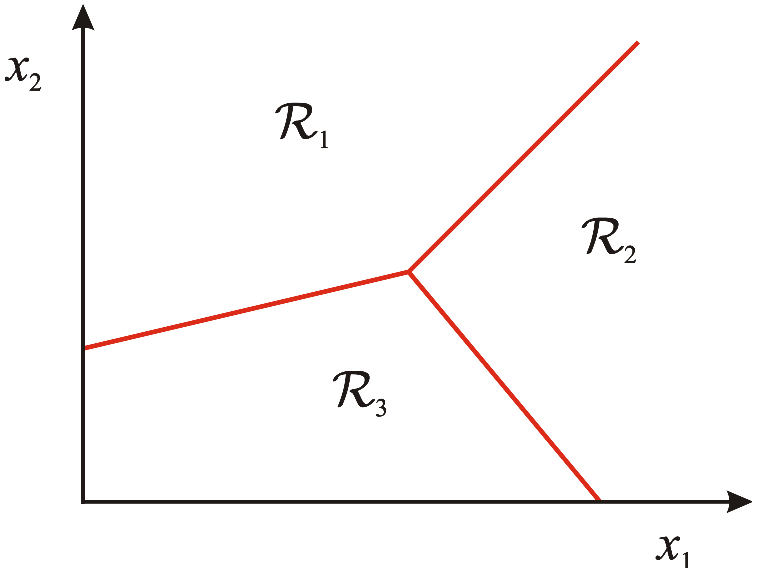
Zebra

Non-zebra

Slide credit: L. Lazebnik

Classification

- Assign input vector to one of two or more classes
- Input space divided into *decision regions* separated by *decision boundaries*



Slide credit: L. Lazebnik

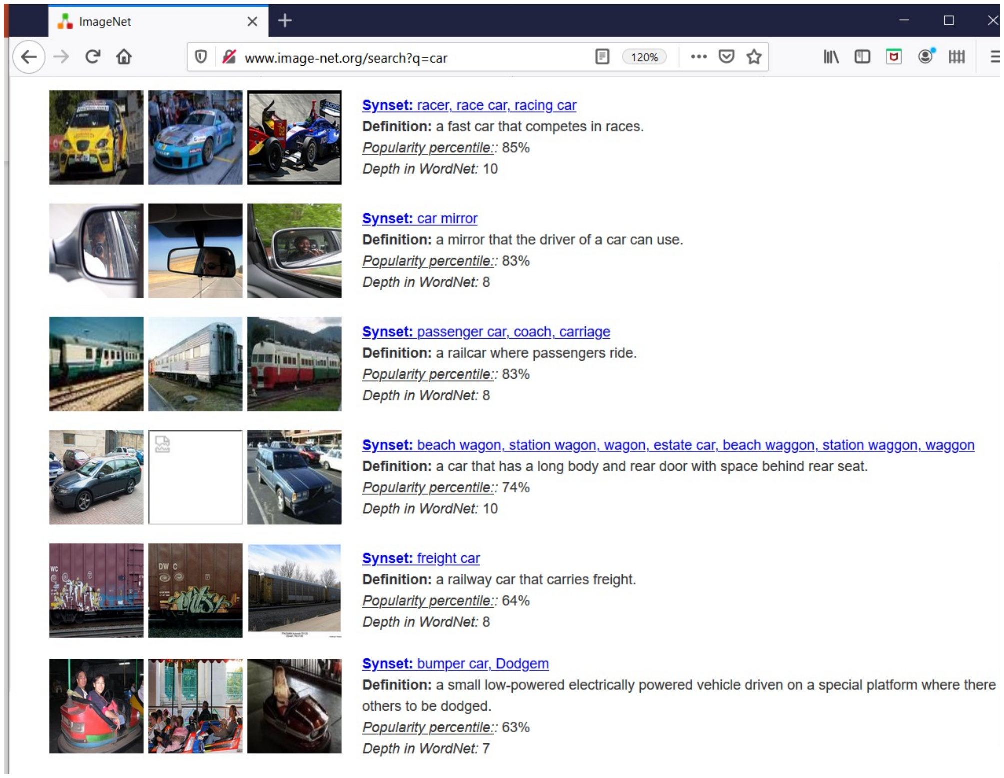
Examples of image classification

- Two-class (binary): Cat vs Dog

Adapted from D. Hoiem

Examples of image classification

- Multi-class (often): Object recognition



Synset: racer, race car, racing car
Definition: a fast car that competes in races.
Popularity percentile:: 85%
Depth in WordNet: 10

Synset: car mirror
Definition: a mirror that the driver of a car can use.
Popularity percentile:: 83%
Depth in WordNet: 8

Synset: passenger car, coach, carriage
Definition: a railcar where passengers ride.
Popularity percentile:: 83%
Depth in WordNet: 8

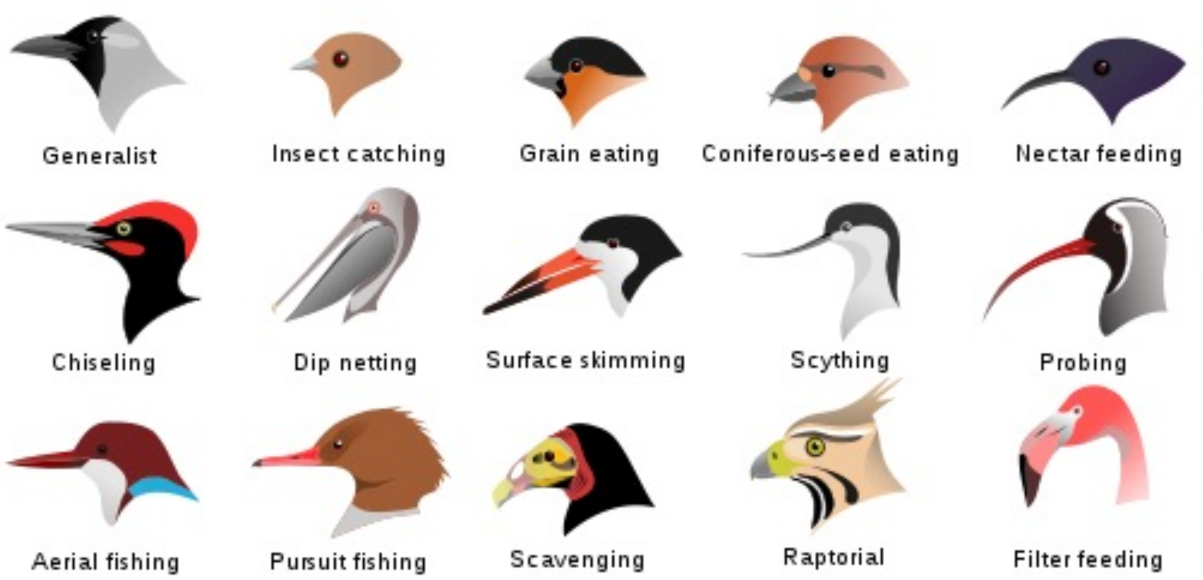
Synset: beach wagon, station wagon, wagon, estate car, beach waggon, station waggon, waggon
Definition: a car that has a long body and rear door with space behind rear seat.
Popularity percentile:: 74%
Depth in WordNet: 10

Synset: freight car
Definition: a railway car that carries freight.
Popularity percentile:: 64%
Depth in WordNet: 8

Synset: bumper car, Dodgem
Definition: a small low-powered electrically powered vehicle driven on a special platform where there others to be dodged.
Popularity percentile:: 63%
Depth in WordNet: 7

Examples of image classification

- Fine-grained recognition

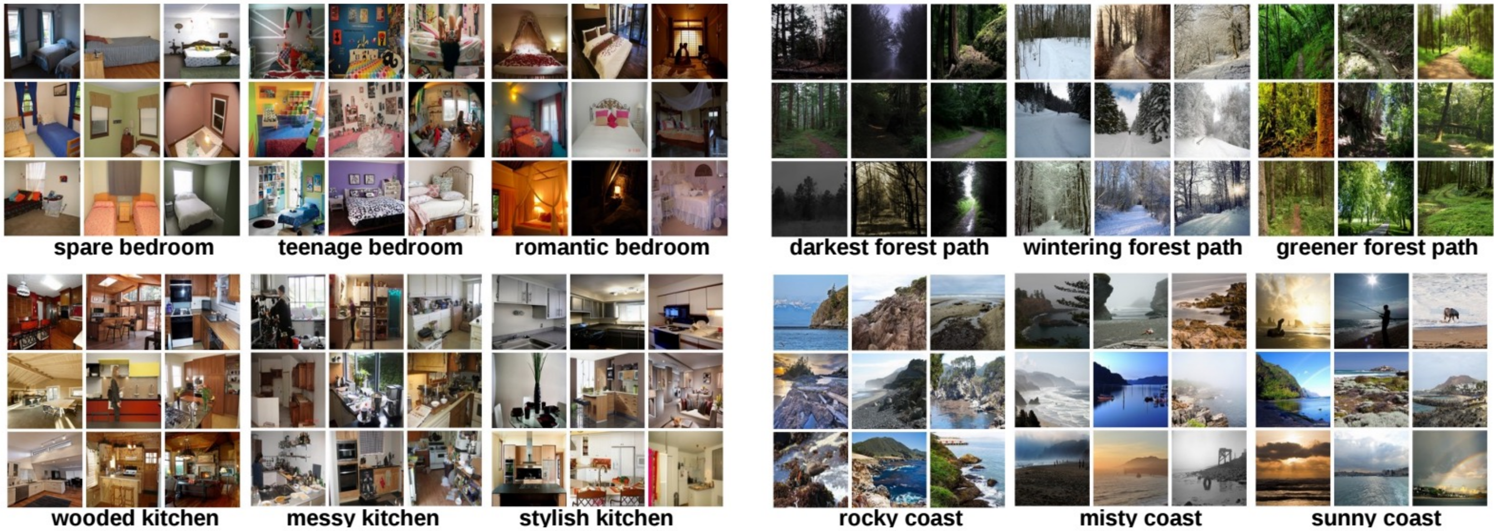


Visipedia Project

Slide credit: D. Hoiem

Examples of image classification

- Place recognition

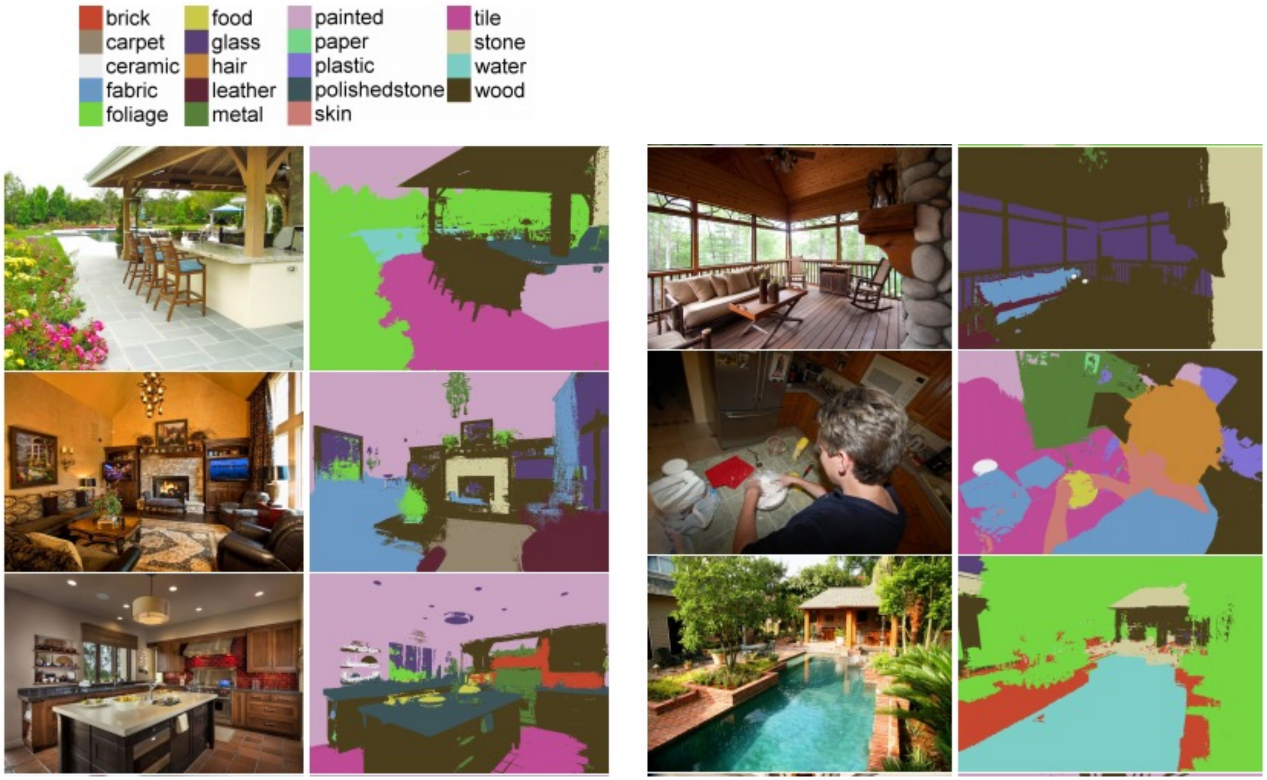


Places Database [Zhou et al. NIPS 2014]

Slide credit: D. Hoiem

Examples of image classification

- Material



[Bell et al. CVPR 2015]

Slide credit: D. Hoiem

Examples of image classification

- Dating historical photos

1940

1953

1966

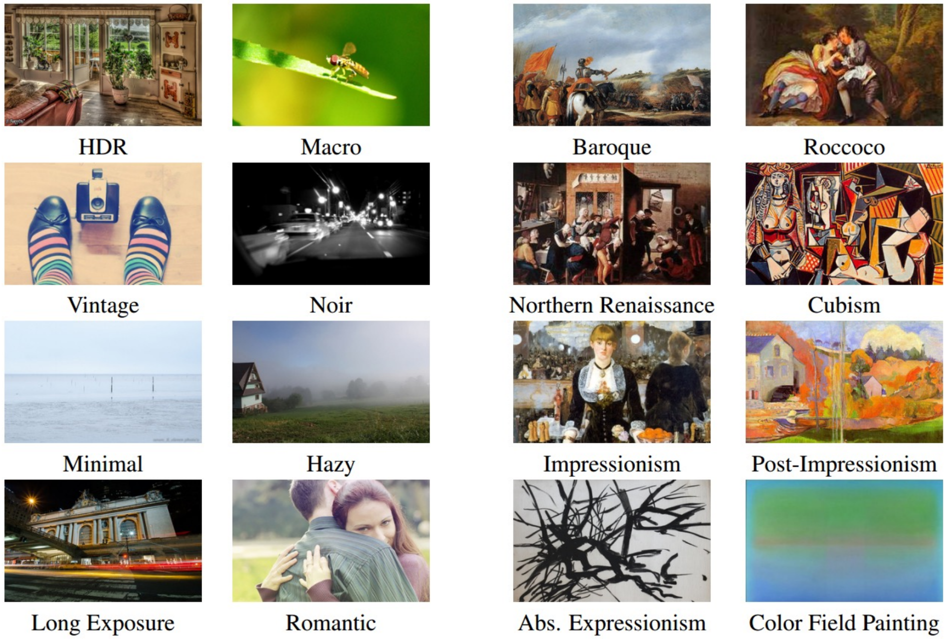
1977

[Palermo et al. ECCV 2012]

Slide credit: D. Hoiem

Examples of image classification

- Image style recognition

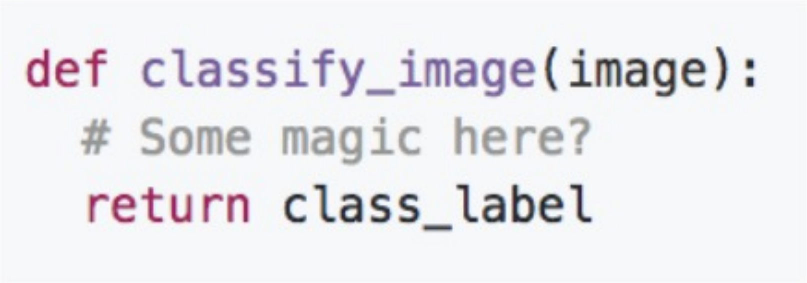


Flickr Style: 80K images covering 20 styles. Wikipaintings: 85K images for 25 art genres.

[Karayev et al. BMVC 2014]

Slide credit: D. Hoiem

Recognition: An Image Classifier



```python
def classify_image(image):
  # Some magic here?
  return class_label
```

Unlike e.g. sorting a list of numbers,

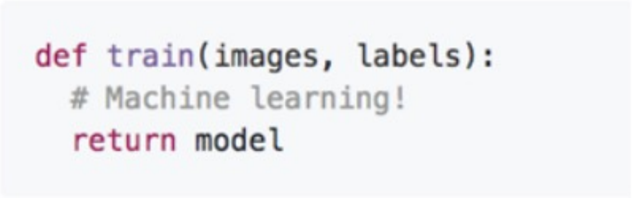
no obvious way to hard-code the algorithm for recognizing a cat, or other classes.

