CAP5415 Computer Vision

Yogesh S Rawat

yogesh@ucf.edu

HEC-241

Course project

- Deadline – October 30
- Send a short description
- Deliverables
  - Code
    - Well commented
    - Modular
  - Report
    - Describe the problem
    - Describe your approach
    - Evaluation, dataset
    - Results
    - Discussion and your analysis
  - Demo – if no other evaluation
    - For example, edge detection in videos

PA-3

- Optional

Questions?

Classification - I

Lecture 11

Outline

- Basics – background
- Nearest neighbor (NN) classifier
- Maximum margin classifier
- Support vector machine

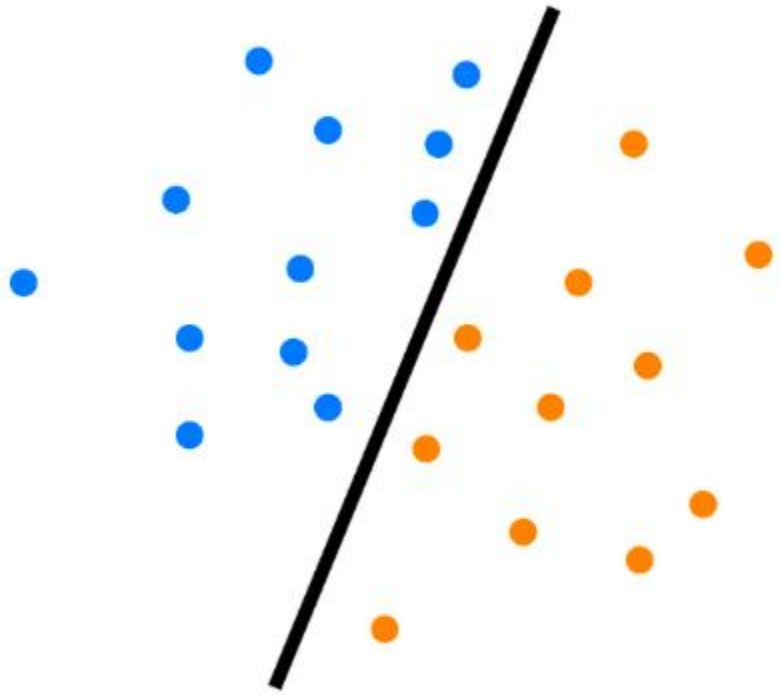
Classification - I

Lecture 11

Basics- background

Classification

- Categorizing given set of data into classes



Classification – computer vision

- Digit classification
  - MNIST
  - Multi-mnist
  - SVHN



- Object classification
  - CIFAR-10
  - CIFAR-100
  - ImageNET



Object Recognition

- Problem: Given an image A, does A contain an image of a person?

Object Recognition

- Problem: Given an image A, does A contain an image of a person?











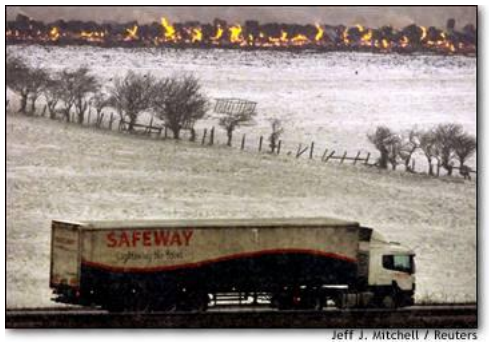
YES

Object Recognition

- Problem: Given an image A, does A contain an image of a person?