Slide Credit: https://cs231n.stanford.edu/

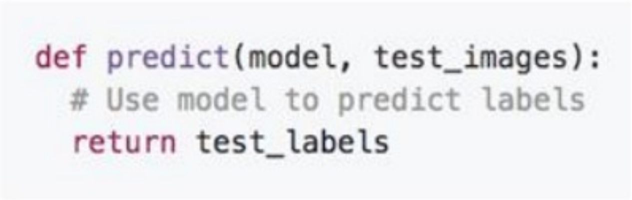
Recognition: A machine learning approach

Recognition: A machine learning approach

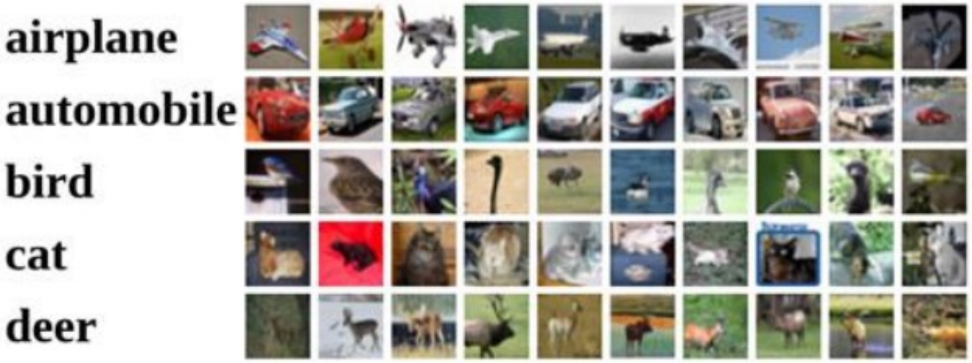
1. Collect a dataset of images and labels
2. Use Machine Learning algorithms to train a classifier
3. Evaluate the classifier on new images



```
def train(images, labels):
  # Machine learning!
  return model
```



```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```



Example training set

Slide Credit: https://cs231n.stanford.edu/

The machine learning framework

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

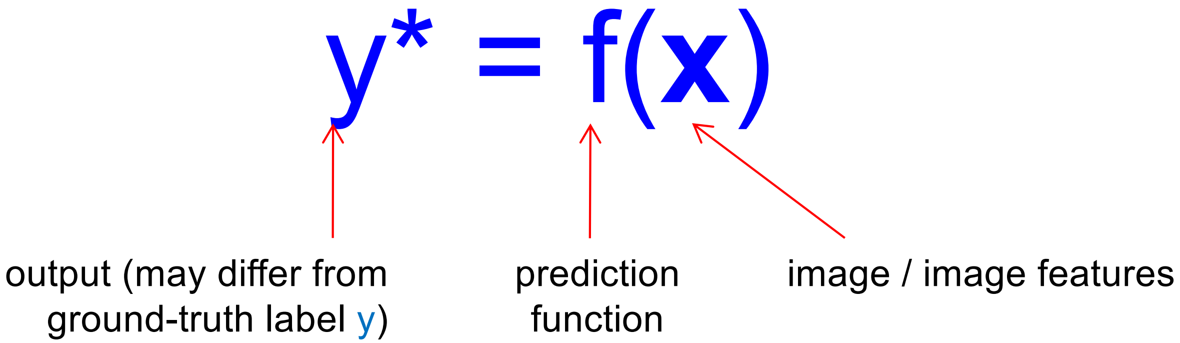
f( ) = “apple”

f( ) = “tomato”

f( ) = “cow”

Slide credit: L. Lazebnik

The machine learning framework



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

output (may differ from ground-truth label y)

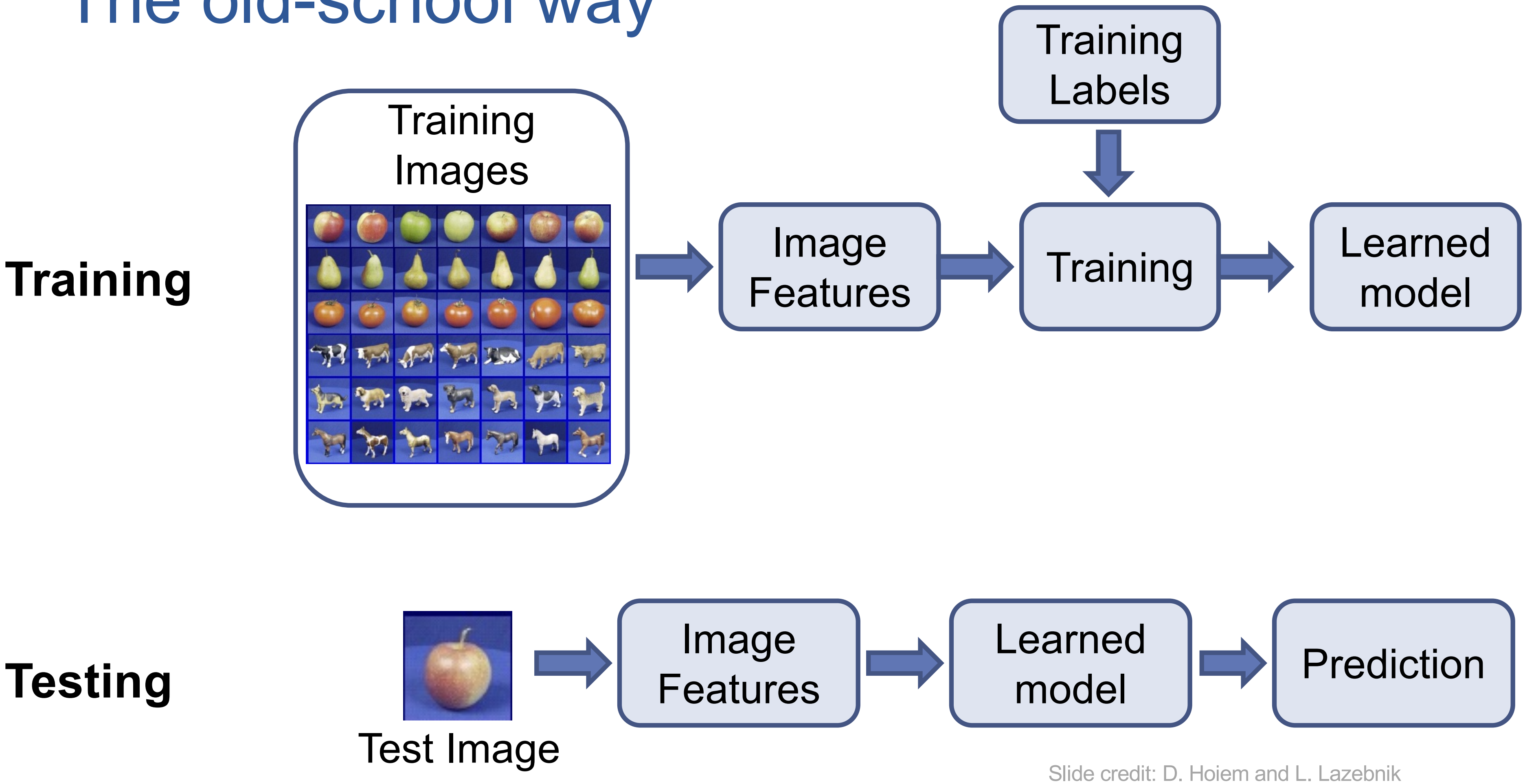
prediction function

image / image features

- **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, e.g. $|f(\mathbf{x}_i) - y_i|$
  - Evaluate multiple hypotheses f_1, f_2, f_H … and pick the best one as f
- **Testing:** apply f to a never-before-seen *test example* $\mathbf{x}$ and output the predicted value $y^* = f(\mathbf{x})$

Slide credit: L. Lazebnik

The old-school way



Training

Training Images

Image Features

Training Labels

Training

Learned model

Testing

Test Image

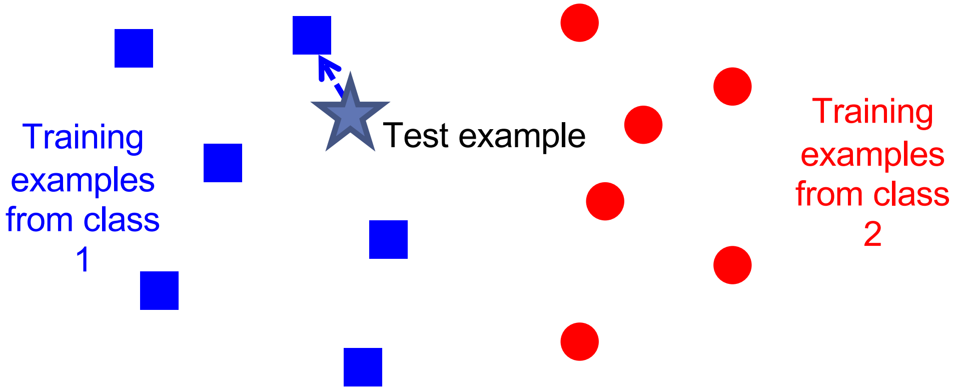
Image Features

Learned model

Prediction

Slide credit: D. Hoiem and L. Lazebnik

The simplest classifier: Nearest Neighbor Classifier


Training examples from class 1
Test example
Training examples from class 2

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

- All we need is a distance function for our inputs
- No training required!

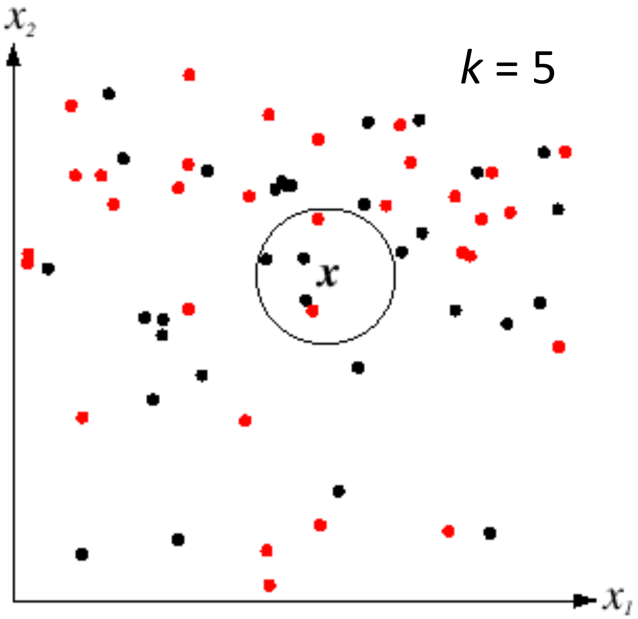
Slide credit: L. Lazebnik

K-Nearest Neighbors classification

- For a new point, find the *k* closest points from training data
- Labels of the *k* points “vote” to classify

Black = positive

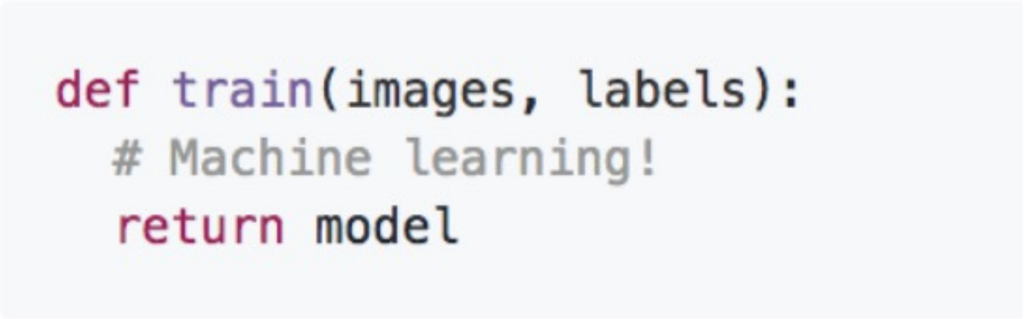
Red = negative



If query lands here, the 5 NN consist of 3 positives and 2 negatives, so we classify it as positive.

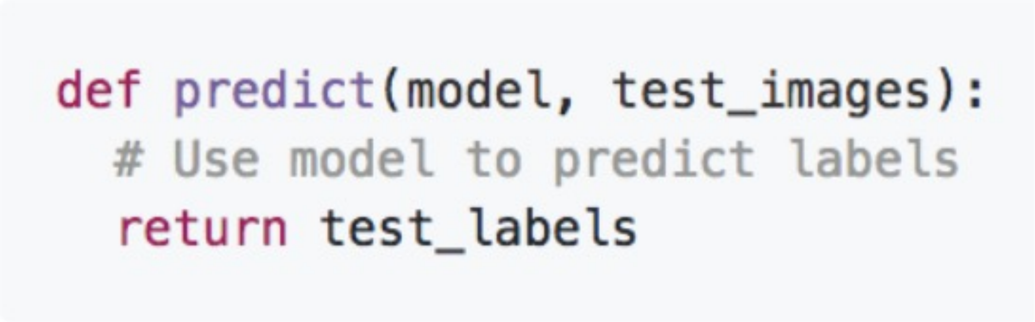
Slide credit: D. Lowe

K-Nearest Neighbor - Summary



```
def train(images, labels):
  # Machine learning!
  return model
```

Memorize all data and labels



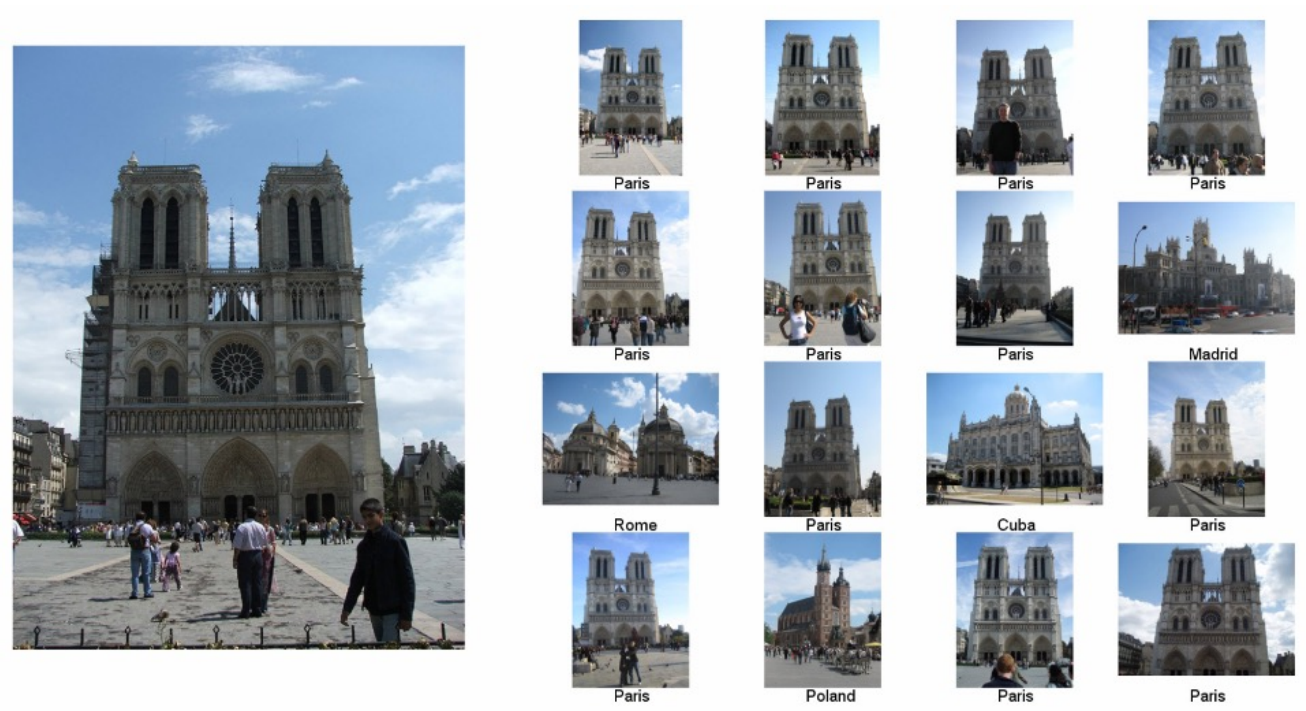
```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

Predict the label of the most similar training image

Slide Credit: https://cs231n.stanford.edu/

[Application]Im2gps: Estimating Geographic Information from a Single Image [James Hays and Alexei Efros, CVPR 2008]

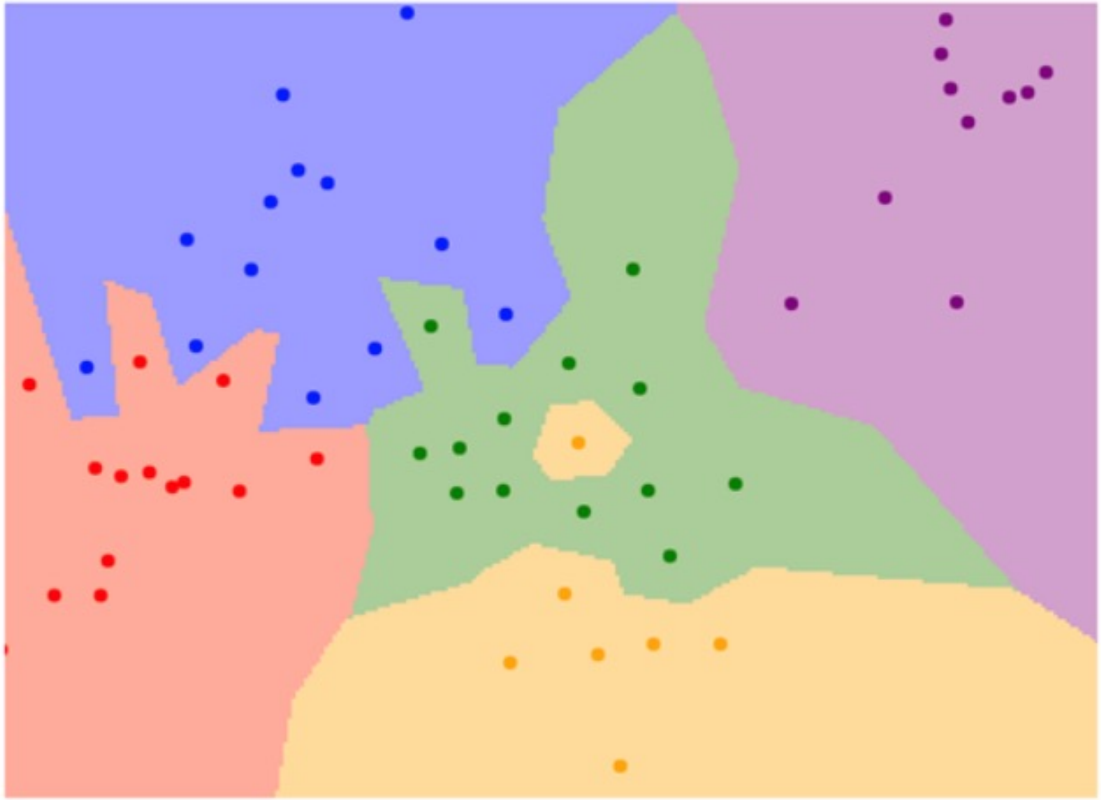
Where was this image taken?



Nearest Neighbors according to BOW-SIFT + color histogram + a few others

Slide credit: James Hays

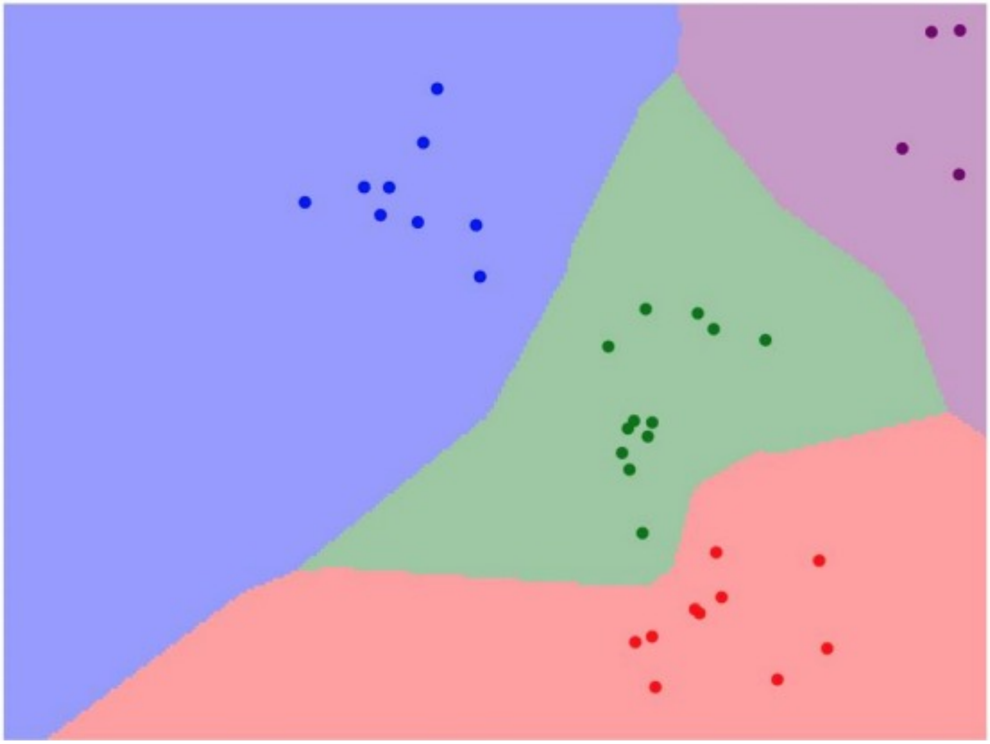
K-Nearest Neighbor - Visualization



1-nearest Neighbor

Slide Credit: https://cs231n.stanford.edu/

K-Nearest Neighbor – Interact – Try it yourself

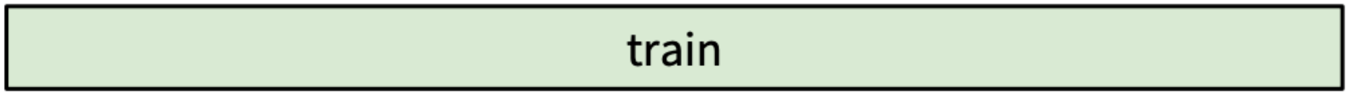


http://vision.stanford.edu/teaching/cs231n-demos/knn/

Slide Credit: https://cs231n.stanford.edu/

Setting Hyperparameters: Best value of k?

Idea #1: Choose hyperparameters that work best on the training data



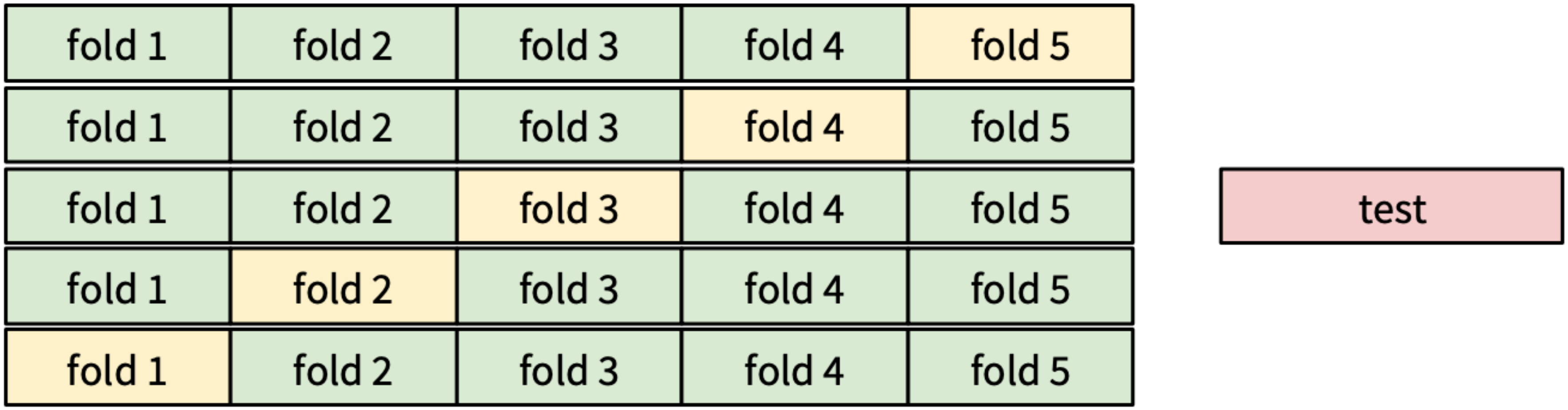
train

Slide Credit: https://cs231n.stanford.edu/

Setting Hyperparameters: Best value of k?

train

Idea #4: Cross-Validation: Split data into folds, try each fold as validation and average the results



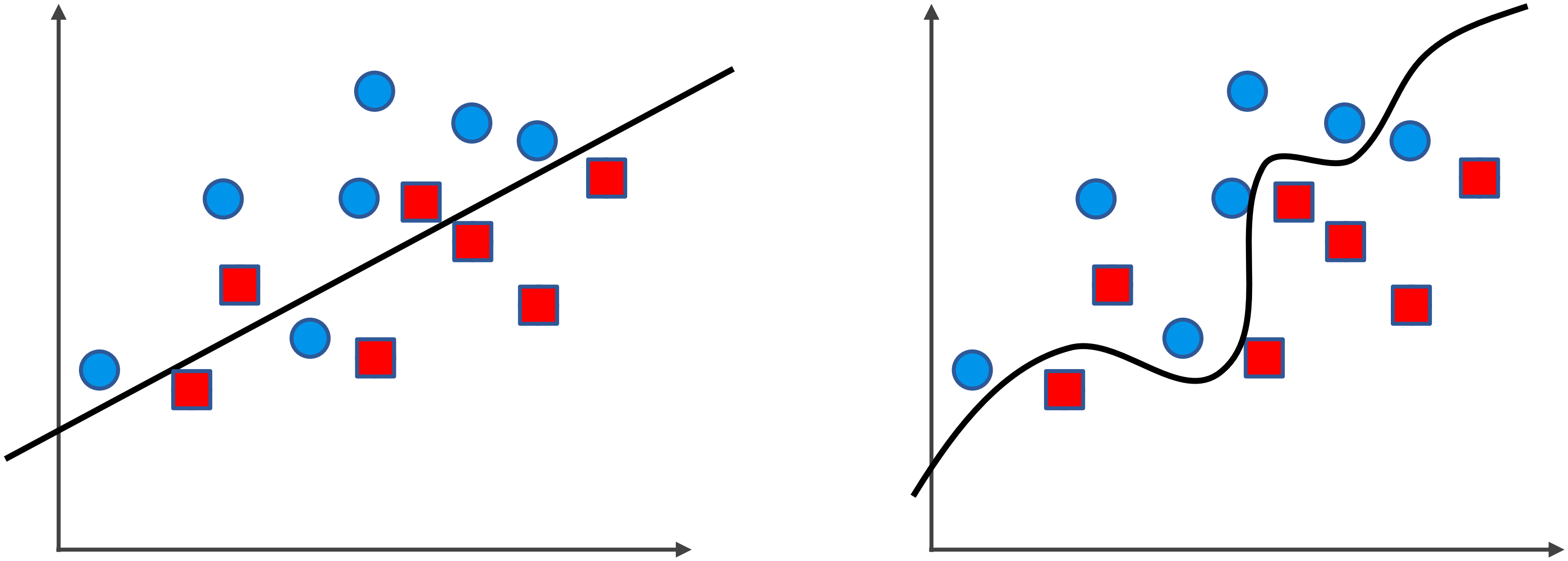
fold 1	fold 2	fold 3	fold 4	fold 5
fold 1	fold 2	fold 3	fold 4	fold 5
fold 1	fold 2	fold 3	fold 4	fold 5
fold 1	fold 2	fold 3	fold 4	fold 5
fold 1	fold 2	fold 3	fold 4	fold 5

test

Useful for small datasets, but not used too frequently in deep learning

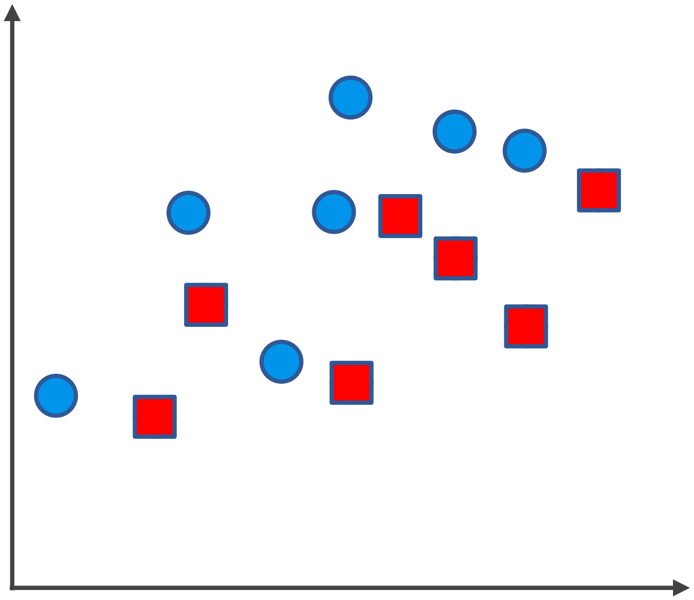
Slide Credit: https://cs231n.stanford.edu/

Which model is better? Why validating?



Slide Credit: Prof. Sandra Avila - UNICAMP

Why validating?

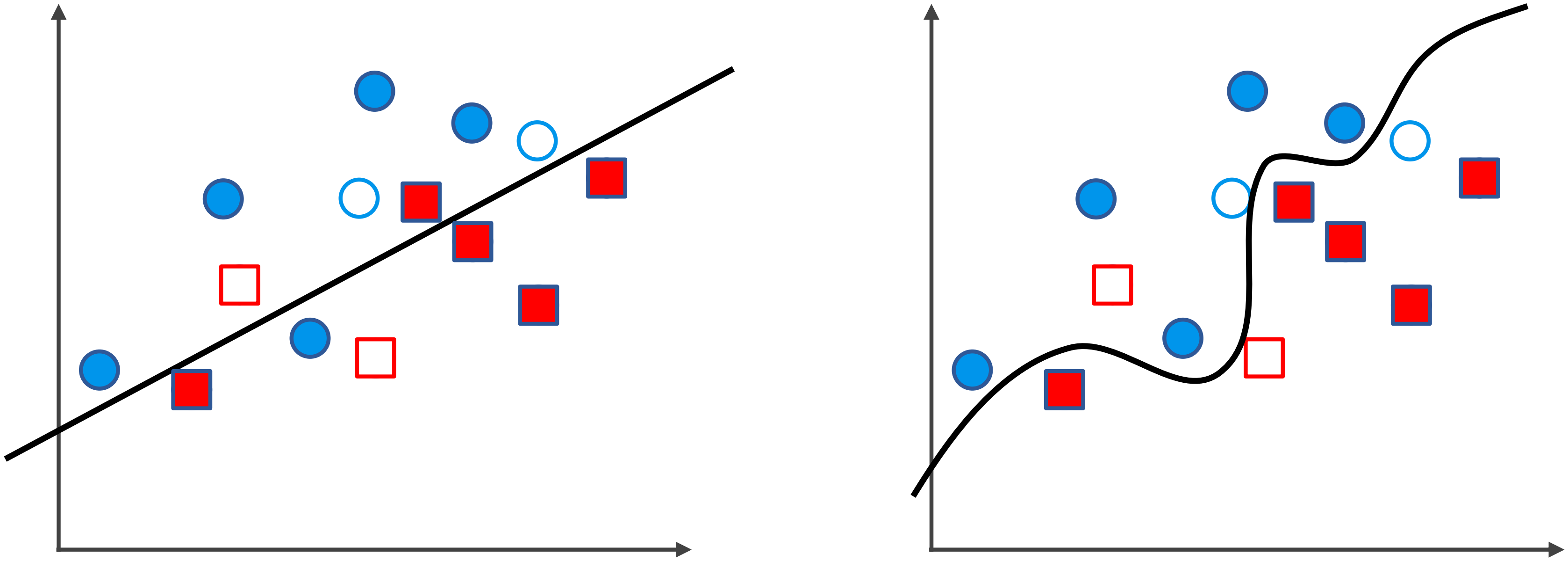


Slide Credit: Prof. Sandra Avila - UNICAMP

● ■ Training

○ □ Validation

Why validating?

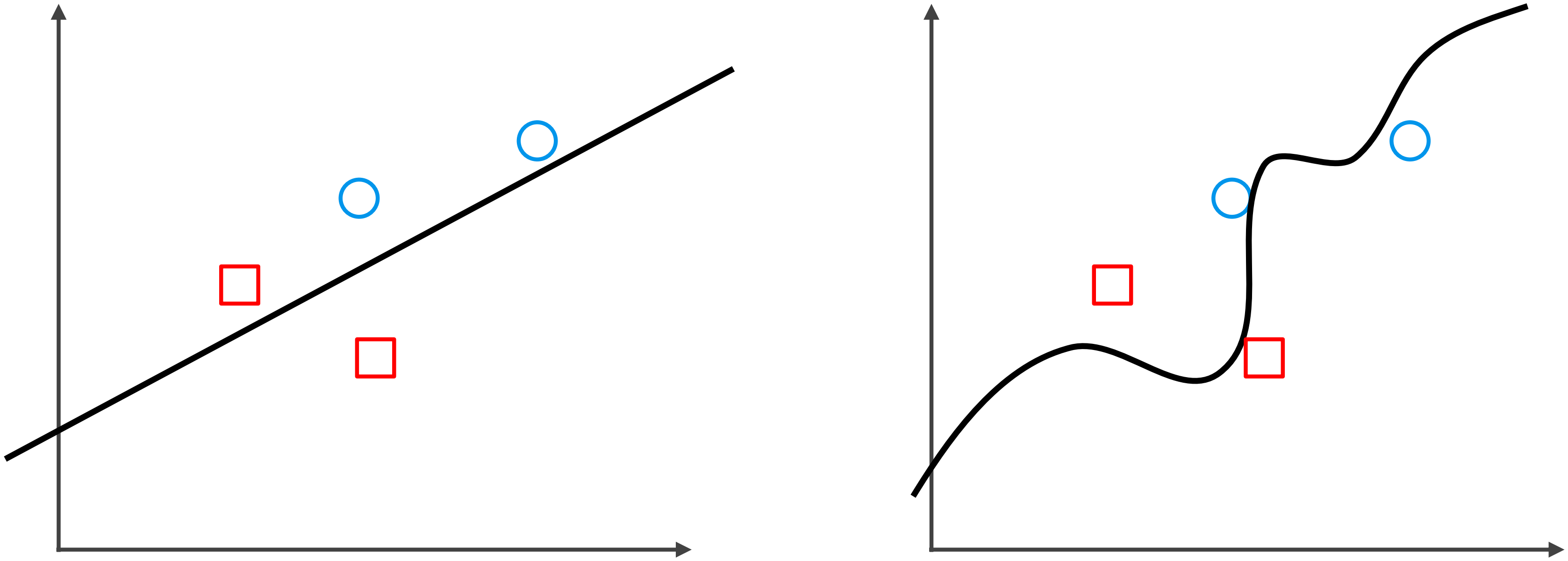


Slide Credit: Prof. Sandra Avila - UNICAMP

● ■ Training

○ □ Validation

Why validating?

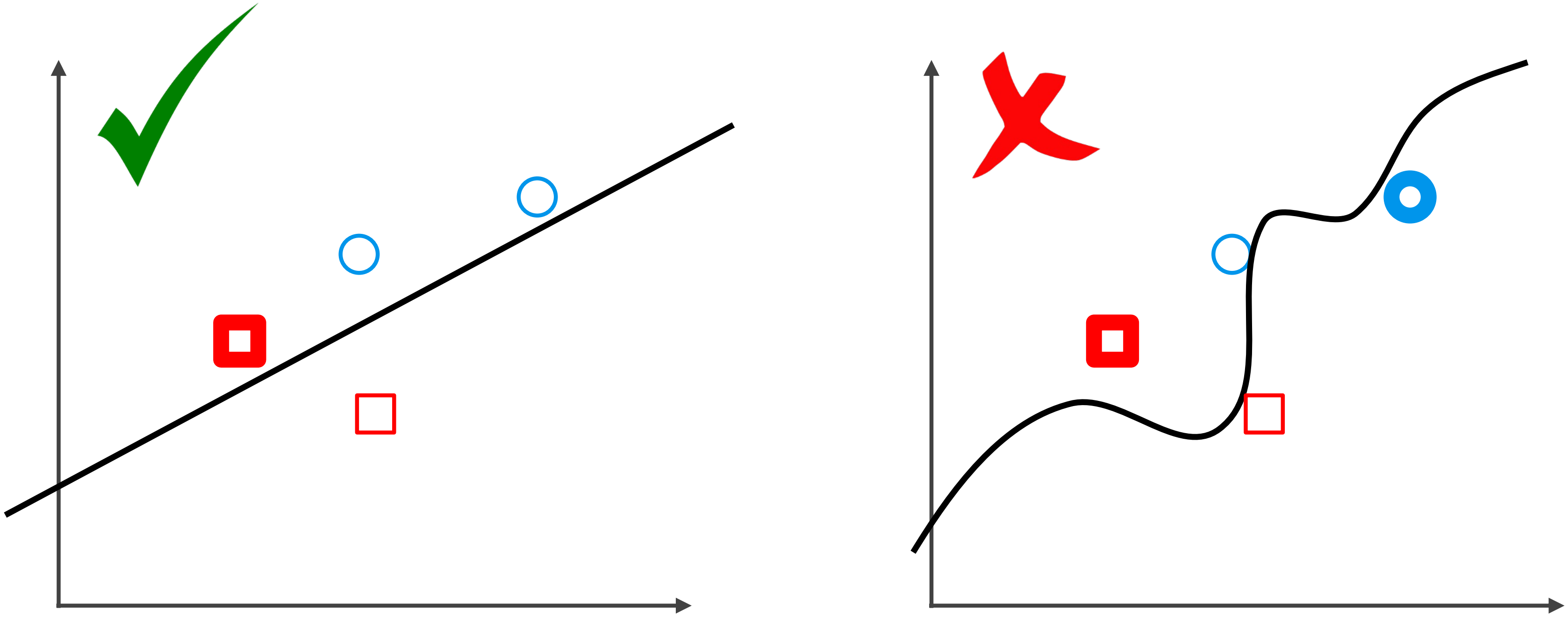


Slide Credit: Prof. Sandra Avila - UNICAMP

Why validating?