NO

Image classification - ImageNet

- Images for each category of WordNet
- 1000 classes
- 14M images
- 100K test



Dataset split

Training Images



- Train classifier

Validation Images



- Measure error
- Tune model hyperparameters

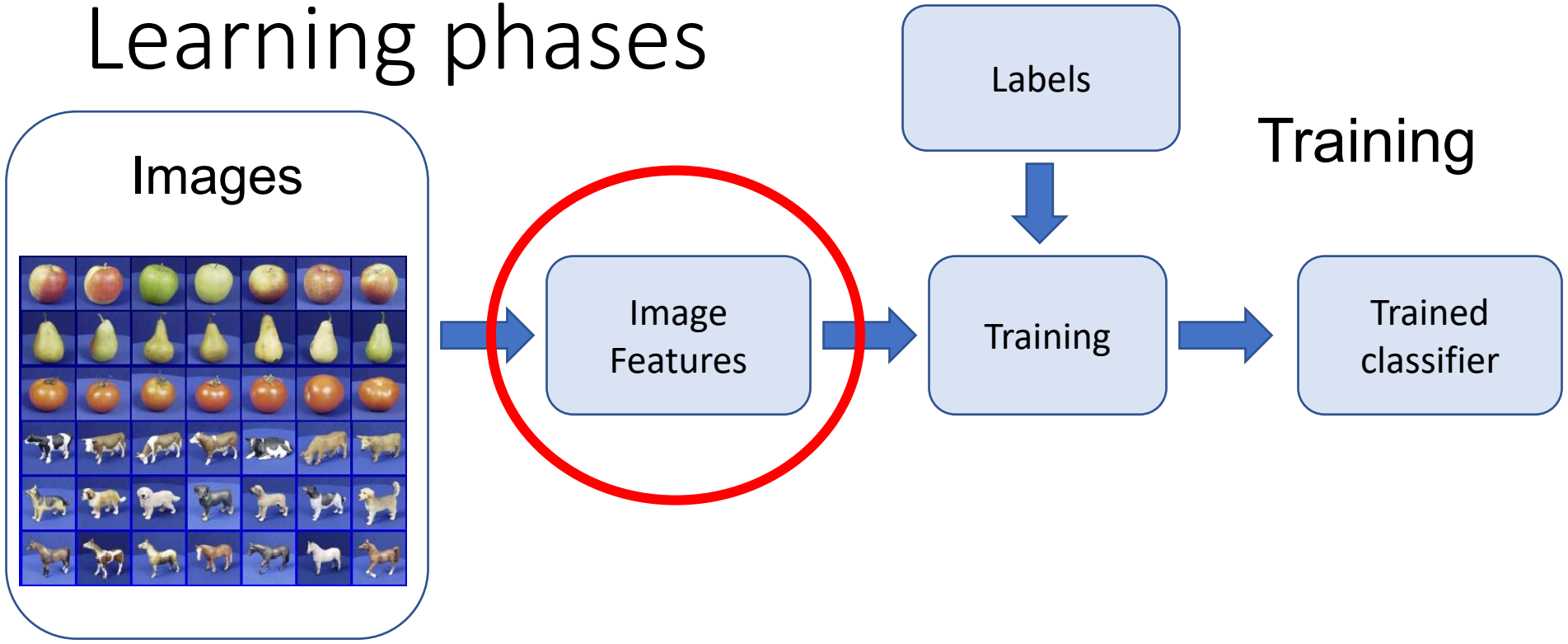
Testing Images



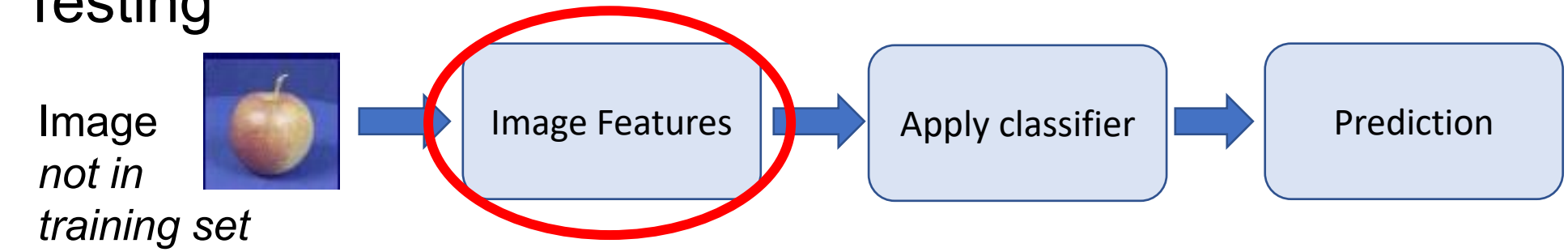
- Secret labels
- Measure error

Random train/validate splits = cross validation

Learning phases



Testing

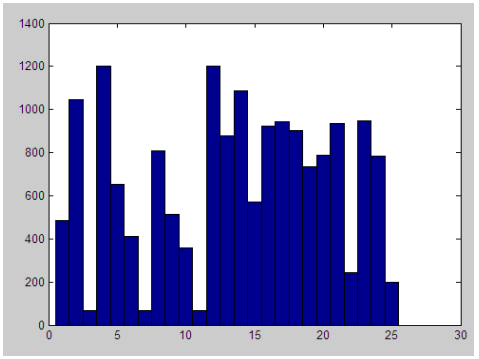


Slide credit: D. Hoiem and L. Lazebnik

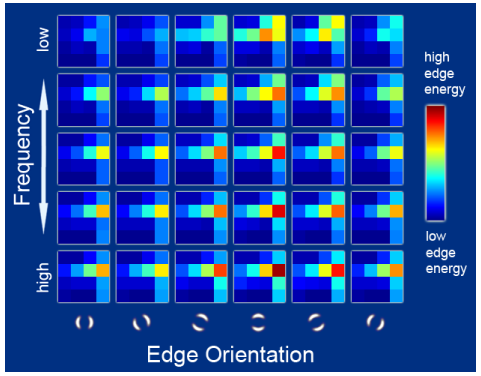
Features

- Raw pixels
- Histograms
- Templates
- SIFT
- GIST
- HOG
- SURF
- …



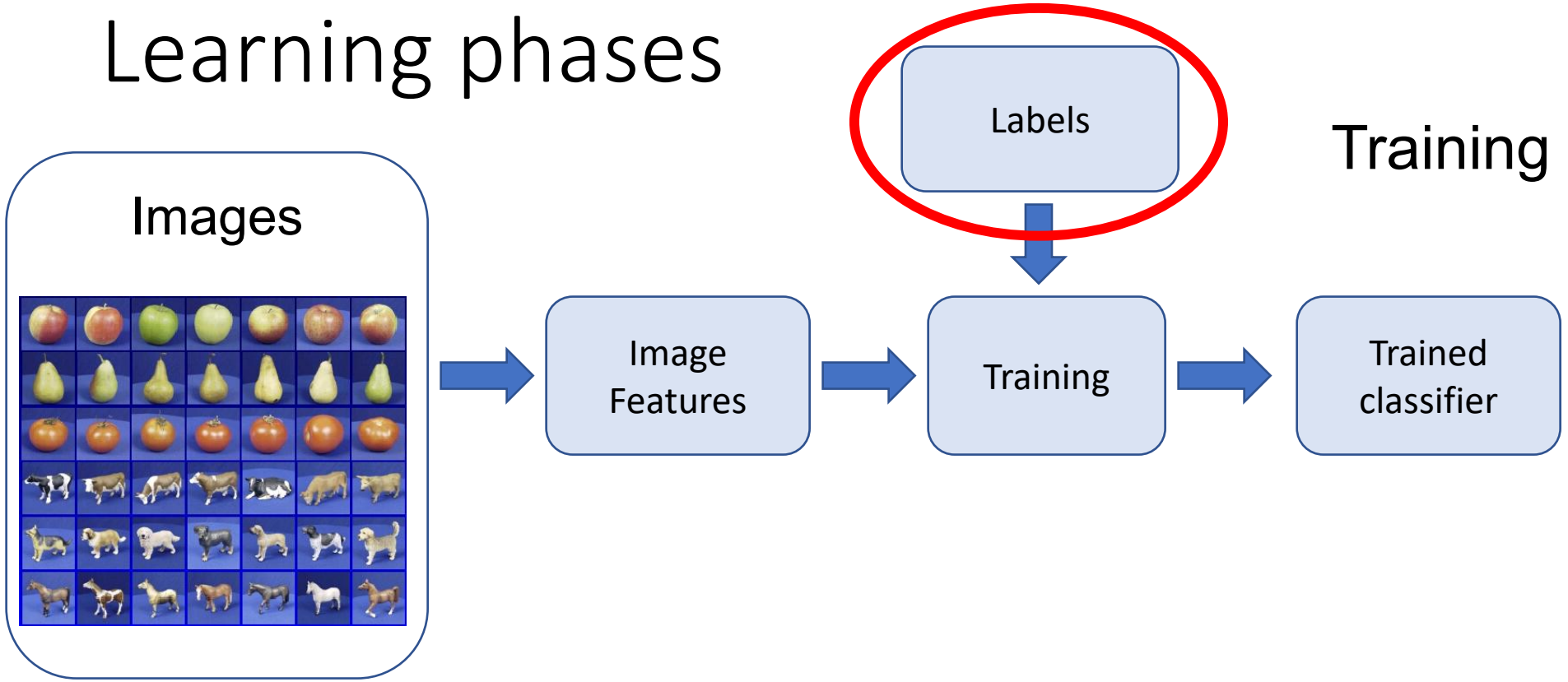




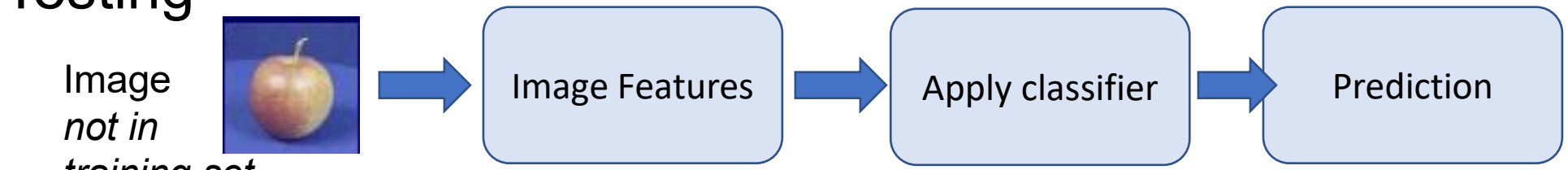


L. Lazebnik

Learning phases

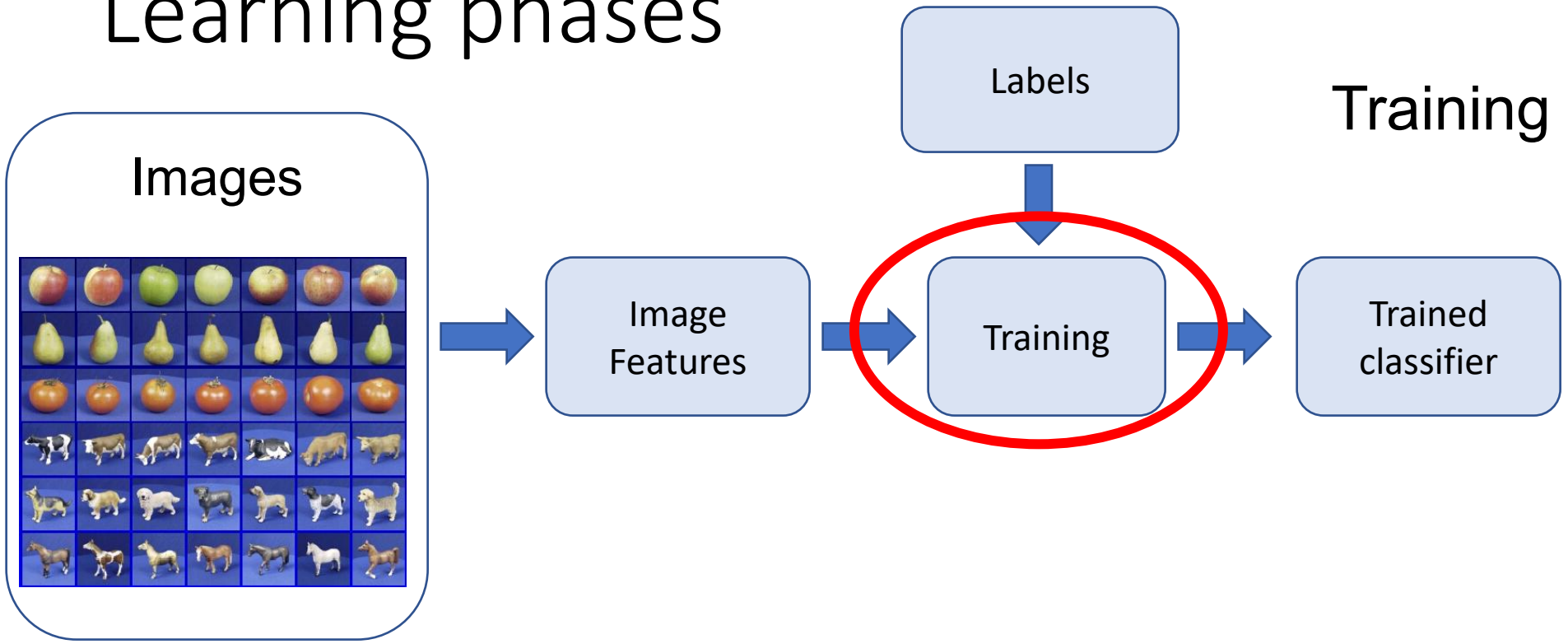


Testing

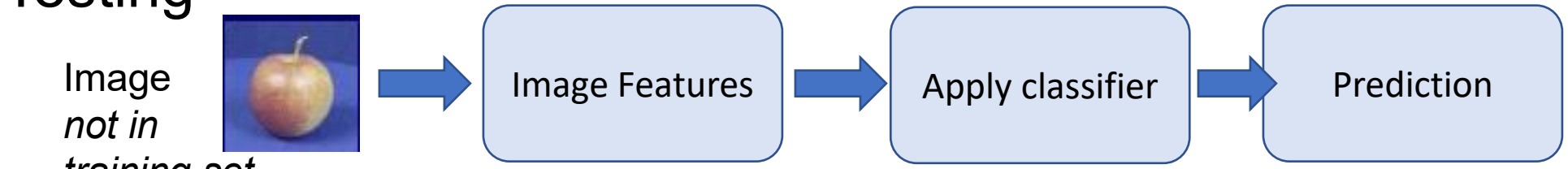


Slide credit: D. Hoiem and L. Lazebnik

Learning phases



Testing



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.

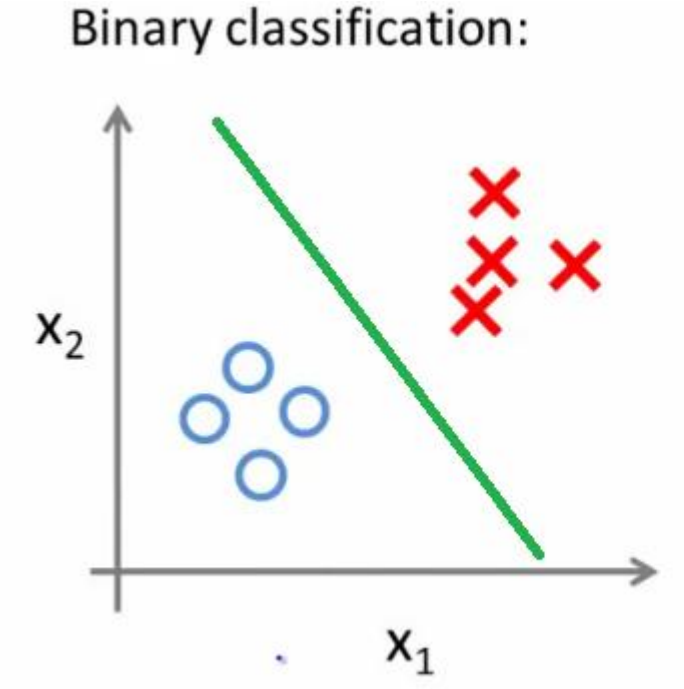
Testing: Apply f to an unseen *test example* $\mathbf{x}_u$ and output the predicted value $y_u = f(\mathbf{x}_u)$ to *classify* $\mathbf{x}_u$.