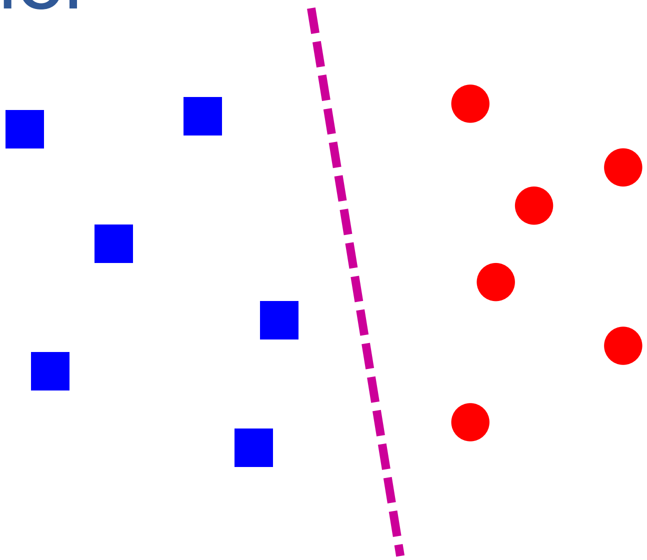
●■ Training

○□ Validation



Slide Credit: Prof. Sandra Avila - UNICAMP

Linear classifier



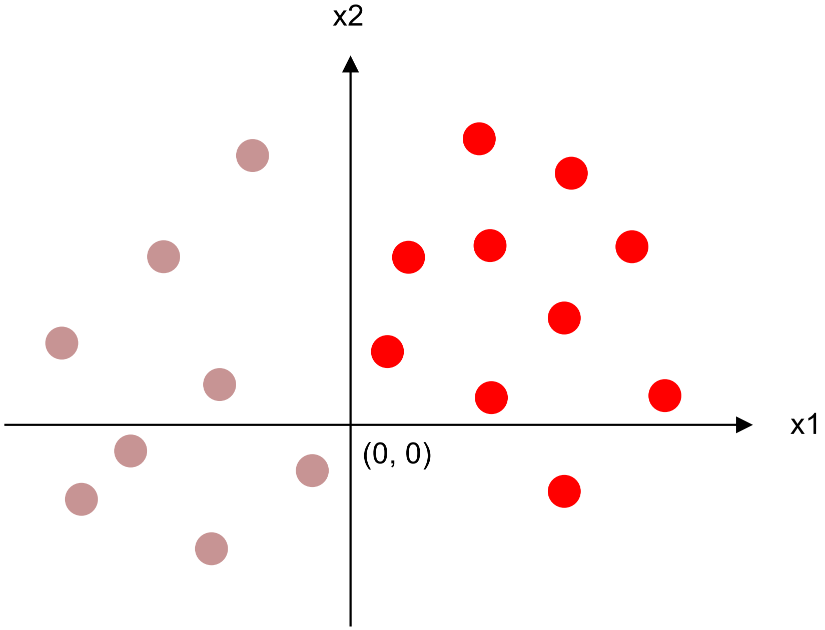
- Find a *linear function* to separate the classes

$$f(\mathbf{x}) = \text{sgn}(w_1x_1 + w_2x_2 + \ldots + w_Dx_D) = \text{sgn}(\mathbf{w} \cdot \mathbf{x})$$

Slide credit: L. Lazebnik

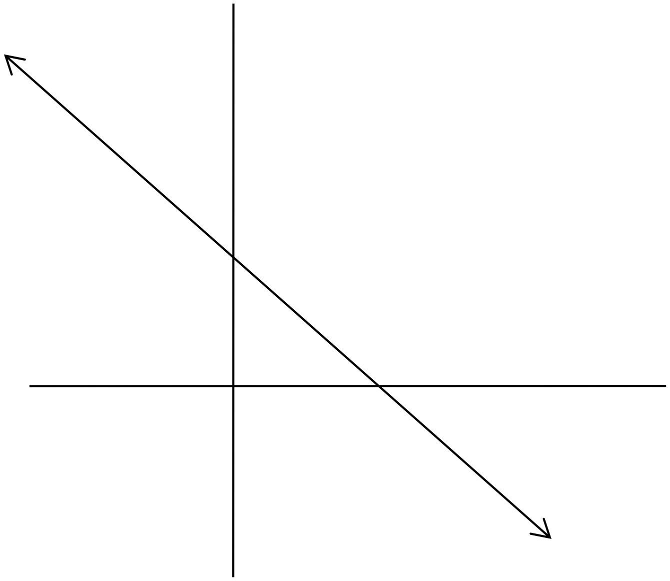
Linear Classifier

- Decision = sign($\mathbf{w}^T\mathbf{x}$) = sign(w1*x1 + w2*x2)



- What should the weights be?

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

Compare to:
*slope**x + *y-intercept* = y

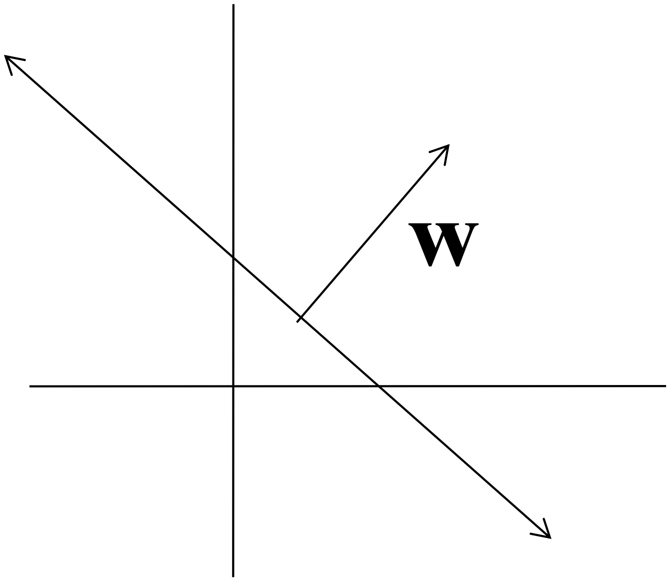
ax + b = -cy
(-a/c) x + (-b/c) = y

Slope: -a/c
Y-intercept: -b/c

Kristen Grauman

Lines in R^2

Slope: -a/c
Y-intercept: -b/c



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

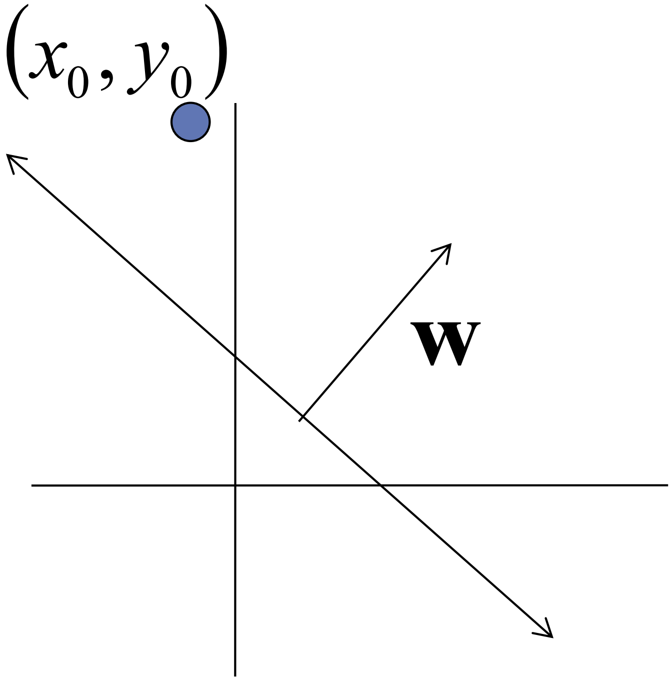
$$ax + cy + b = 0$$

$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Kristen Grauman

Lines in R^2



Slope: -a/c
Y-intercept: -b/c

Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

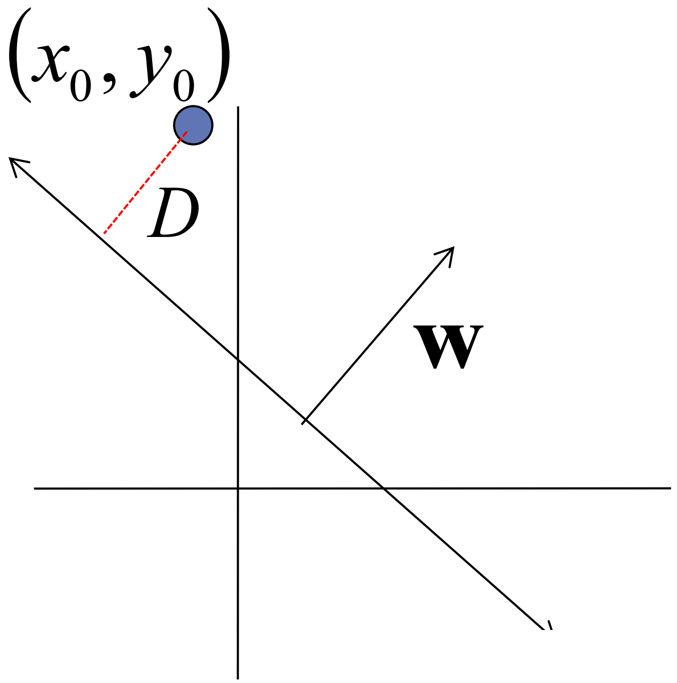
$$ax + cy + b = 0$$

$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Kristen Grauman

Lines in R^2



$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}}$$

Slope: -a/c
Y-intercept: -b/c

Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

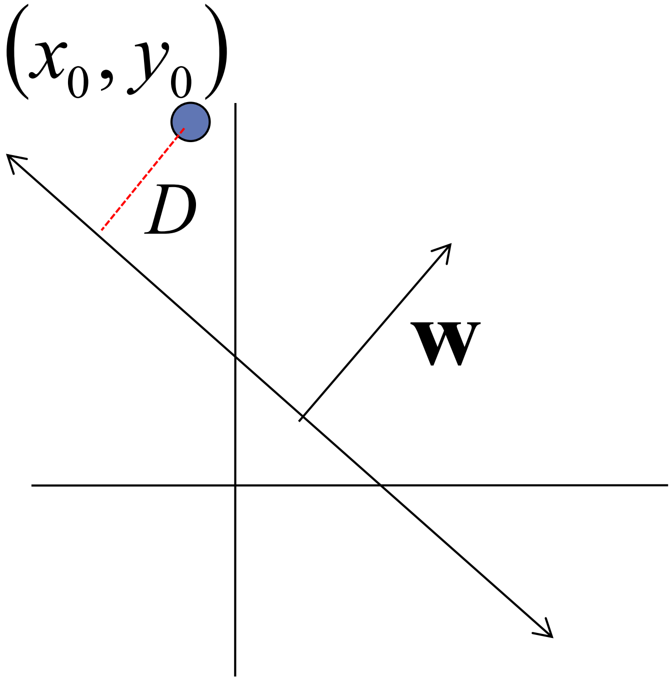
$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

$\}$ distance from point to line

Kristen Grauman

Lines in R^2



Slope: -a/c
Y-intercept: -b/c

Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

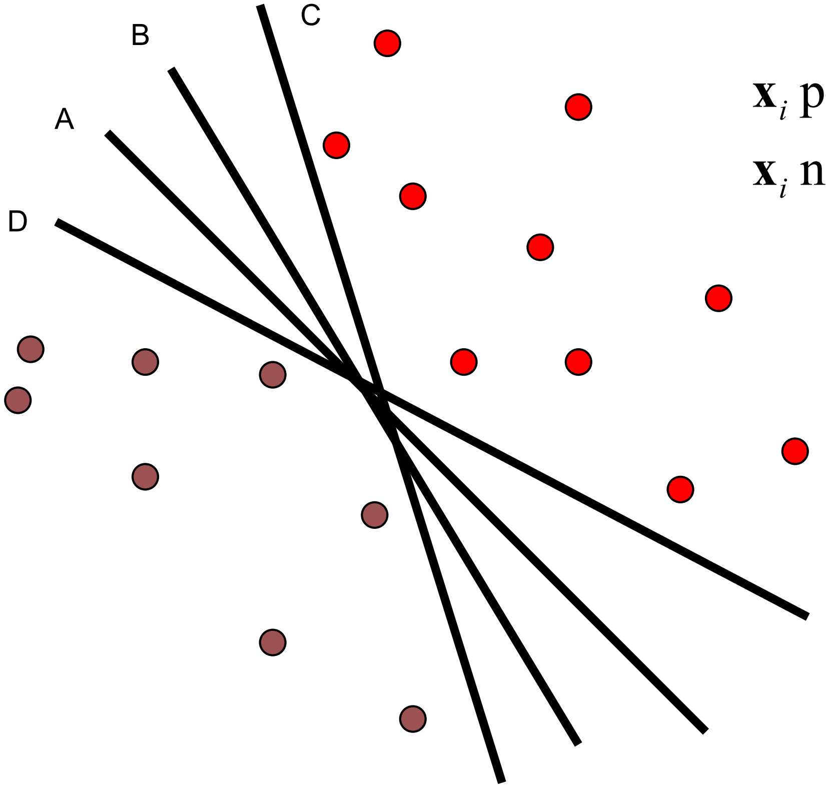
$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}} = \frac{|\,\mathbf{w}^{\mathrm{T}}\mathbf{x} + b\,|}{\|\mathbf{w}\|}$$

distance from point to line

Adapted from Kristen Grauman

Linear classifiers

- Find linear function to separate positive and negative examples



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

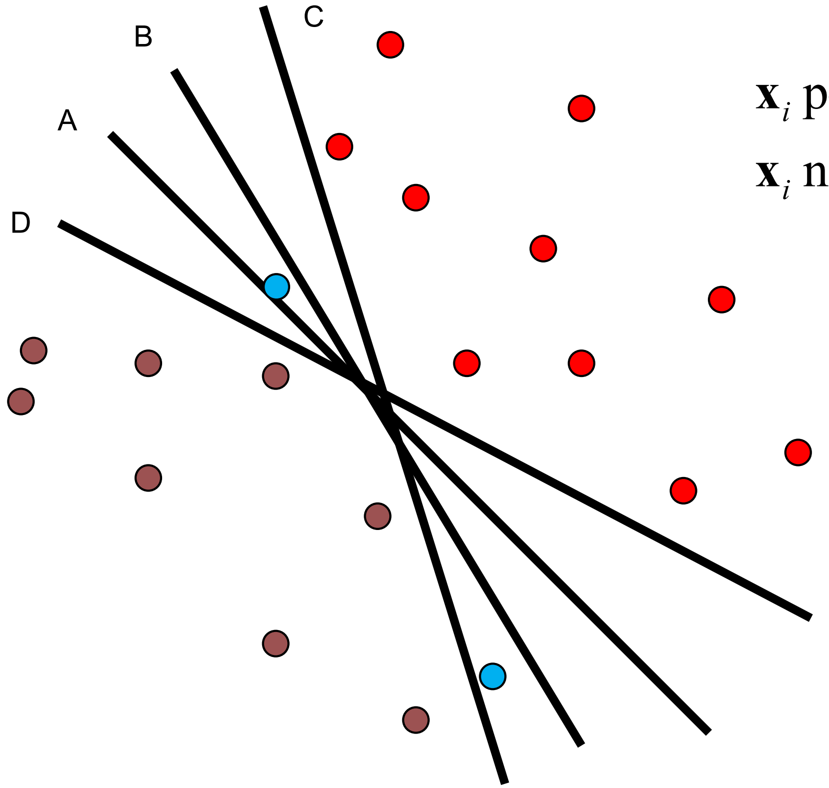
$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Which line
is best?

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Linear classifiers

- Find linear function to separate positive and negative examples



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

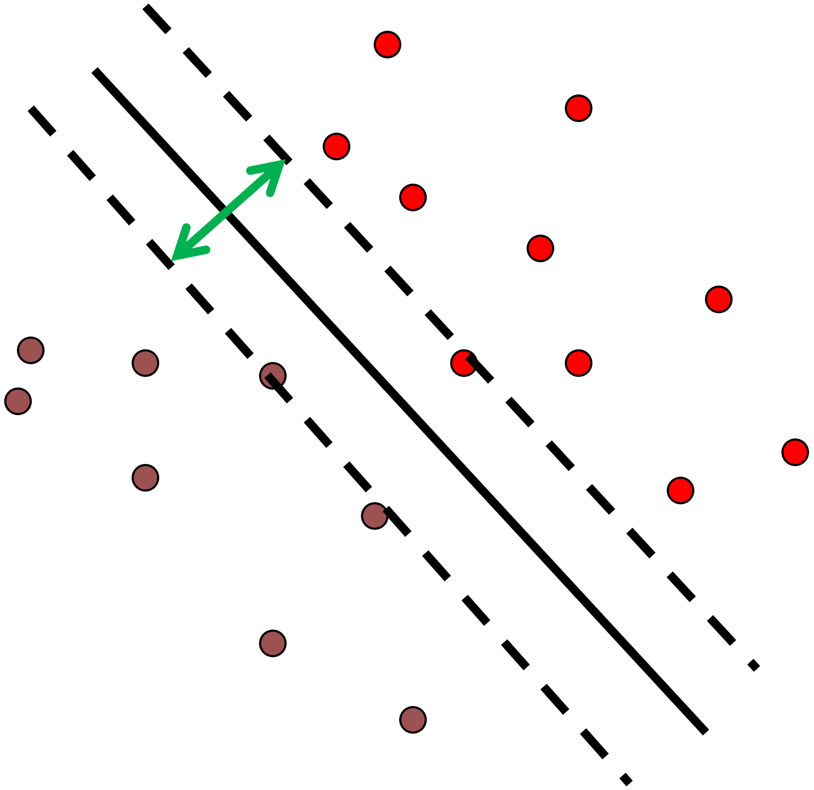
$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

● Not seen until test time, of class blue

Which line is best?