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* – literally boundaries in the space of the features.

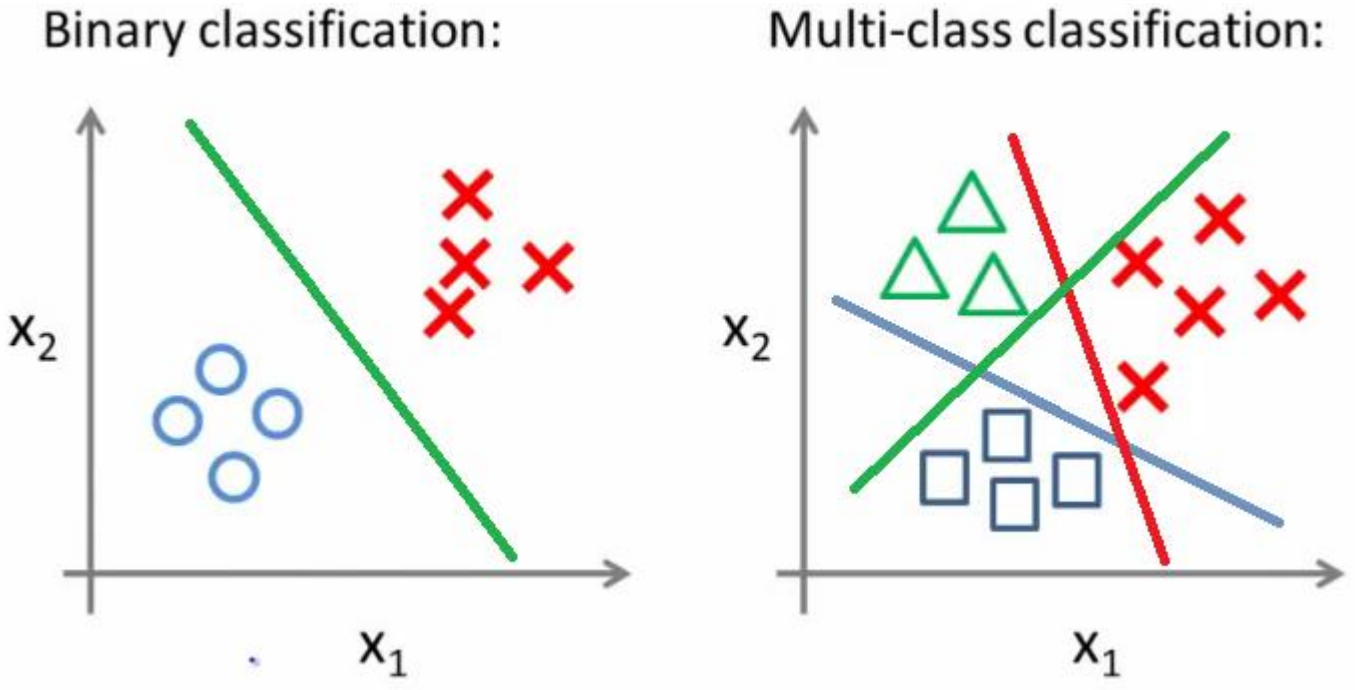


L. Lazebnik

Classification

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

A decision rule divides input space into *decision regions* separated by *decision boundaries* – literally boundaries in the space of the features.



L. Lazebnik

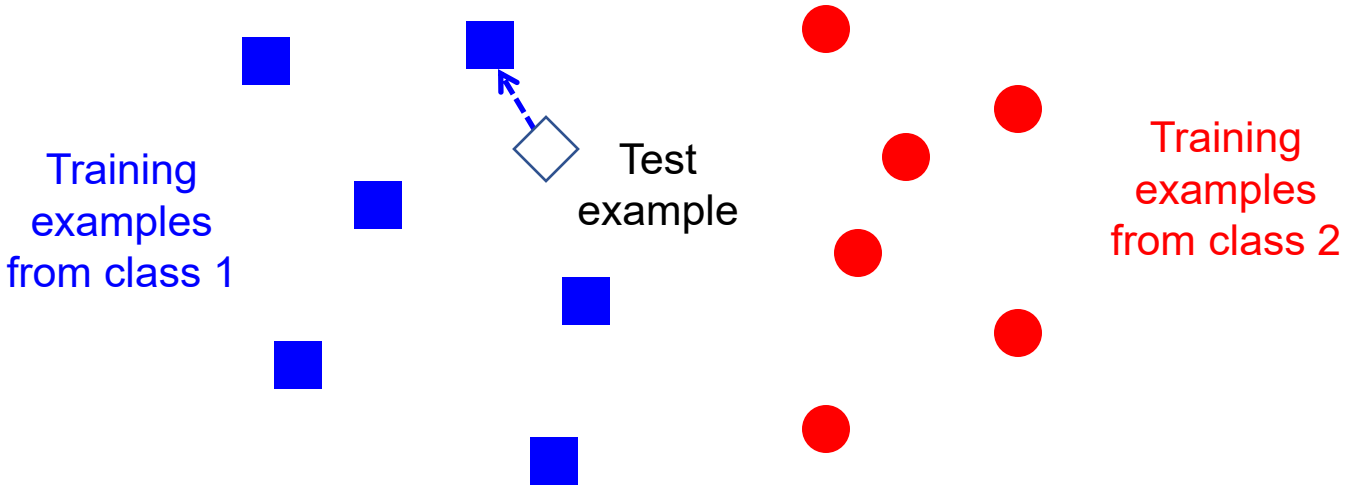
Questions?

Classification - I

Lecture 11

Nearest neighbor classifier

Classifiers: Nearest neighbor

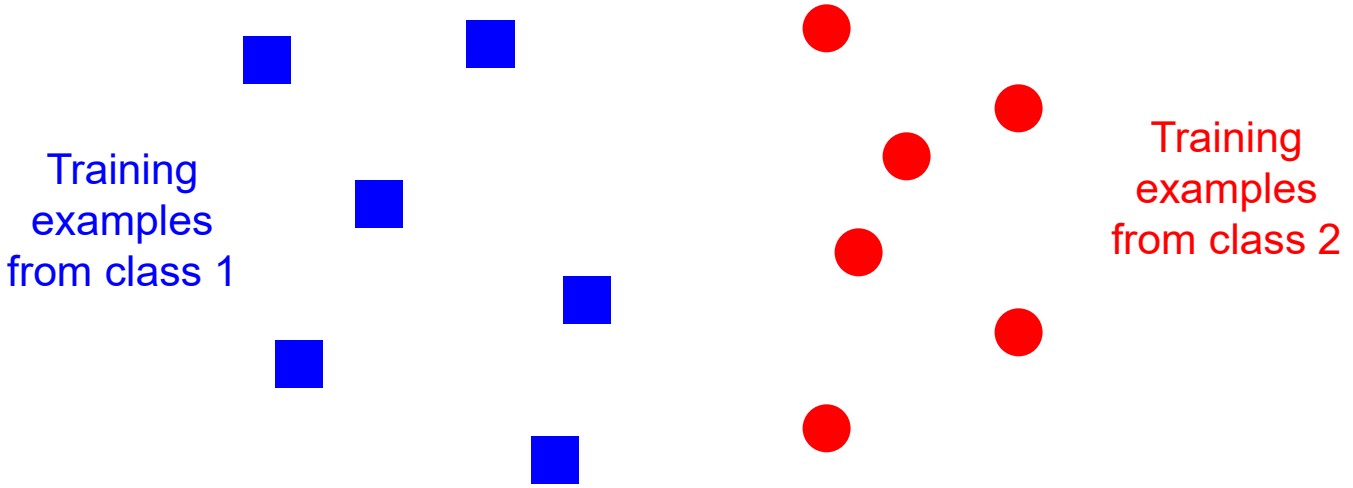


f(**x**) = label of the training example nearest to **x**

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

Slide credit: L. Lazebnik

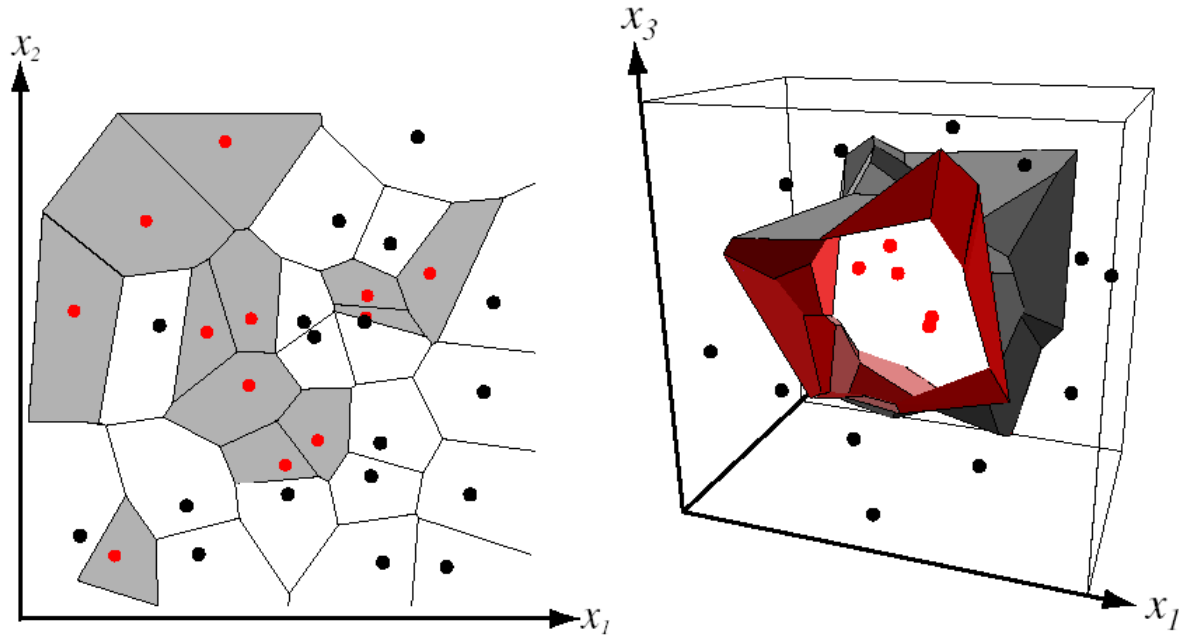
Classifiers: Nearest neighbor



What does the decision boundary look like?