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines



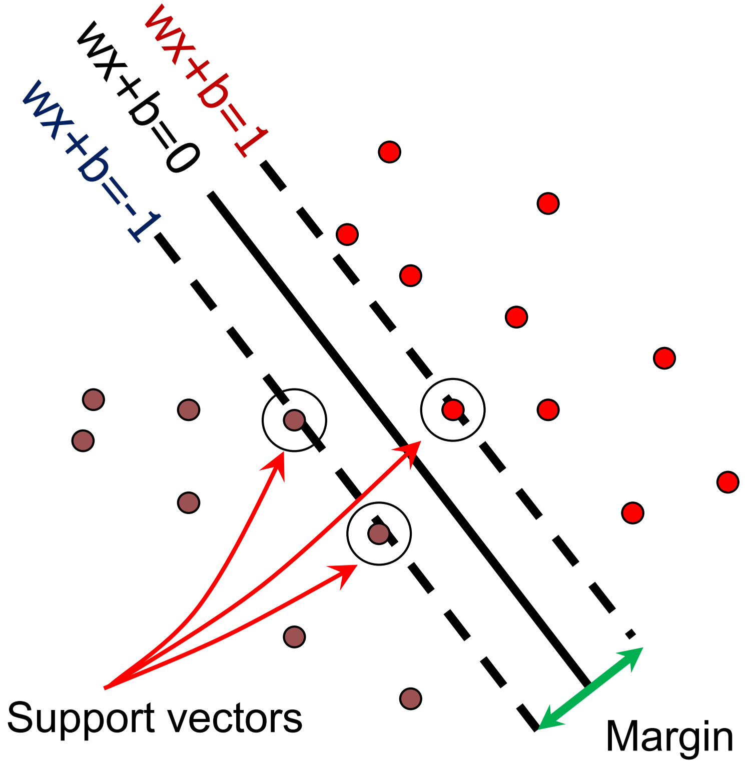
- Discriminative classifier based on *optimal separating line (for 2d case)*

- Maximize the *margin* between the positive and negative training examples

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

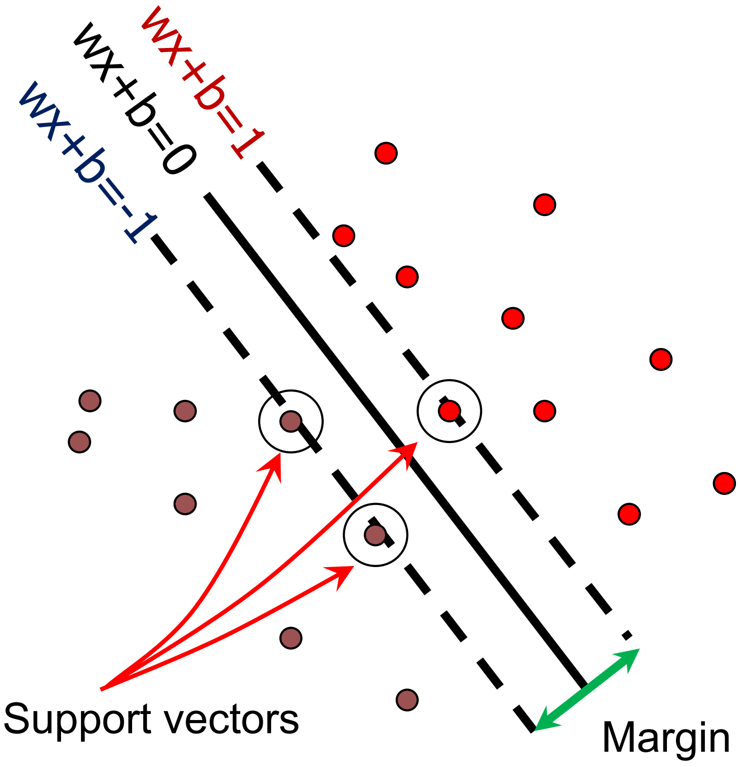
$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \ge 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \le -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\dfrac{|\,\mathbf{x}_i \cdot \mathbf{w} + b\,|}{\|\,\mathbf{w}\,\|}$

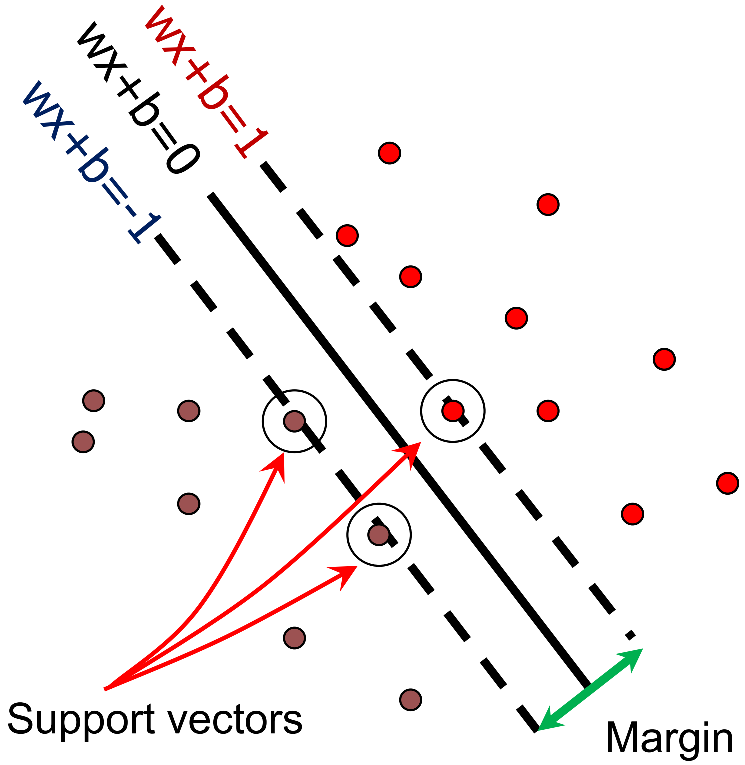
For support vectors:

$$\frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\quad \dfrac{|\,\mathbf{x}_i \cdot \mathbf{w} + b\,|}{\|\,\mathbf{w}\,\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin line

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:

$$\mathbf{x}_i \text{ positive } (y_i = 1): \qquad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \qquad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

- *Quadratic optimization problem*:

Minimize $\dfrac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $y_i(\mathbf{w} \cdot \boldsymbol{x}_i + b) \geq 1$

One constraint per training point.

Note sign trick:
$w \cdot x_i + b >= 1$ (if $y_i = 1$)

$w \cdot x_i + b <= -1$ (if $y_i = -1$)
$(-1)\ w \cdot x_i - b >= 1$

Adapted from C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

Learned weight

Support vector

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)
- Classification function:

$$f(x) = \text{sign}\,(\mathbf{w} \cdot \mathbf{x} + \text{b})$$
$$= \text{sign}\left(\sum_{\text{i}} \alpha_{\text{i}} y_i \boxed{\mathbf{x}_i \cdot \mathbf{x}} + b\right)$$

If f(x) < 0, classify as negative, otherwise classify as positive.

- Notice that it relies on an *inner product* between the test point ***x*** and the support vectors $\boldsymbol{x}_i$
- (Solving the optimization problem also involves computing the inner products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ between all pairs of training points)

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Inner product

- The decision boundary for the SVM and its optimization depend on the inner product of two data points (vectors):

$$\mathbf{x}_i^{T}\mathbf{x}_j$$

$$f(x) = \mathrm{sign}\,(\mathbf{w}\cdot\mathbf{x} + \mathrm{b})$$
$$= \mathrm{sign}\left(\sum_{\mathrm{i}} \alpha_{\mathrm{i}} y_i \boxed{\mathbf{x}_i \cdot \mathbf{x}} + b\right)$$

- The inner product is equal

$$(\mathbf{x}_i^{T}\mathbf{x}) = \|\mathbf{x}_i\| * \|\mathbf{x}_i\| \cos\,\theta$$

If the angle in between them is 0 then: $(\mathbf{x}_i^{T}\mathbf{x}) = \|\mathbf{x}_i\| * \|\mathbf{x}_i\|$

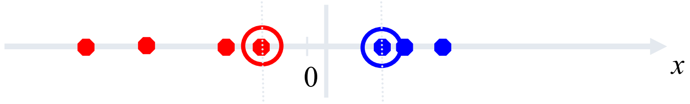
If the angle between them is 90 then: $(\mathbf{x}_i^{T}\mathbf{x}) = 0$

The inner product measures how similar the two vectors are

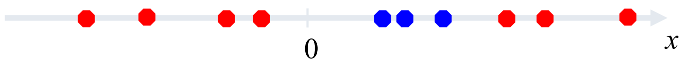
Adapted from Milos Hauskrecht

Nonlinear SVMs

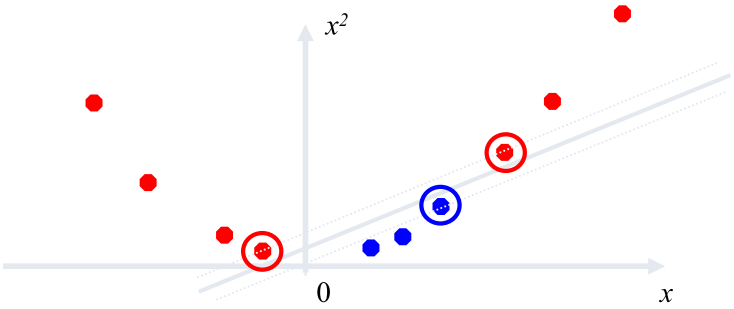
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?

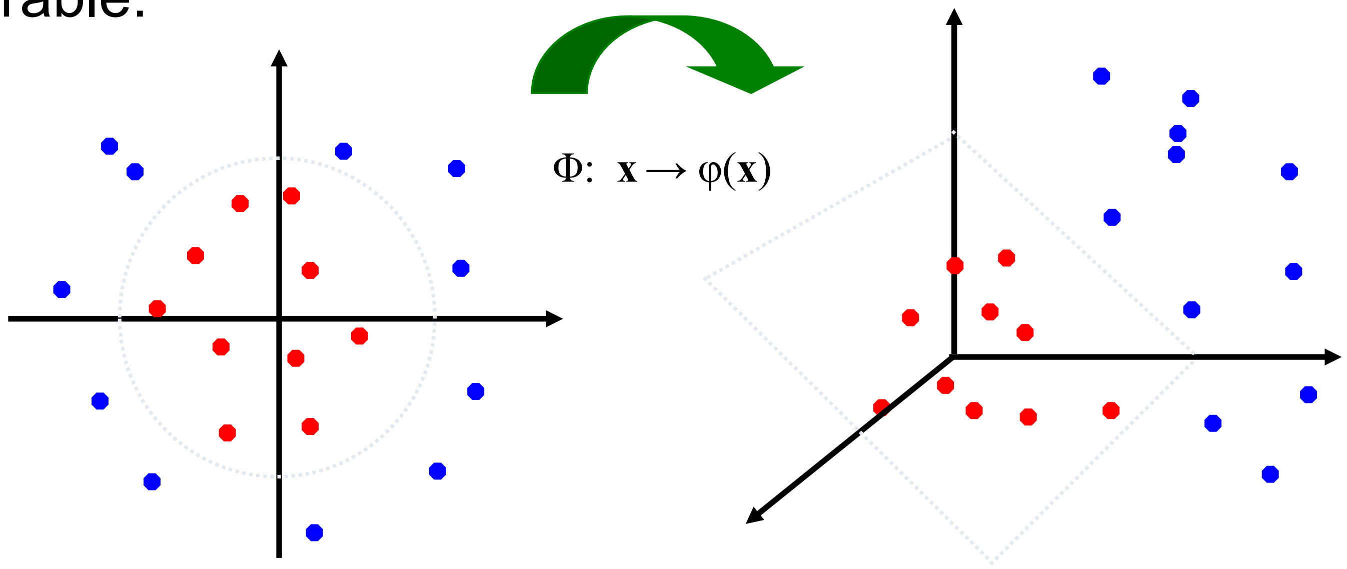


- We can map it to a higher-dimensional space:



Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

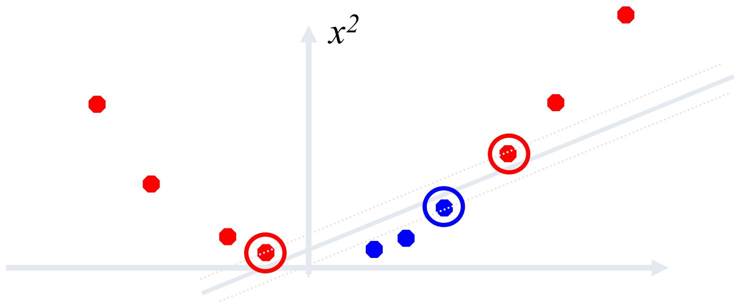


$\Phi: \; \mathbf{x} \rightarrow \varphi(\mathbf{x})$

Andrew Moore

Nonlinear kernel: Example

- Consider the mapping $\varphi(x) = (x, x^2)$



$$\varphi(x) \cdot \varphi(y) = (x, x^2) \cdot (y, y^2) = xy + x^2 y^2$$

$$K(x, y) = xy + x^2 y^2$$

Svetlana Lazebnik

The "Kernel Trick"

- The linear classifier relies on dot product between vectors $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$
- If every data point is mapped into high-dimensional space via some transformation *Φ*: $\mathbf{x}_i \rightarrow \varphi(\mathbf{x}_i)$, the dot product becomes: $K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$
- A *kernel function* is similarity function that corresponds to an inner product in some expanded feature space
- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(\mathbf{x})$, define a kernel function K such that: $K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$

Andrew Moore

Examples of kernel functions

- Linear: $K(x_i, x_j) = x_i^T x_j$
- Polynomials of degree up to *d*:

$$K(x_i, x_j) = (x_i^T x_j + 1)^d$$

- Gaussian RBF:

$$K(x_i, x_j) = \exp(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2})$$

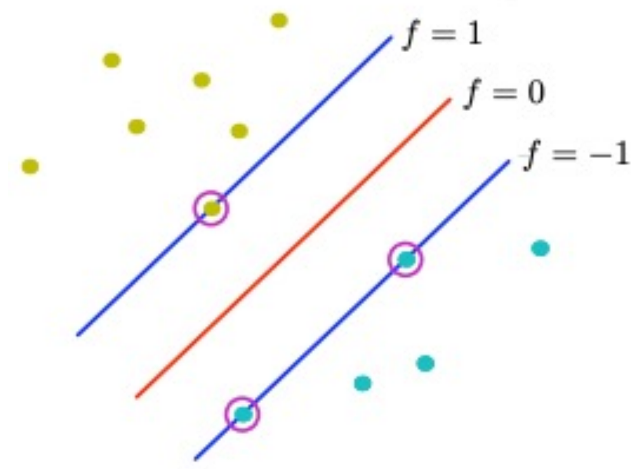
- Histogram intersection:

$$K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))$$

Hard-margin SVMs

$$\boxed{\min_{\boldsymbol{w}}} \quad \underbrace{\frac{1}{2}\|\boldsymbol{w}\|^2}_{\text{Maximize margin}}$$

The *w* that minimizes…



$$\text{subject to} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 \qquad , \quad \forall i = 1, \ldots, N$$

Soft-margin SVMs

# data samples

Misclassification cost

Slack variable

$$\min_{\boldsymbol{w}} \underbrace{\frac{1}{2}\|\boldsymbol{w}\|^2}_{\text{Maximize margin}} + \underbrace{C\sum_{i=1}^{N}\xi_i}_{\text{Minimize misclassification}}$$

The *w* that minimizes...

$$\text{subject to} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i,$$
$$\xi_i \geq 0, \quad \forall i = 1, \ldots, N$$

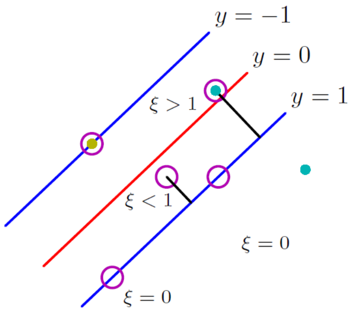
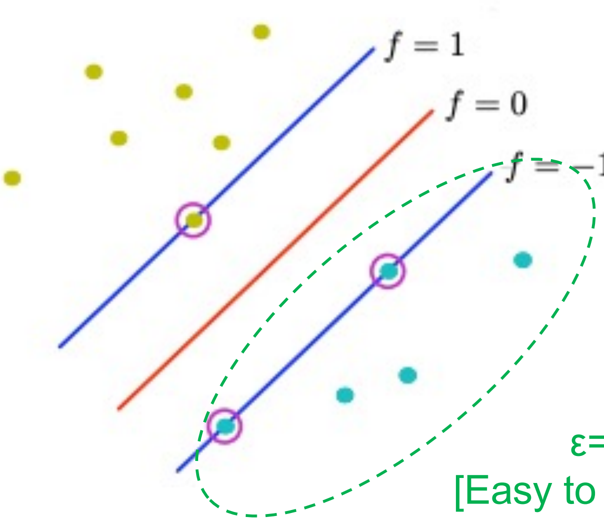


Figure from Chris Bishop

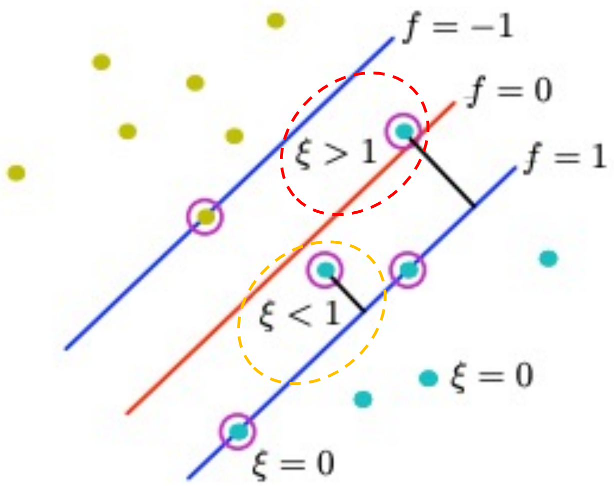
Soft-margin SVMs



ε=0
[Easy to classify]

Ideal Case

ε>1
[Miss-classified points]

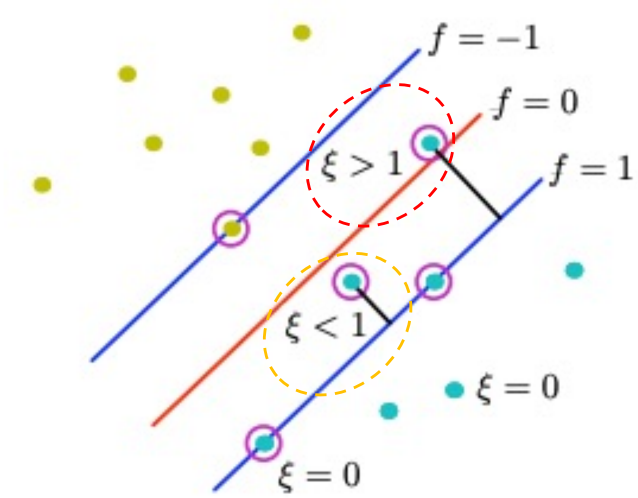


ε<1
[Points close to
decision boundary]

Figure from Chris Bishop

Soft-margin SVMs

ε>1
[Miss-classified points]



ε<1
[Points close to decision boundary]

Figure from Chris Bishop

Slack variables allow:

- Certain training points can be within the margin.
- We want these number of points as small as possible.

How do we minimize the second term in the optimization?

- A lot of examples with ε=0 (easy correctly classified)
- Medium quantity of examples with 0<ε<1 (correct classified inside margin)
- Few examples with ε>1 (misclassified examples)

What about multi-class SVMs?

- Unfortunately, there is no “definitive” multi-class SVM formulation
- In practice, we have to obtain a multi-class SVM by combining multiple two-class SVMs

- One vs. others/all
  - Training: learn an SVM for each class vs. the others
  - Testing: apply each SVM to the test example, and assign it to 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

Svetlana Lazebnik

Multi-class problems

- One-vs-all (a.k.a. one-vs-others)
  - Train K classifiers
  - In each, pos = data from class *i*, neg = data from classes other than *i*
  - The class with the most confident prediction wins
  - Example:
    - You have 4 classes, train 4 classifiers
    - 1 vs others: score 3.5
    - 2 vs others: score 6.2
    - 3 vs others: score 1.4
    - 4 vs other: score 5.5
    - Final prediction: class 2

Multi-class problems

- One-vs-one (a.k.a. all-vs-all)
  - Train K(K-1)/2 binary classifiers (all pairs of classes)
  - They all vote for the label
  - Example:
    - You have 4 classes, then train 6 classifiers
    - 1 vs 2, 1 vs 3, 1 vs 4, 2 vs 3, 2 vs 4, 3 vs 4
    - Votes: 1, 1, 4, 2, 4, 4
    - Final prediction is class 4

Using SVMs

1. Select a kernel function.

2. Compute pairwise kernel values between labeled examples.

3. Use this "kernel matrix" to solve for SVM support vectors & alpha weights.

4. To classify a new example: compute kernel values between new input and support vectors, apply alpha weights, check sign of output.

Adapted from Kristen Grauman

Some SVM packages

- LIBSVM http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- LIBLINEAR https://www.csie.ntu.edu.tw/~cjlin/liblinear/
- SVM Light http://svmlight.joachims.org/
- Scikit Learn https://scikit-learn.org/stable/modules/svm.html

Linear classifiers vs nearest neighbors

- Linear pros:
  - + Low-dimensional *parametric* representation
  - + Very fast at test time
- Linear cons:
  - Can be tricky to select best kernel function for a problem
  - Learning can take a very long time for large-scale problem
- NN pros:
  - + Works for any number of classes
  - + Decision boundaries not necessarily linear
  - + *Nonparametric* method
  - + Simple to implement
- NN cons:
  - Slow at test time (large search problem to find neighbors)
  - Storage of data
  - Especially need good distance function (but true for all classifiers)

Adapted from L. Lazebnik

Lab 5: SVM

Duration: 30 min

Use JPEG, PNG and GIF files less than 15 MB [ahaslides]

To join, go to: ahaslides.com/FHBCJ

Please, run linear SVM on our face dataset and upload your resulted accuracy.

Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories

CVPR 2006

Winner of 2016 Longuet-Higgins Prize

Svetlana Lazebnik (slazebni@uiuc.edu)

Beckman Institute, University of Illinois at Urbana-Champaign

Cordelia Schmid (cordelia.schmid@inrialpes.fr)

INRIA Rhône-Alpes, France

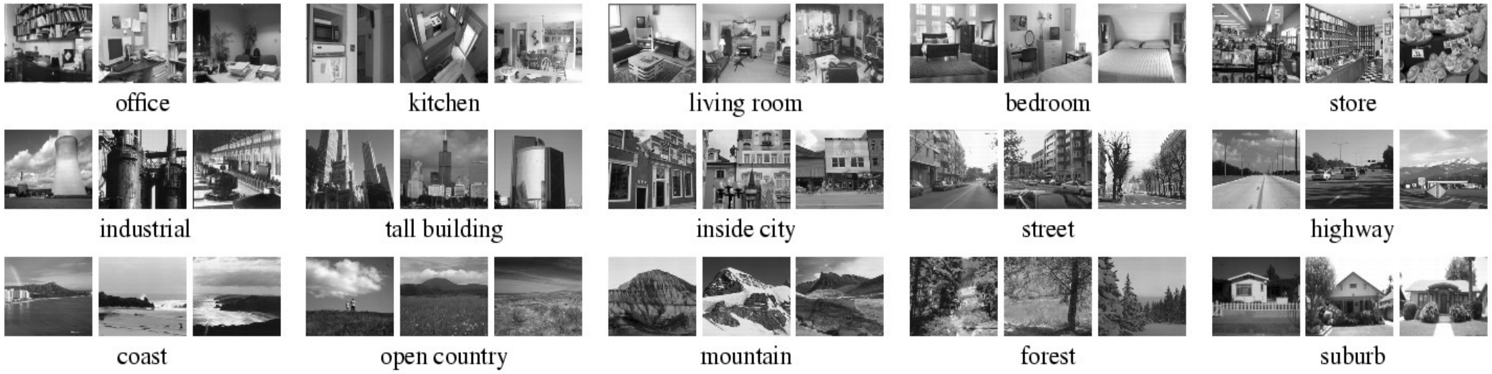
Jean Ponce (ponce@di.ens.fr)

Ecole Normale Supérieure, France

Scene category dataset

Fei-Fei & Perona (2005), Oliva & Torralba (2001)

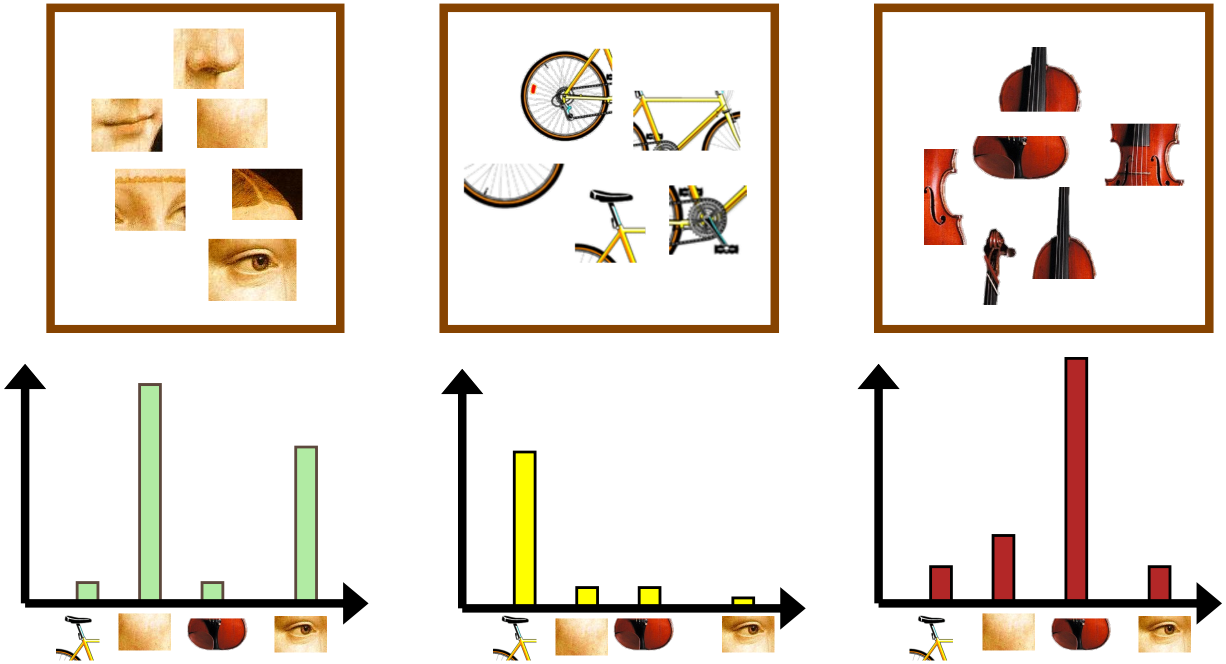
http://www-cvr.ai.uiuc.edu/ponce_grp/data



Slide credit: L. Lazebnik

Bag-of-words representation

1. Extract local features
2. Learn "visual vocabulary" using clustering
3. Quantize local features using visual vocabulary
4. Represent images by frequencies of "visual words"



Slide credit: L. Lazebnik

Image categorization with bag of words

Training

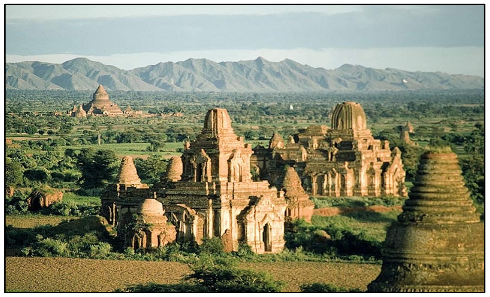
1. Compute bag-of-words representation for training images
2. Train classifier on labeled examples using histogram values as features
3. Labels are the scene types (e.g. mountain vs field)

Testing

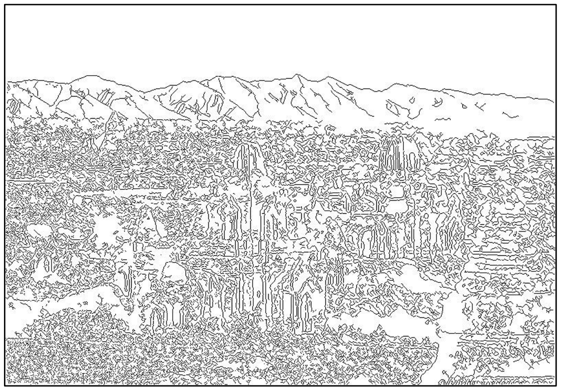
1. Extract keypoints / descriptors for test images
2. Quantize into visual words using the clusters computed at training time
3. Compute visual word histogram for test images
4. Compute labels on test images using classifier obtained at training time
5. **Evaluation only, do only once:** Measure accuracy of test predictions by comparing them to ground-truth test labels (obtained from humans)

Adapted from D. Hoiem

Feature extraction (on which BOW is based)

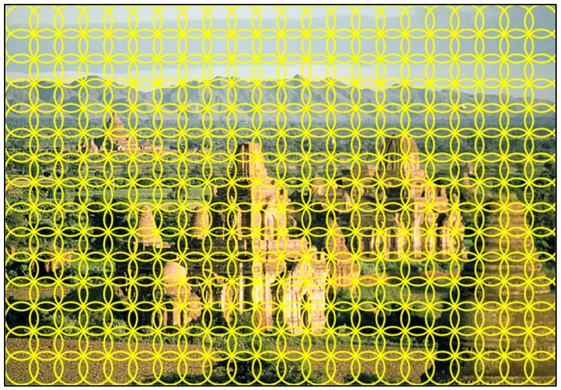


Weak features



Edge points at 2 scales and 8 orientations (vocabulary size 16)

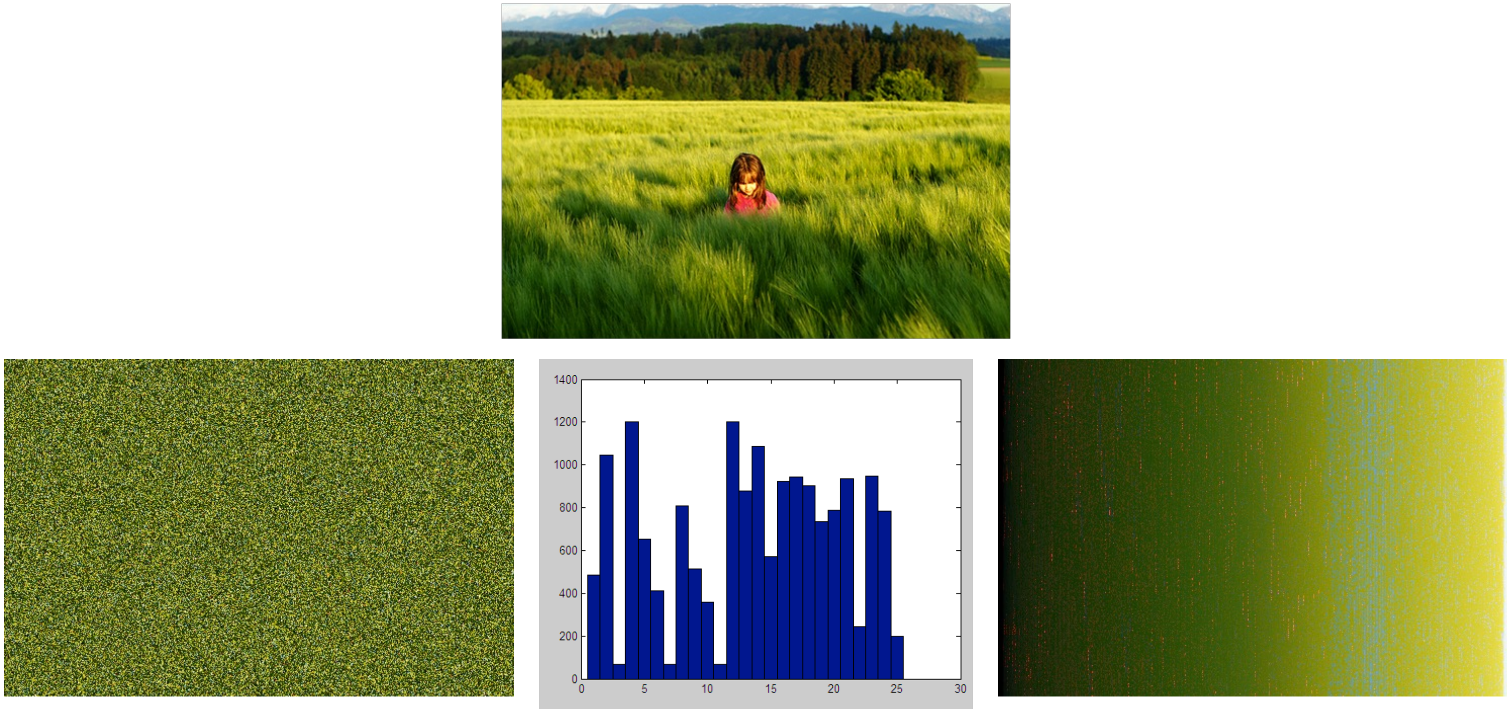
Strong features



SIFT descriptors of 16x16 patches sampled on a regular grid, quantized to form visual vocabulary (size 200, 400)

Slide credit: L. Lazebnik

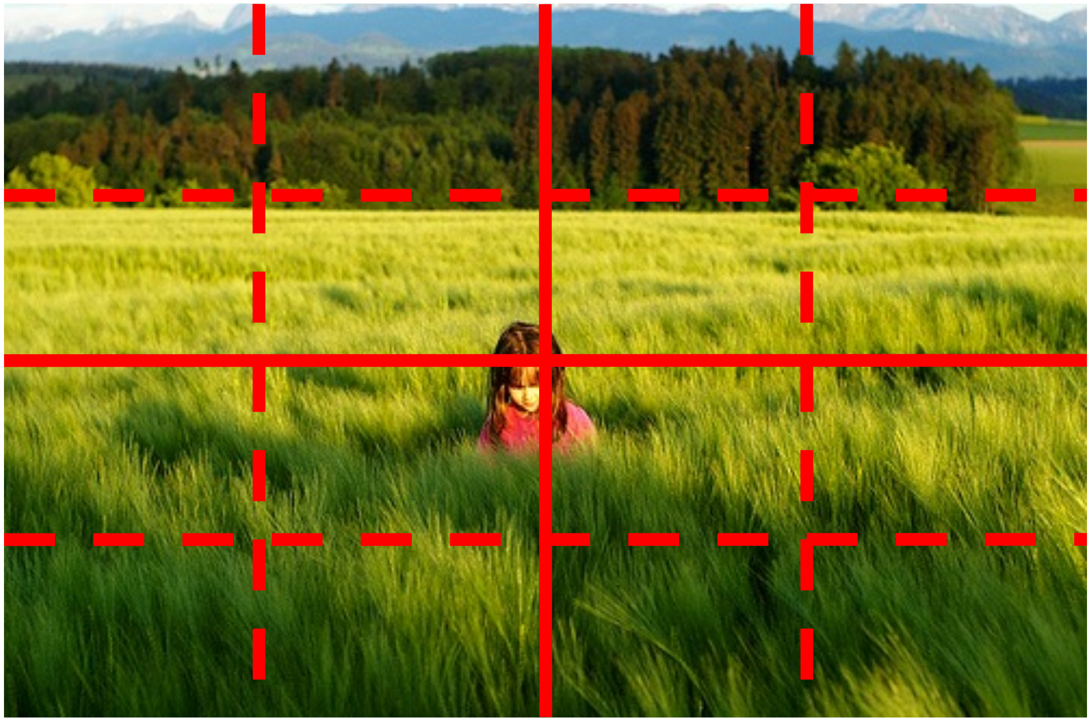
What about spatial layout?