Decision boundary for NN Classifier

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

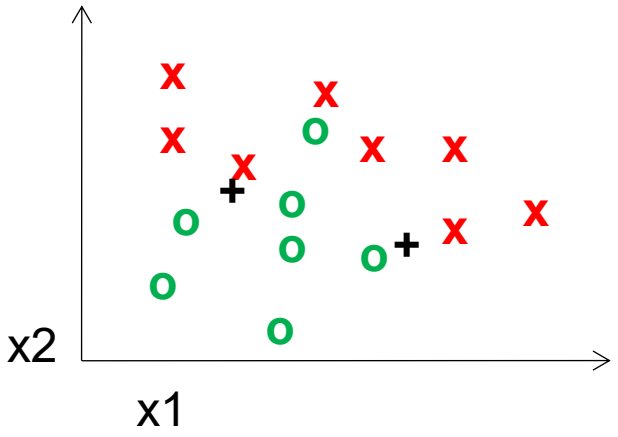


Voronoi partitioning of feature space for two-category 2D and 3D data

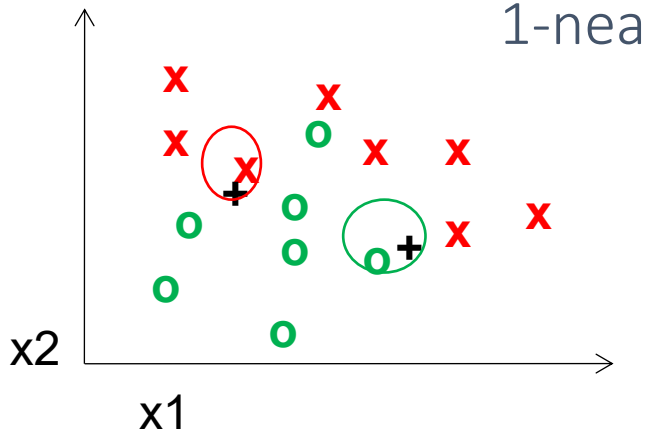
from Duda *et al.*

Source: D. Lowe

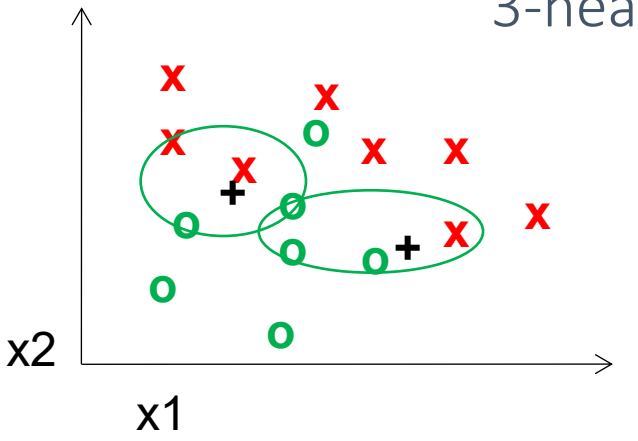
K-nearest neighbor



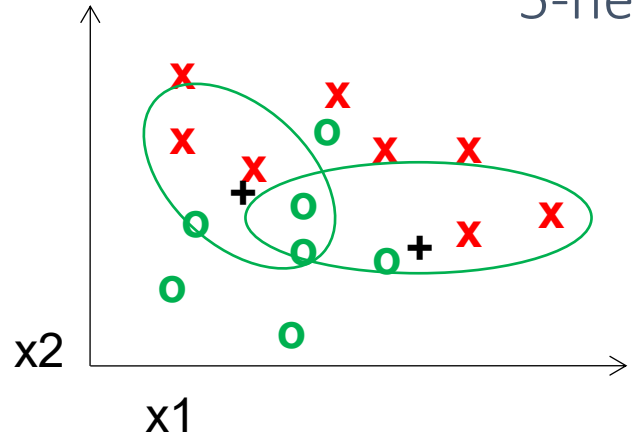
1-nearest



3-nearest



5-nearest



Algorithm

1. Get features for training samples
2. Get features for testing sample
3. For each training sample
    1. Computer distance from testing sample
4. Sort distance vector for all samples
5. Pick classes based on top k samples

K-nearest neighbor

- Pros
  - Simple and easy to implement
  - No training required
  - Classification, regression, search, etc.

- Cons
  - Slow with large samples
  - Slow with high dimensional features

Questions?

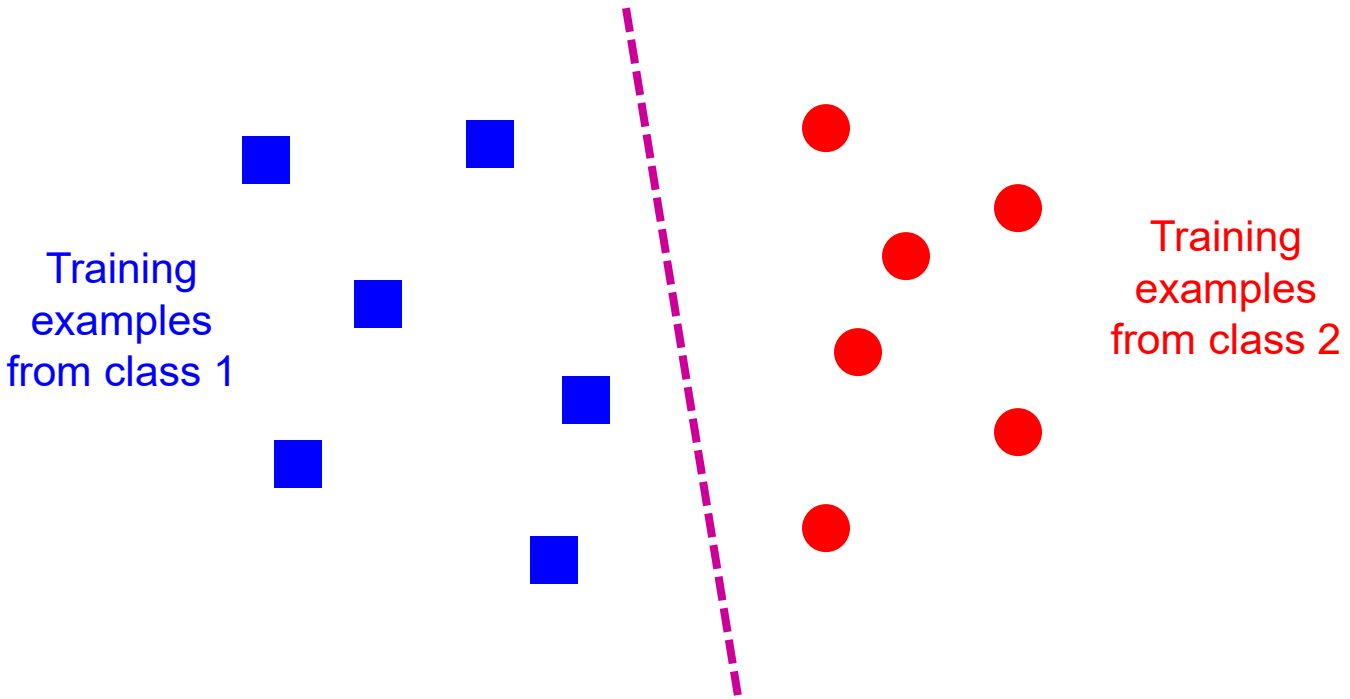
Pop quiz?

Classification - I

Lecture 11

Maximum margin classifier

Classifiers: Linear



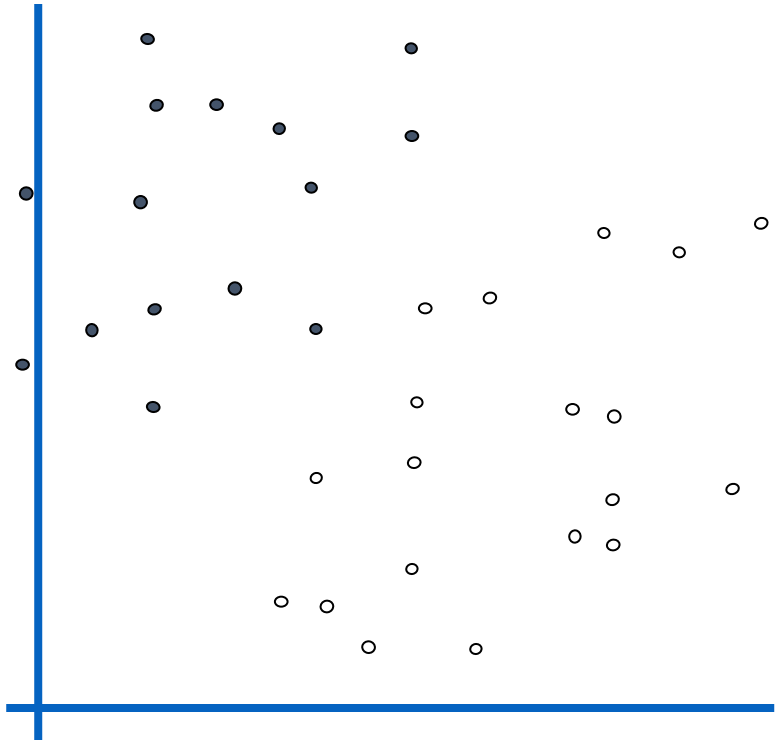
Find a *linear function* to separate the classes

Slide credit: L. Lazebnik

Motivation

denotes +1

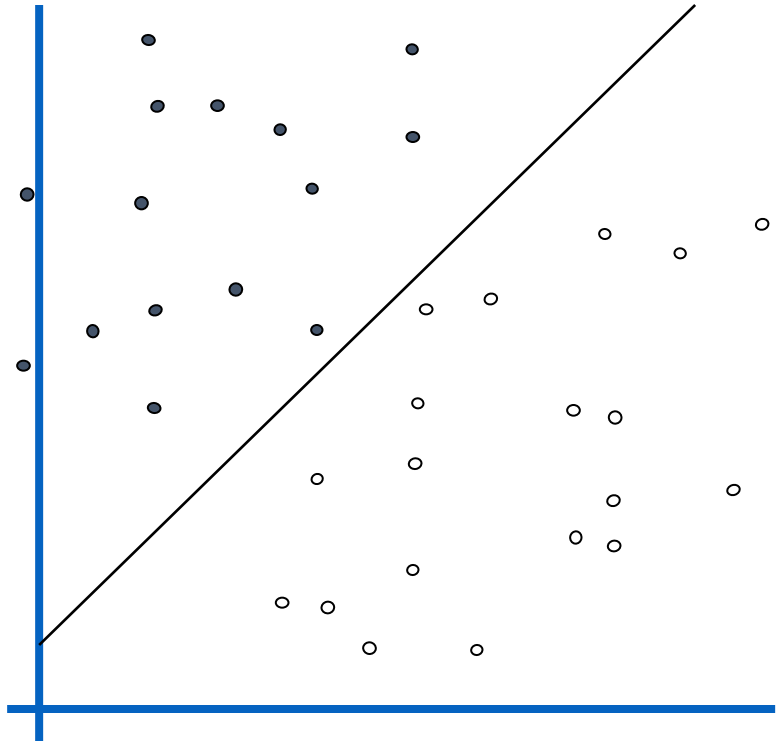
denotes -1



How would you classify this data?

Motivation

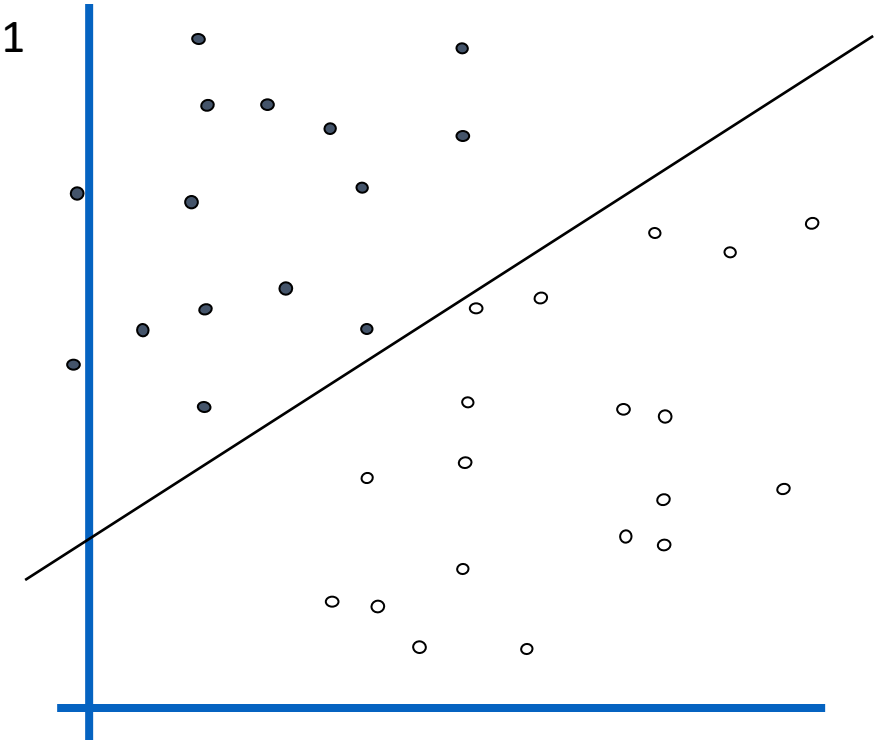
denotes +1
denotes -1



How would you classify this data?

Motivation

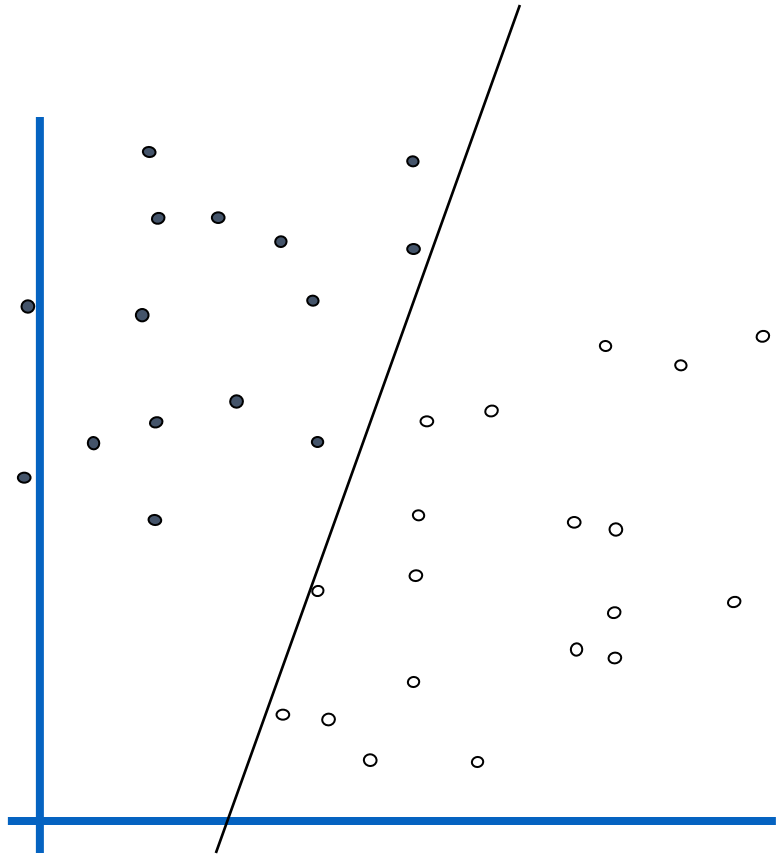
denotes +1
denotes -1



How would you classify this data?

Motivation

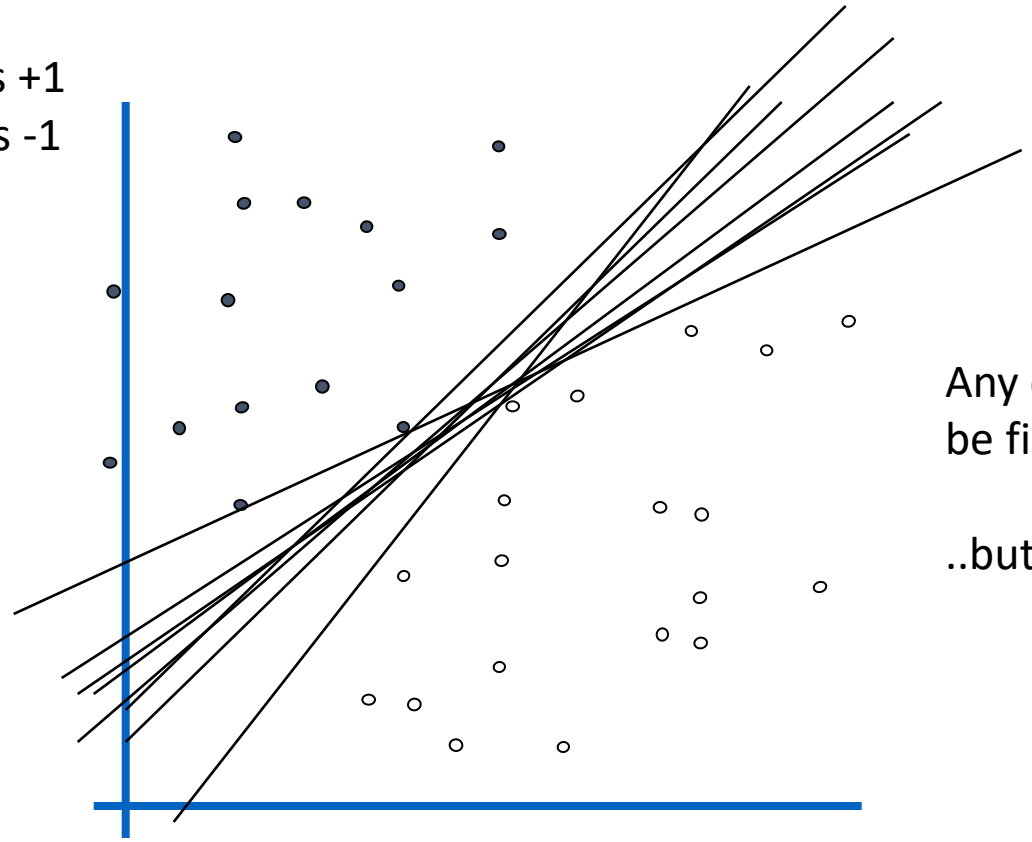
denotes +1
denotes -1



How would you classify this data?