All of these images have the same color histogram

Slide credit: D. Hoiem

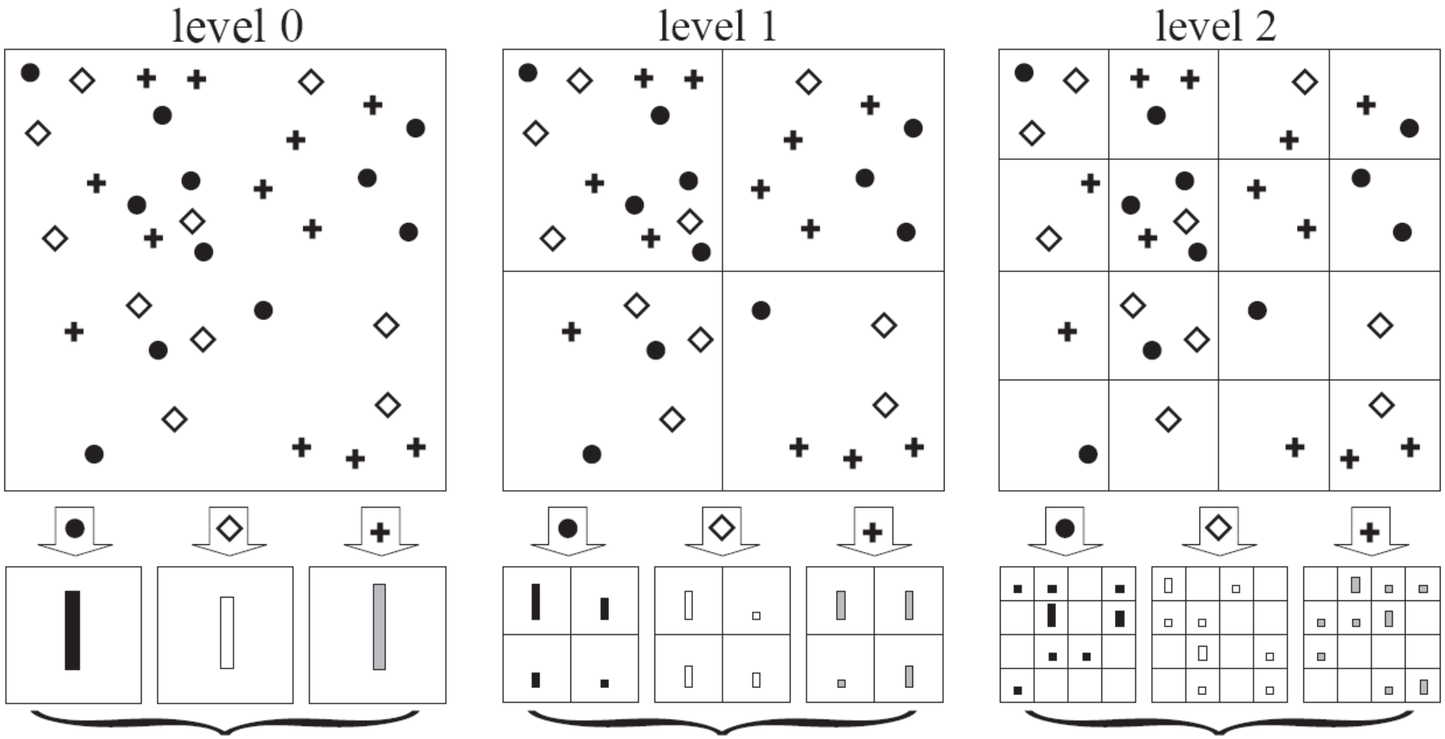
Spatial pyramid



Compute histogram in each spatial bin

Slide credit: D. Hoiem

Spatial pyramid



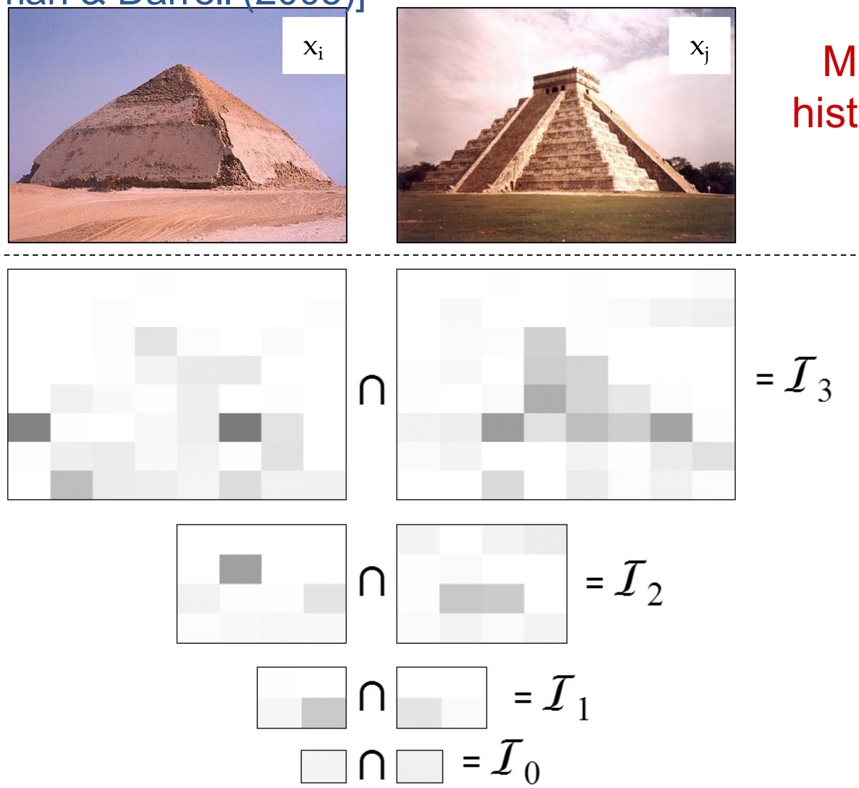
[Lazebnik et al. CVPR 2006]

Slide credit: D. Hoiem

Pyramid Matching

[Indyk & Thaper (2003), Grauman & Darrell (2005)]

Matching using pyramid and histogram intersection for some particular visual word:



Original images

Feature histograms:

Level 3

Level 2

Level 1

Level 0

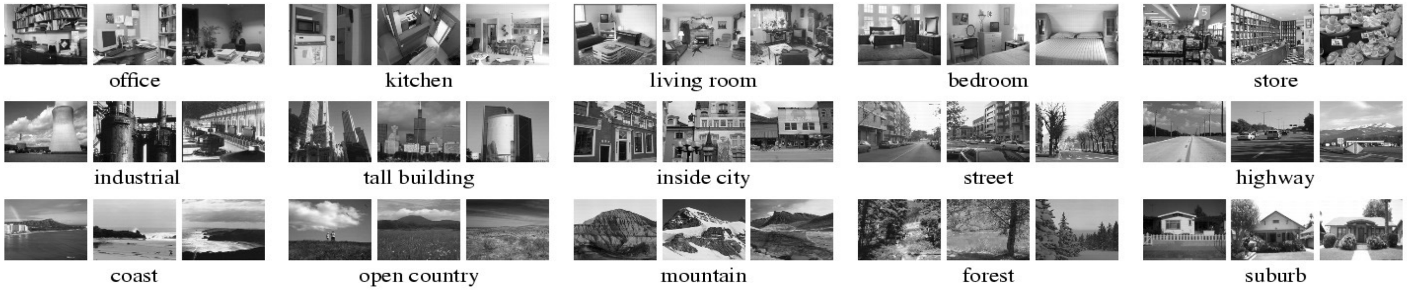
$K(x_i, x_j)$ (value of *pyramid match kernel*): $\mathcal{I}_3 + \frac{1}{2}(\mathcal{I}_2 - \mathcal{I}_3) + \frac{1}{4}(\mathcal{I}_1 - \mathcal{I}_2) + \frac{1}{8}(\mathcal{I}_0 - \mathcal{I}_1)$

Adapted from L. Lazebnik

Scene category dataset

Fei-Fei & Perona (2005), Oliva & Torralba (2001)

http://www-cvr.ai.uiuc.edu/ponce_grp/data



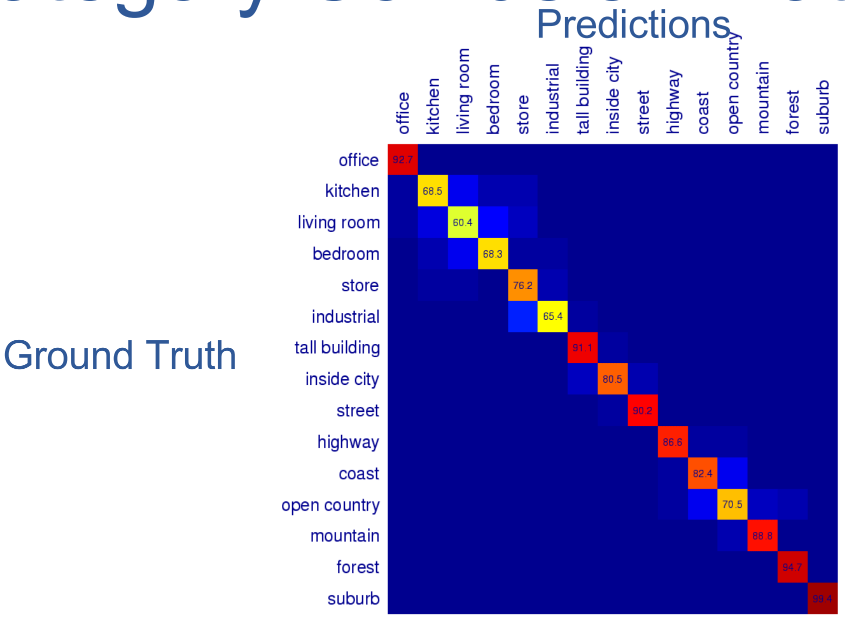
Multi-class classification results (100 training images per class)

	Weak features (vocabulary size: 16)		Strong features (vocabulary size: 200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0 (1 × 1)	45.3 ±0.5		72.2 ±0.6	
1 (2 × 2)	53.6 ±0.3	56.2 ±0.6	77.9 ±0.6	79.0 ±0.5
2 (4 × 4)	61.7 ±0.6	64.7 ±0.7	79.4 ±0.3	**81.1** ±0.3
3 (8 × 8)	63.3 ±0.8	**66.8** ±0.6	77.2 ±0.4	80.7 ±0.3

Fei-Fei & Perona: 65.2%

Slide credit: L. Lazebnik

Scene Category Confusion Matrix



Difficult indoor images

kitchen

living room

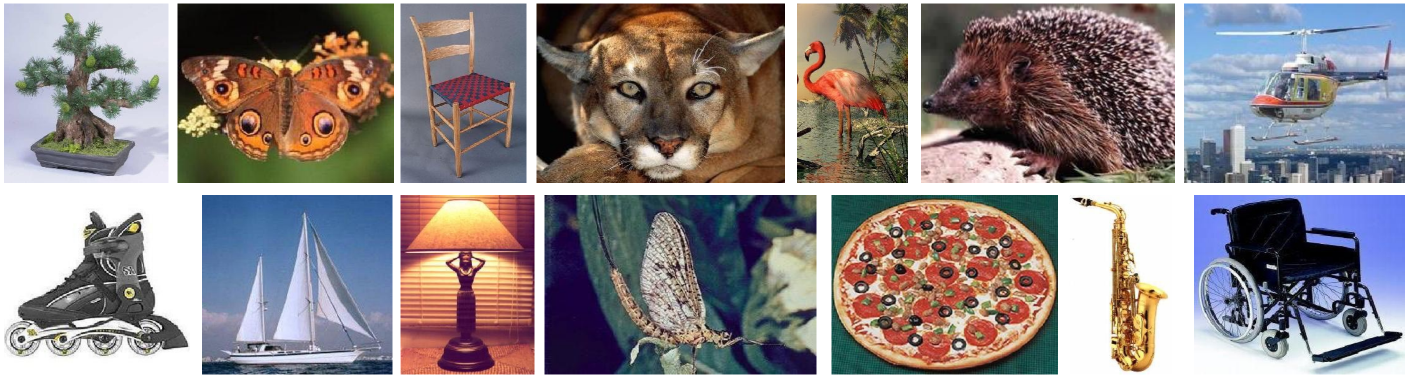
bedroom

Slide credit: L. Lazebnik

Caltech101 dataset

Fei-Fei et al. (2004)

http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html

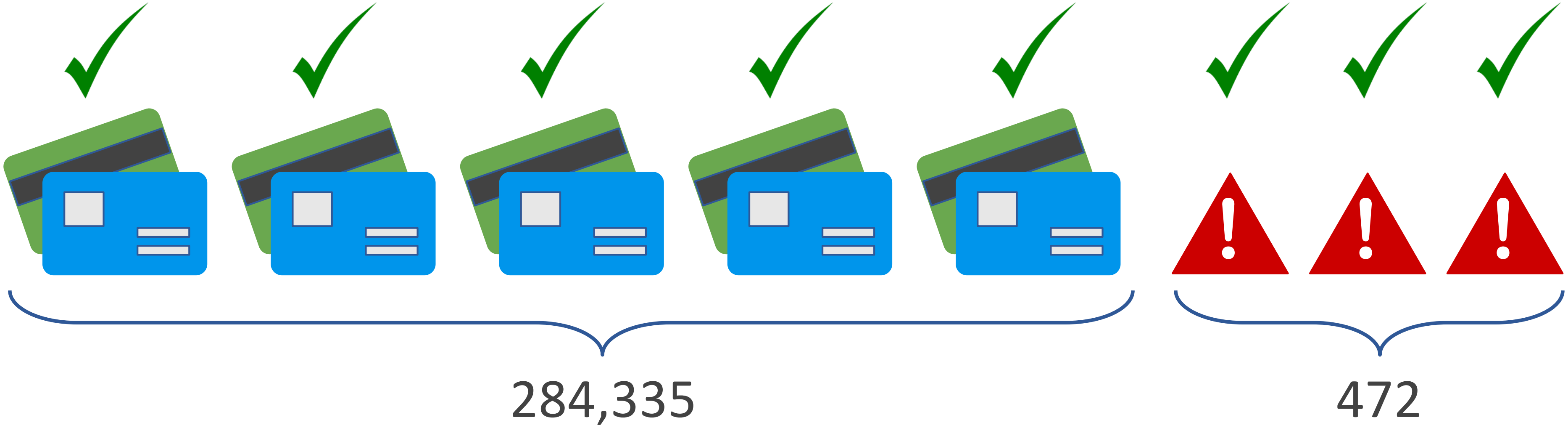


Multi-class classification results (30 training images per class)

	Weak features (16)		Strong features (200)	
Level	Single-level	Pyramid	Single-level	Pyramid
0	15.5 ±0.9		41.2 ±1.2	
1	31.4 ±1.2	32.8 ±1.3	55.9 ±0.9	57.0 ±0.8
2	47.2 ±1.1	49.3 ±1.4	63.6 ±0.9	**64.6** ±0.8
3	52.2 ±0.8	**54.0** ±1.1	60.3 ±0.9	64.6 ±0.7

Slide credit: L. Lazebnik

Evaluation Metrics: Credit Card Fraud

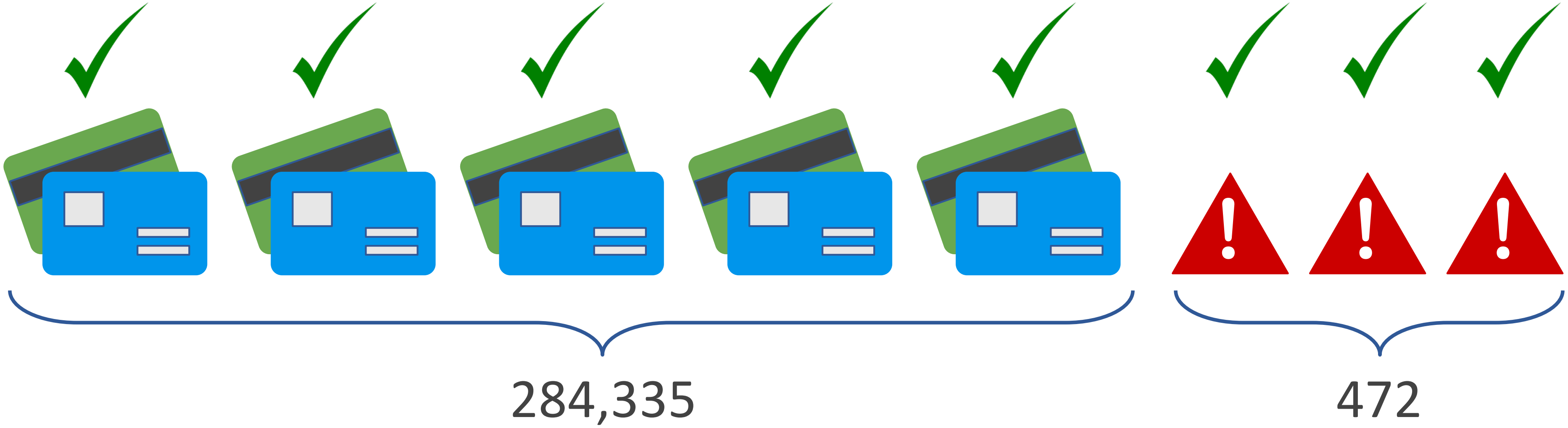


Model: All transactions are good.

$$\text{Correct} = \frac{284{,}335}{284{,}807} = 99.83\%$$

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Credit Card Fraud

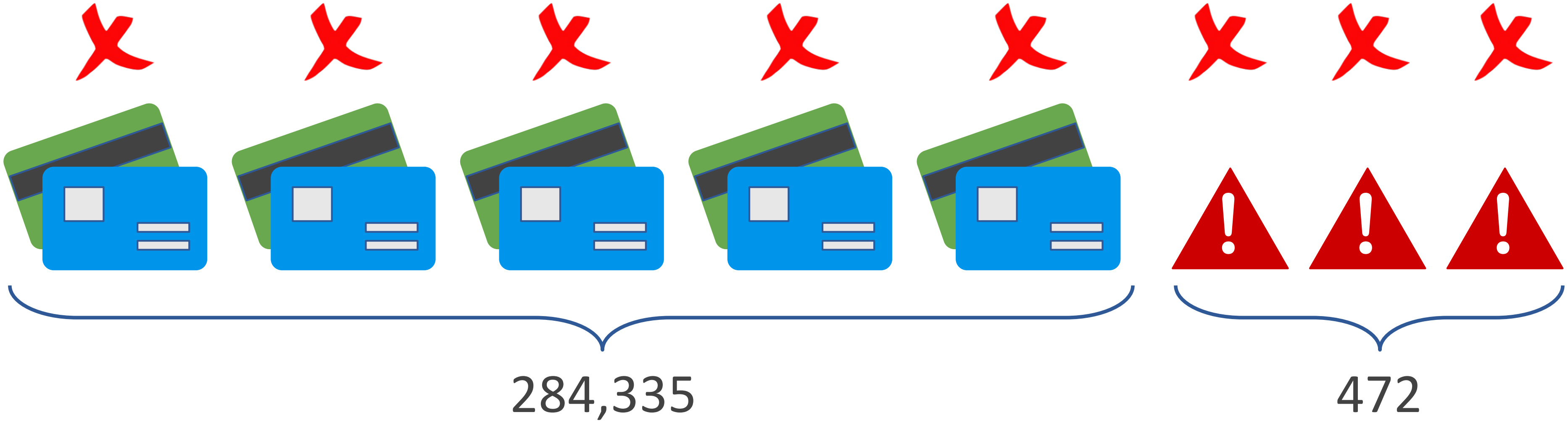


Model: All transactions are good.