Motivation

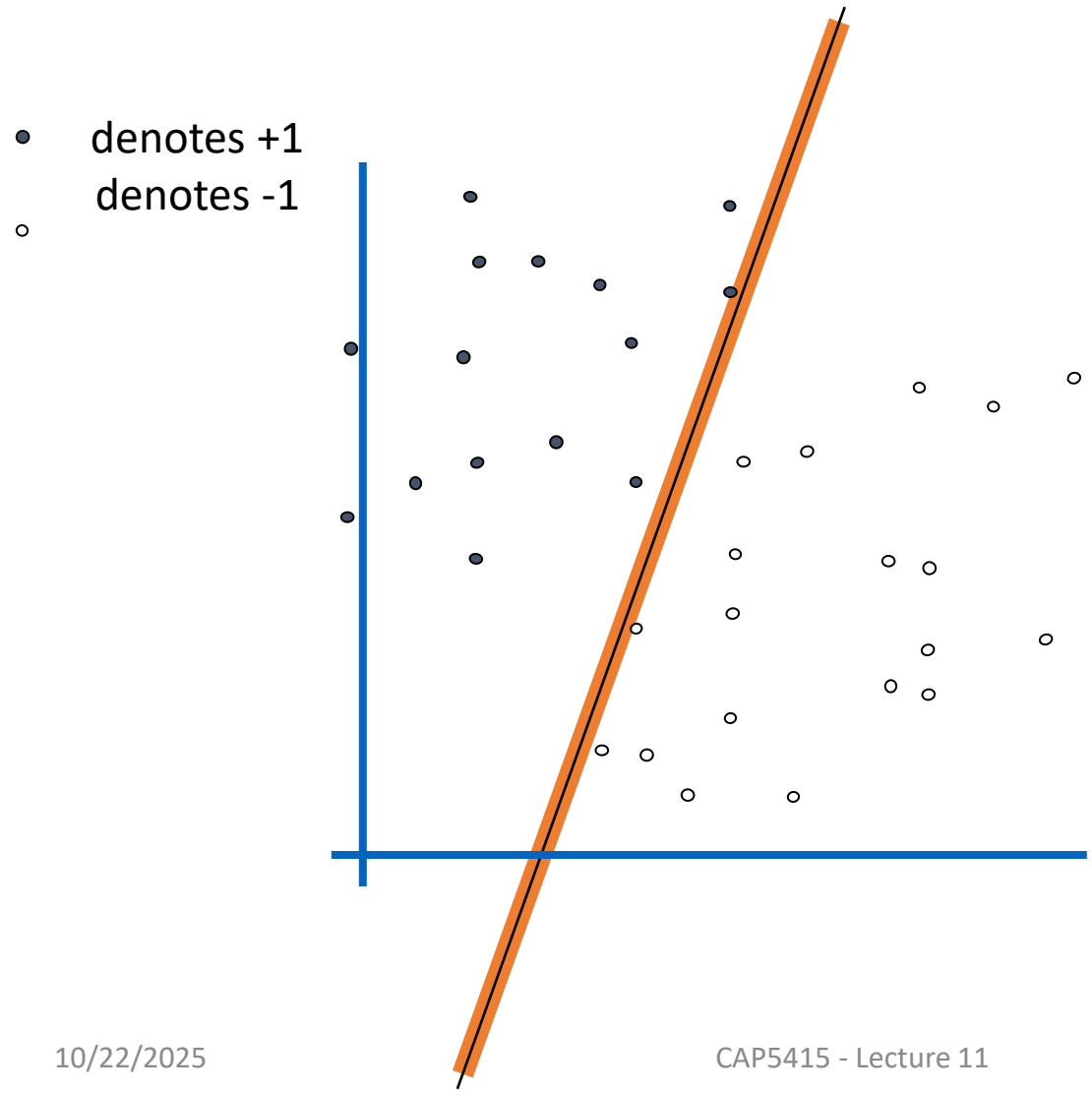
denotes +1
denotes -1



Any of these would be fine..

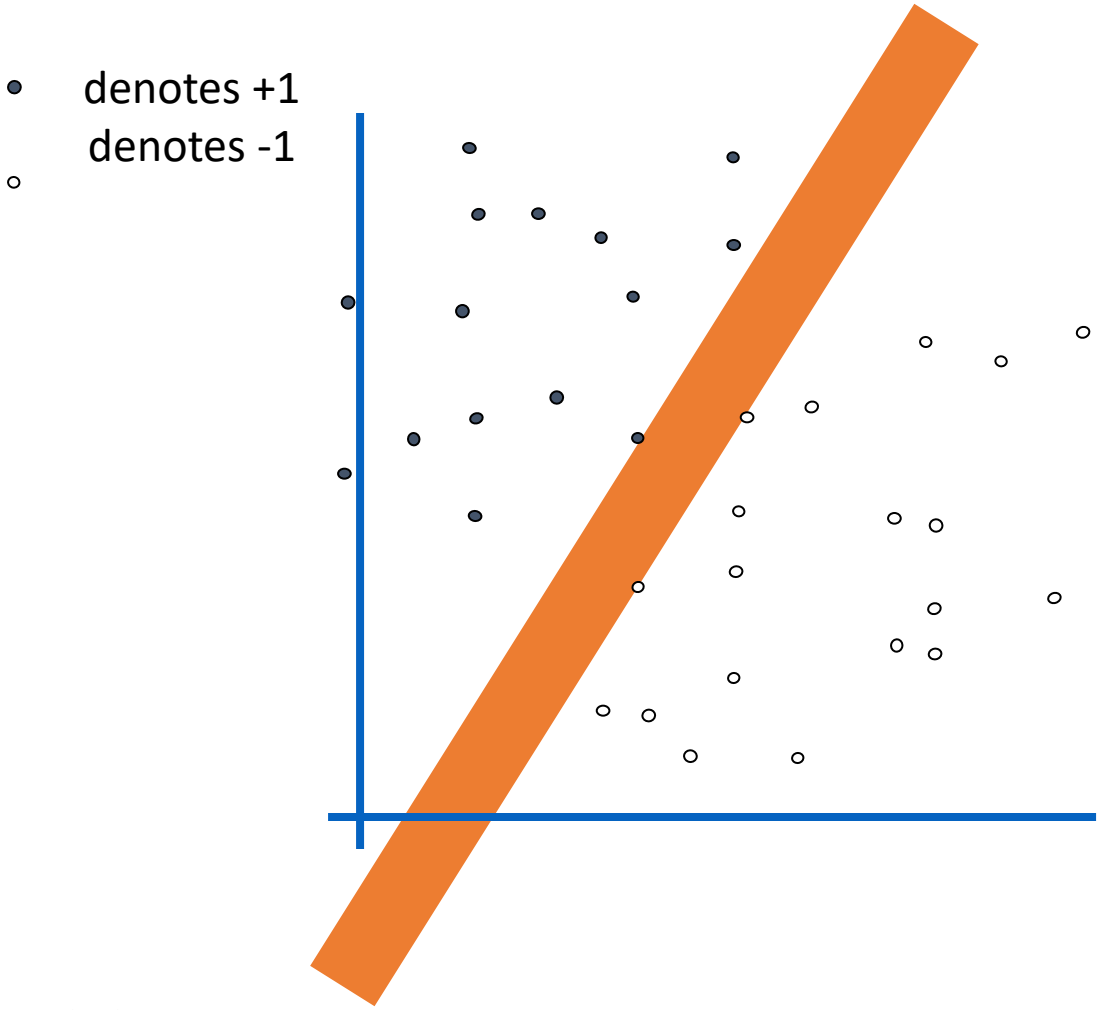
..but which is best?

Classifier Design



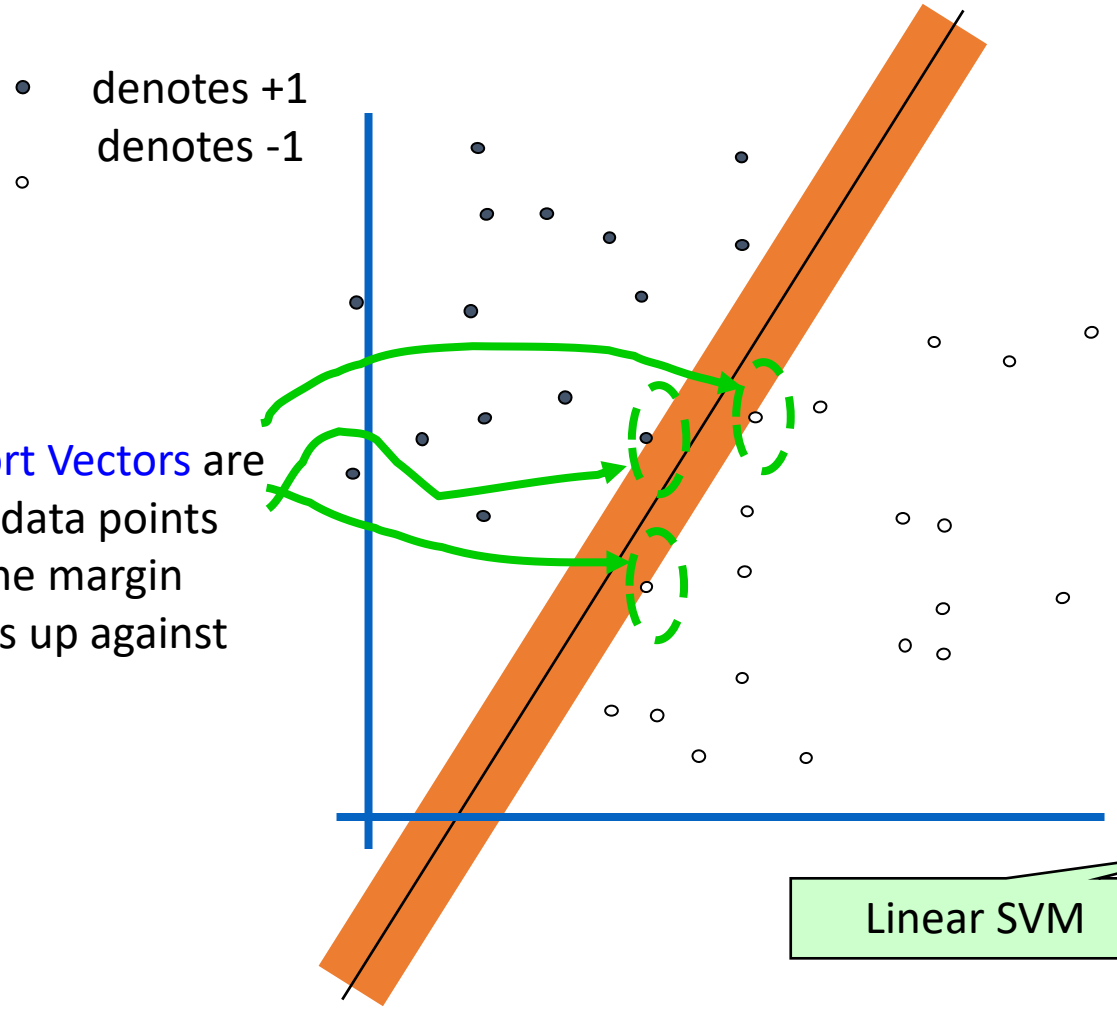
Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



The maximum margin linear classifier is the linear classifier with the, maximum margin.
This is the simplest kind of SVM (Called an LSVM)

Maximum Margin



The maximum margin linear classifier is the linear classifier with the, maximum margin.
This is the simplest kind of SVM (Called an LSVM)

SVM - background

- A supervised approach for classification and regression

SVM - background

- A supervised approach for classification and regression
  - Developed in the computer science society at 1990s, has grown in popularity since then.

SVM - background

- A supervised approach for classification and regression
  - Developed in the computer science society at 1990s
  - Has grown in popularity since then
  - Shown to perform well in variety of settings
  - And often considered as one of the best “out-of-the box” classifiers

Neural networks were a big deal in early 90s

Kind of delayed Deep Learning era ☺

SVM - background

- An approach for classification
  - Developed in the computer science society at 1990s, has grown in popularity since then.
  - Shown to perform well in variety of settings, and often considered as one of the best “out-of-the box” classifiers.



(historical appearances in the literature)

Maximal-Margin Classifier

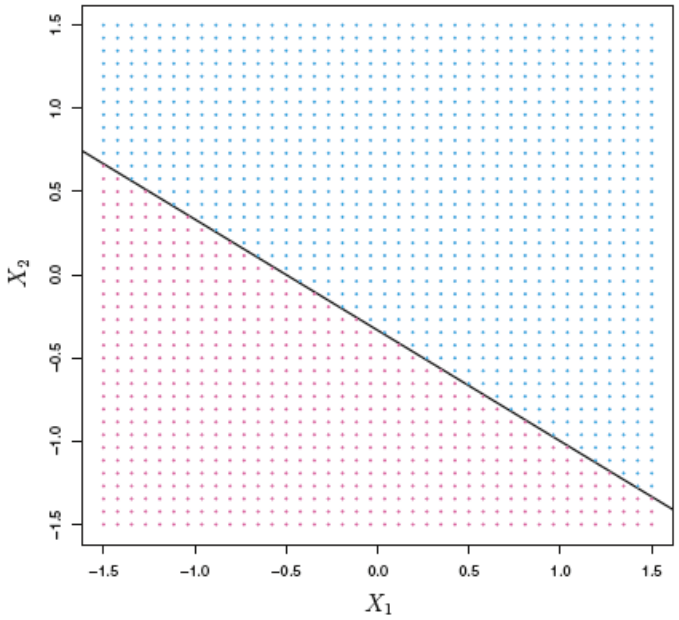
- In a *p*-dimensional space, a hyperplane is a flat affine subspace of dimension *p-1.*

Maximal-Margin Classifier

- In a *p*-dimensional space, a hyperplane is a flat affine subspace of dimension *p-1.*
  - Ex. In 2D, a hyperplane is 1D line,
  - Ex. In 3D, a hyperplane is 2D plane.

Maximal-Margin Classifier

- In a *p*-dimensional space, a hyperplane is a flat affine subspace of dimension *p-1.*
  - Ex. In 2D, a hyperplane is 1D line,
  - Ex. In 3D, a hyperplane is 2D plane.

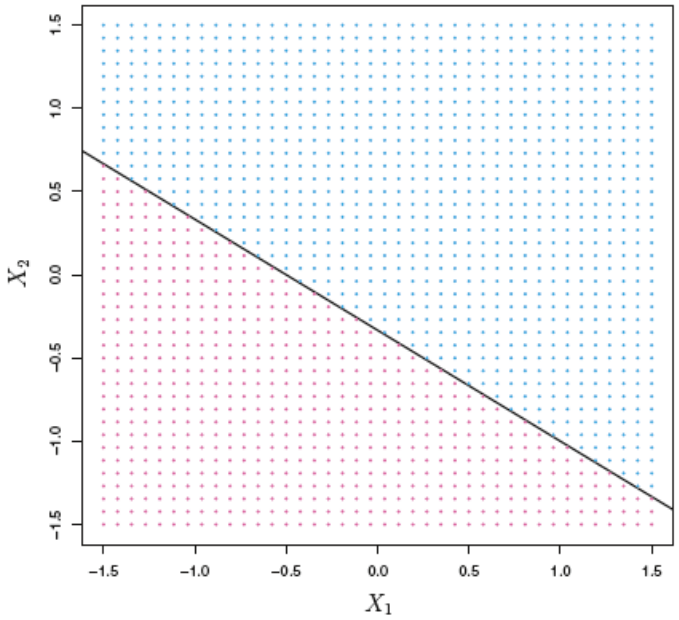


General hyperplane definition

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ...\beta_p X_p = 0$$

Maximal-Margin Classifier

- In a *p*-dimensional space, a hyperplane is a flat affine subspace of dimension *p-1.*
  - Ex. In 2D, a hyperplane is 1D line,
  - Ex. In 3D, a hyperplane is 2D plane.



General hyperplane definition

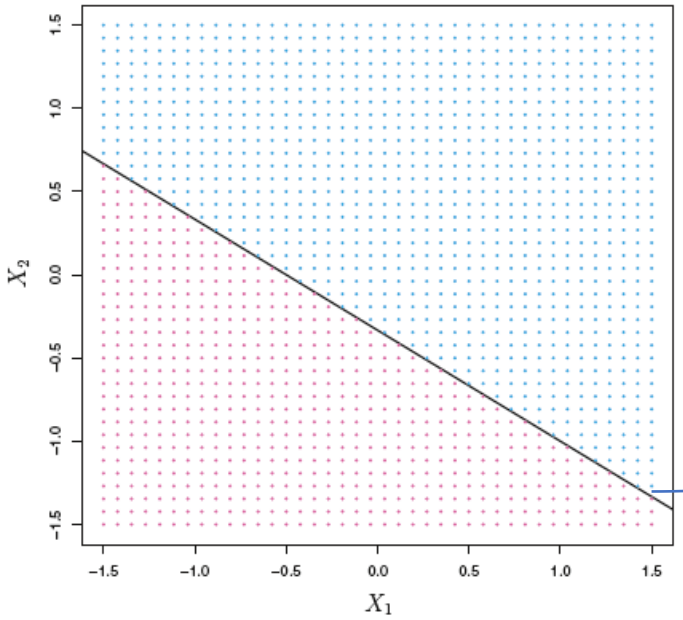
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ...\beta_p X_p = 0$$

Hyperplane for 2D data.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

Maximal-Margin Classifier

- In a *p*-dimensional space, a hyperplane is a flat affine subspace of dimension *p-1.*
  - Ex. In 2D, a hyperplane is 1D line,
  - Ex. In 3D, a hyperplane is 2D plane.