Problem: I’m not catching any of the bad ones!

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Credit Card Fraud



Model: All transactions are fraudulent.

Problem: I’m accidently catching all the good ones!

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Spam Classifier Model

Not Spam

Spam

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table

Predictions

Annotations

	Sent to Spam Folder	Sent to Inbox
Spam	True Positive	False Negative
Not Spam	False Positive	True Negative

Slide Credit: Prof. Sandra Avila - UNICAMP

Slide Credit: Prof. Sandra Avila - UNICAMP

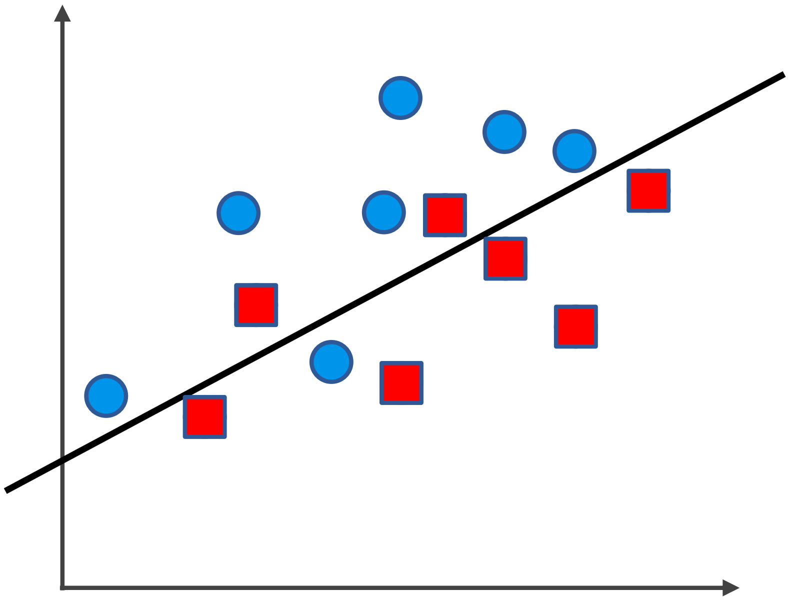
Evaluation Metrics: Confusion Matrix Table

1,000 emails

		Folder	
		Spam Folder	Inbox
Email	Spam	100	170
	Not Spam	30	700

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table



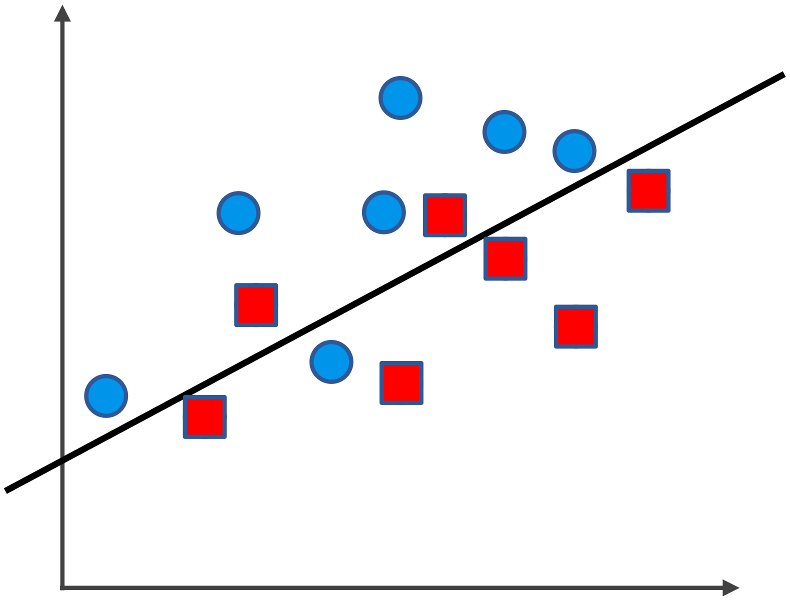
Prediction

Data

	Guessed Positive	Guessed Negative
Positive		
Negative		

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table

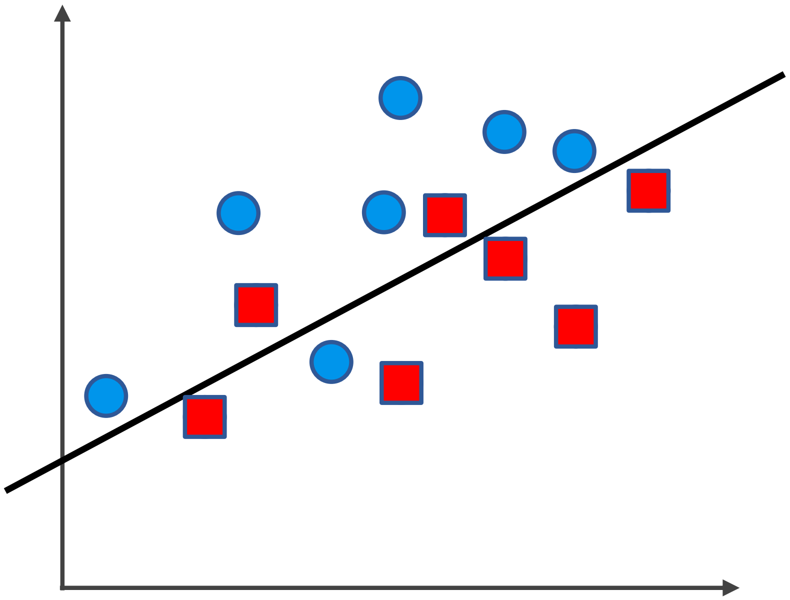


Prediction

Data	Guessed Positive	Guessed Negative
Positive	6 True positives	
Negative		

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table

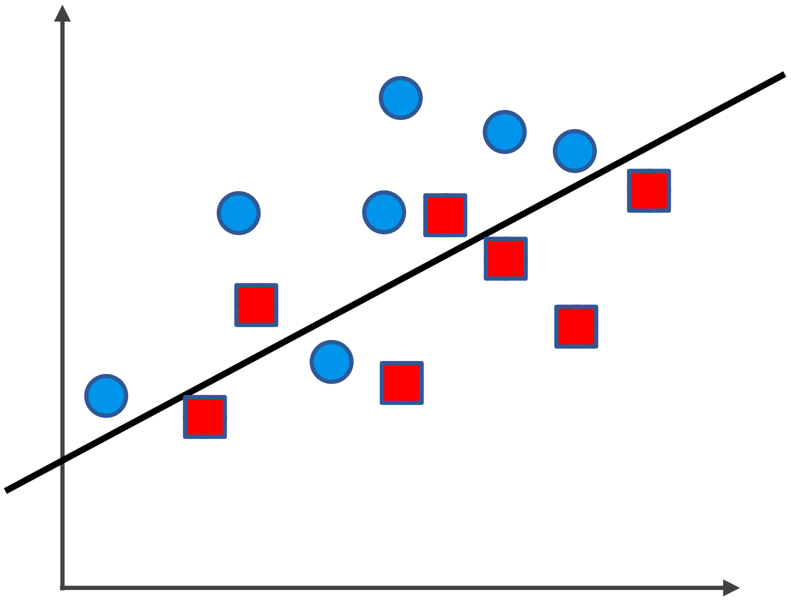


Prediction

		Guessed Positive	Guessed Negative
Data	Positive	6 True positives	
	Negative		5 True negatives

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table



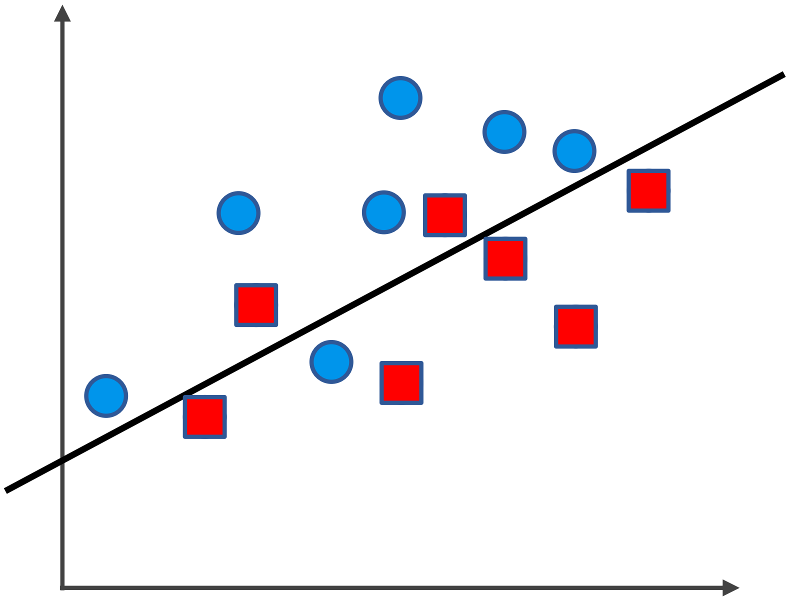
Prediction

Data

	Guessed Positive	Guessed Negative
Positive	6 True positives	1 False negative
Negative		5 True negatives

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table



Prediction

Data

	Guessed Positive	Guessed Negative
Positive	6 True positives	1 False negative
Negative	2 False positives	5 True negatives

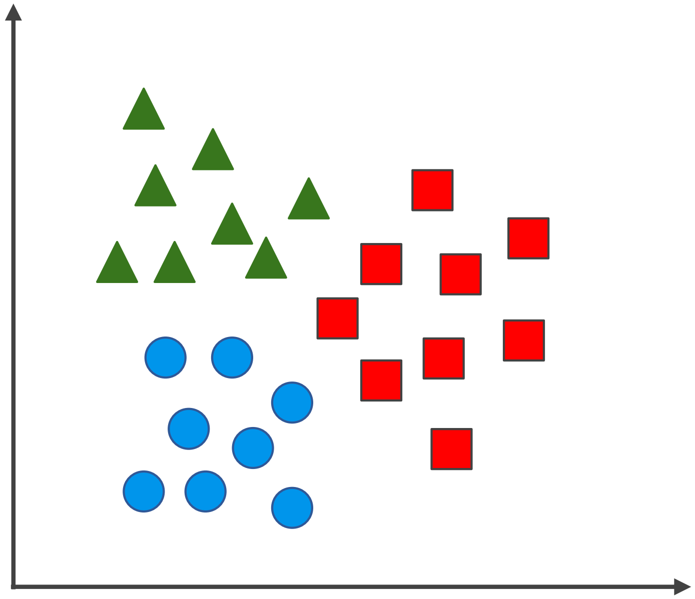
Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table (*n* classes)

Class 1: ▲

Class 2: 🟥

Class 3: 🔵



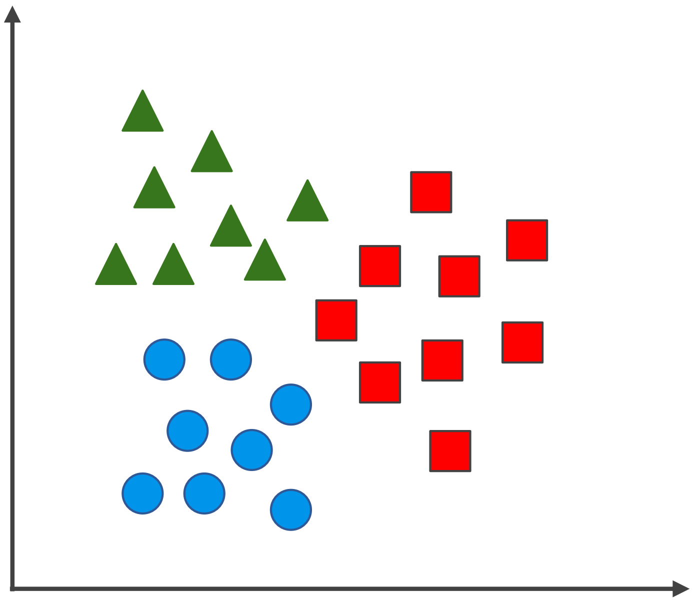
Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table (n classes)

Class 1: ▲

Class 2: 🟥

Class 3: ●



		Predicted Class		
		Guessed Class 1	Guessed Class 2	Guessed Class 3
True Class	Class 1			
	Class 2			
	Class 3			

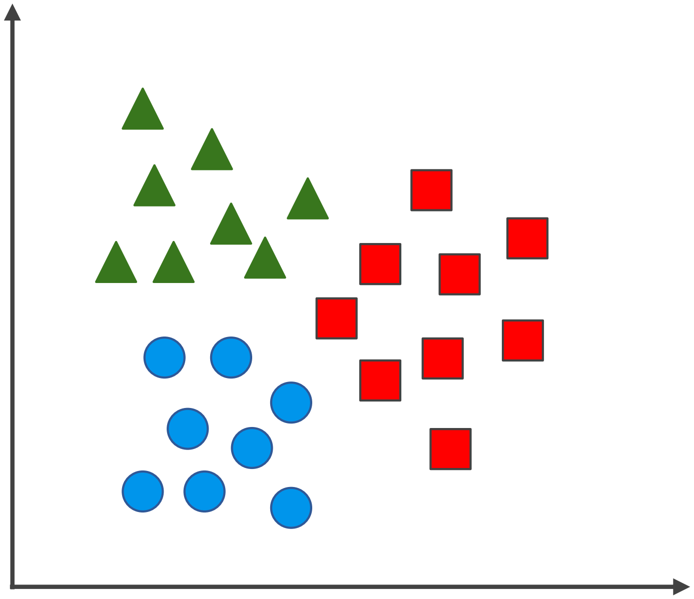
Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Confusion Matrix Table (*n* classes)

Class 1: ▲

Class 2: 🟥

Class 3: ●



Predicted Class

True Class

	Guessed Class 1	Guessed Class 2	Guessed Class 3
Class 1	5	2	1
Class 2	3	6	0
Class 3	0	1	7

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Accuracy

Diagnosis

		Diagnosed Sick	Diagnosed Healthy
Patients	Sick	1,000	200
	Healthy	800	8,000

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Accuracy

		Diagnosis	
		Diagnosed Sick	Diagnosed Healthy
Patients	Sick	1,000	200
	Healthy	800	8,000

Accuracy:
Out of all the **patients**, how many did we classify correctly?

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Accuracy

		Diagnosis	
		Diagnosed Sick	Diagnosed Healthy
Patients	Sick	1,000	200
	Healthy	800	8,000

Accuracy:
Out of all the **patients**, how many did we classify correctly?

Accuracy =

$$\frac{1,000 + 8,000}{\quad}$$

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Accuracy

		Diagnosis	
		Diagnosed Sick	Diagnosed Healthy
Patients	Sick	1,000	200
	Healthy	800	8,000

Accuracy:
Out of all the **patients**, how many did we classify correctly?

Accuracy =

$$\frac{1{,}000 + 8{,}000}{10{,}000} = 90\%$$

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Accuracy

		Folder	
		Spam Folder	Inbox
Email	Spam	100	170
	Not Spam	30	700

Accuracy:
Out of all the **emails**, how many did we classify correctly?

Slide Credit: Prof. Sandra Avila - UNICAMP

Evaluation Metrics: Accuracy

		Folder	
		Spam Folder	Inbox
Email	Spam	100	170
	Not Spam	30	700

Accuracy:
Out of all the **emails**, how many did we classify correctly?

Accuracy =

$$\frac{100 + 700}{1{,}000} = 80\%$$

Slide Credit: Prof. Sandra Avila - UNICAMP

Summary

- Visual Classification
- Models
  - K-Nearest Neighbor
  - Suppor Vector Machines (SVM)
- Spatial Pyramid Feature Extractor
- Evaluation Metrics