General hyperplane definition

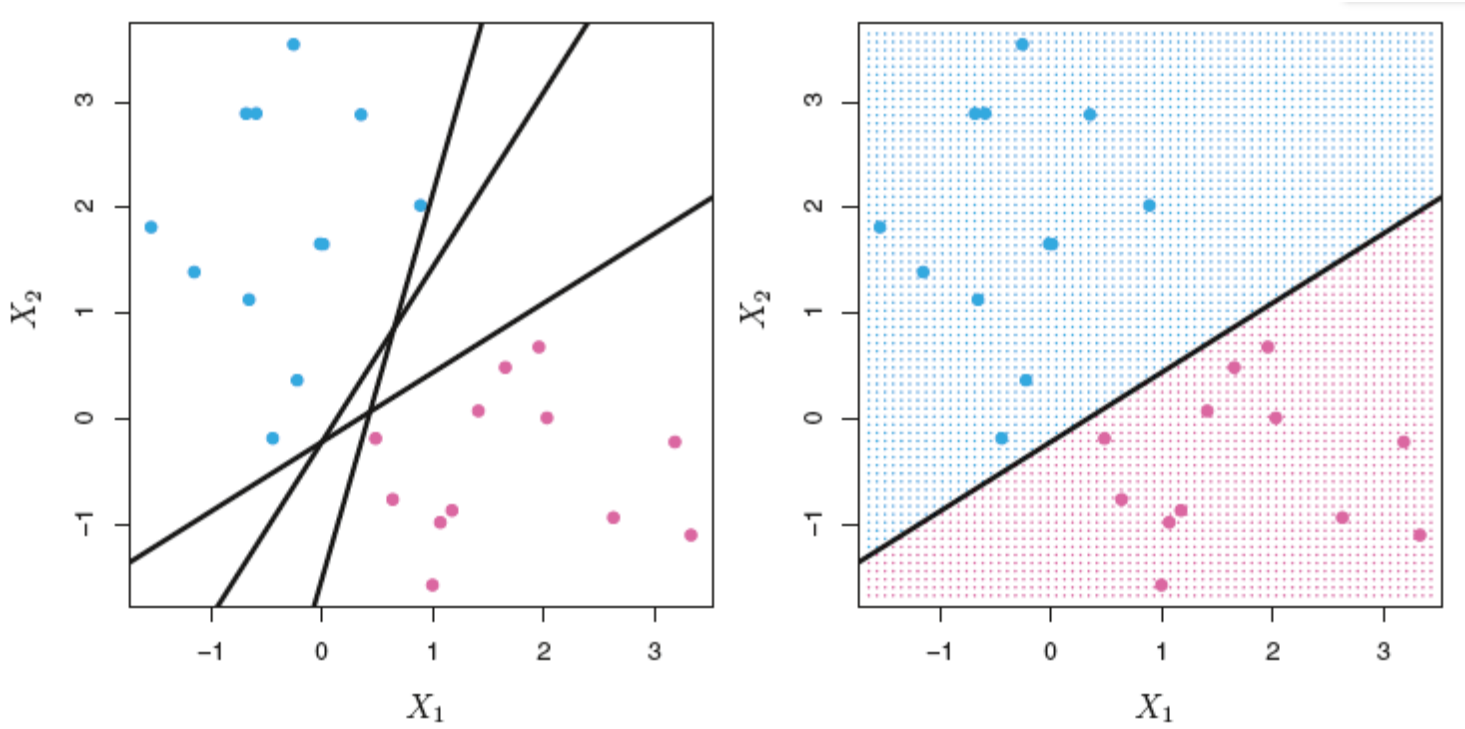
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ...\beta_p X_p = 0$$

Hyperplane for 2D data.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

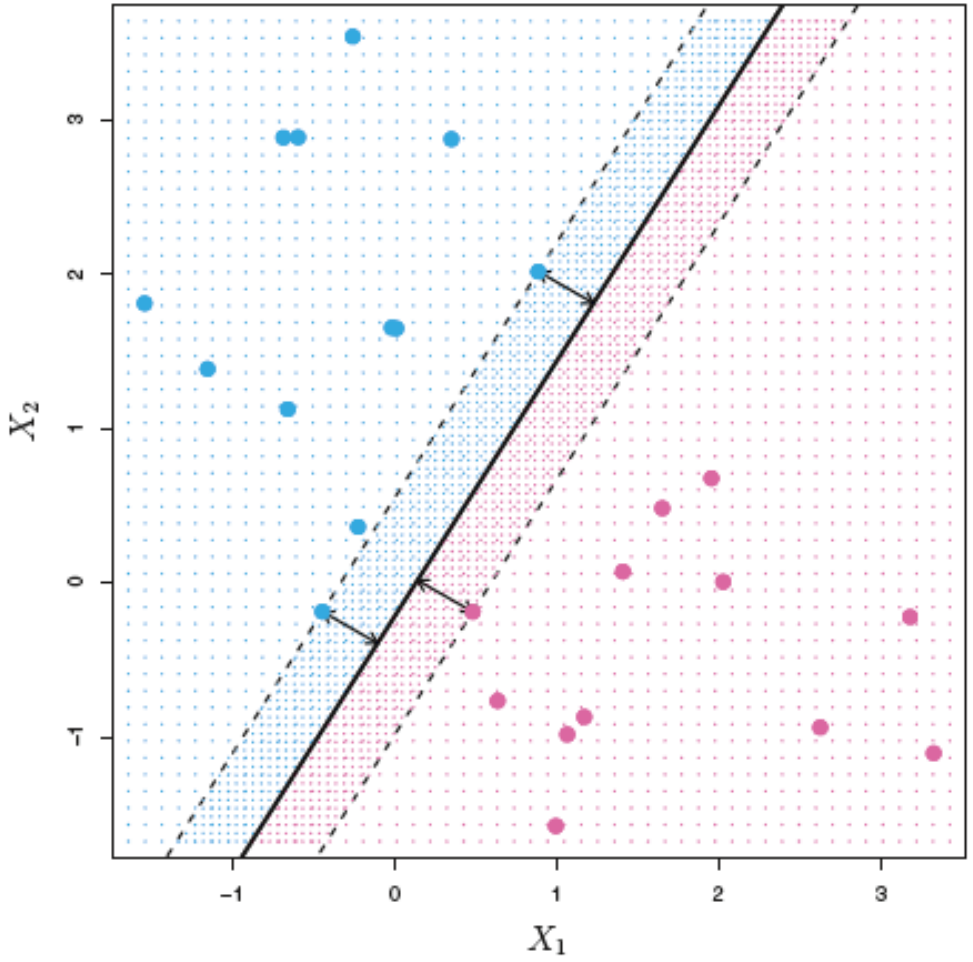
$$1 + 2X_1 + 3X_2 = 0$$

Max-margin classifier



Left: There are two classes of observations: blue and in purple (each of which has measurements on two variables). Three separating hyperplanes, out of many possible, are shown in black. Right: A separating hyperplane is shown in black: a test observation that falls in the blue portion of the grid will be assigned to the blue class, and a test observation that falls into the purple portion of the grid will be assigned to the purple class.

Max-margin classifier



There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line.

The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support vectors, and the distance from those points to the margin is indicated by arrows.

The purple and blue grid indicates the decision rule made by a classifier based on this separating hyperplane.

Construction of Max-Margin Classifier

- Consider n training observations $x_1, \ldots, x_n \in \mathbb{R}^p$

Construction of Max-Margin Classifier

- Consider n training observations $x_1, \ldots, x_n \in \mathbb{R}^p$
- Associated class labels $y_1, \ldots, y_n \in \{-1, +1\}$

Construction of Max-Margin Classifier

- Consider n training observations $x_1, \ldots, x_n \in \mathbb{R}^p$
- Associated class labels $y_1, \ldots, y_n \in \{-1, +1\}$
- Max-margin hyperplane is the solution to:

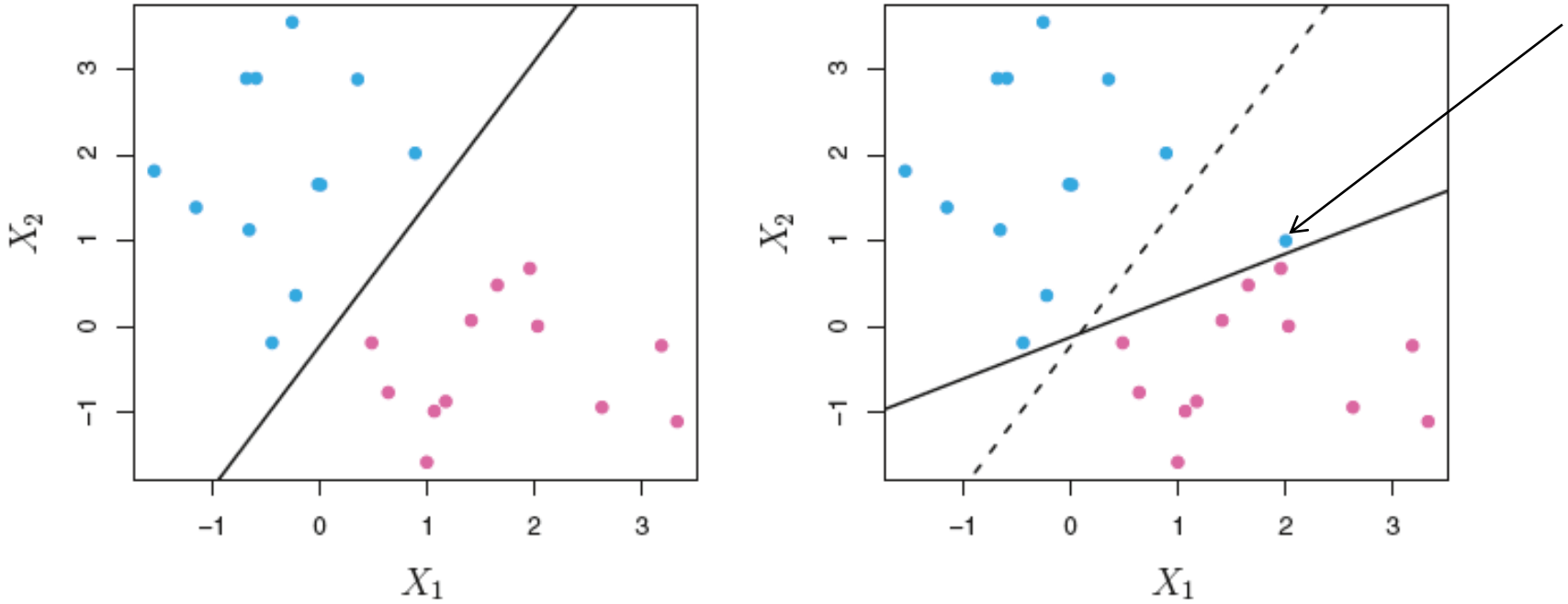
$$\underset{\beta_0, \beta_1, \ldots, \beta_p}{\text{maximize}}\, M$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

- It ensures that each observation is on the correct side
  - at least a distance M from the hyperplane

Is Max-Margin Robust ?



Left: Two classes of observations are shown in blue and purple, along with the maximal margin hyperplane.

Right: An additional blue observation has been added, leading to a dramatic shift in the maximal margin hyperplane shown as a solid line. The dashed line indicates the maximal Margin hyperplane that was obtained in the absence of this additional point.
Max-margin classifier is sensitive to individual observations!

CAP5415 Computer Vision

Yogesh S Rawat

yogesh@ucf.edu

HEC-241

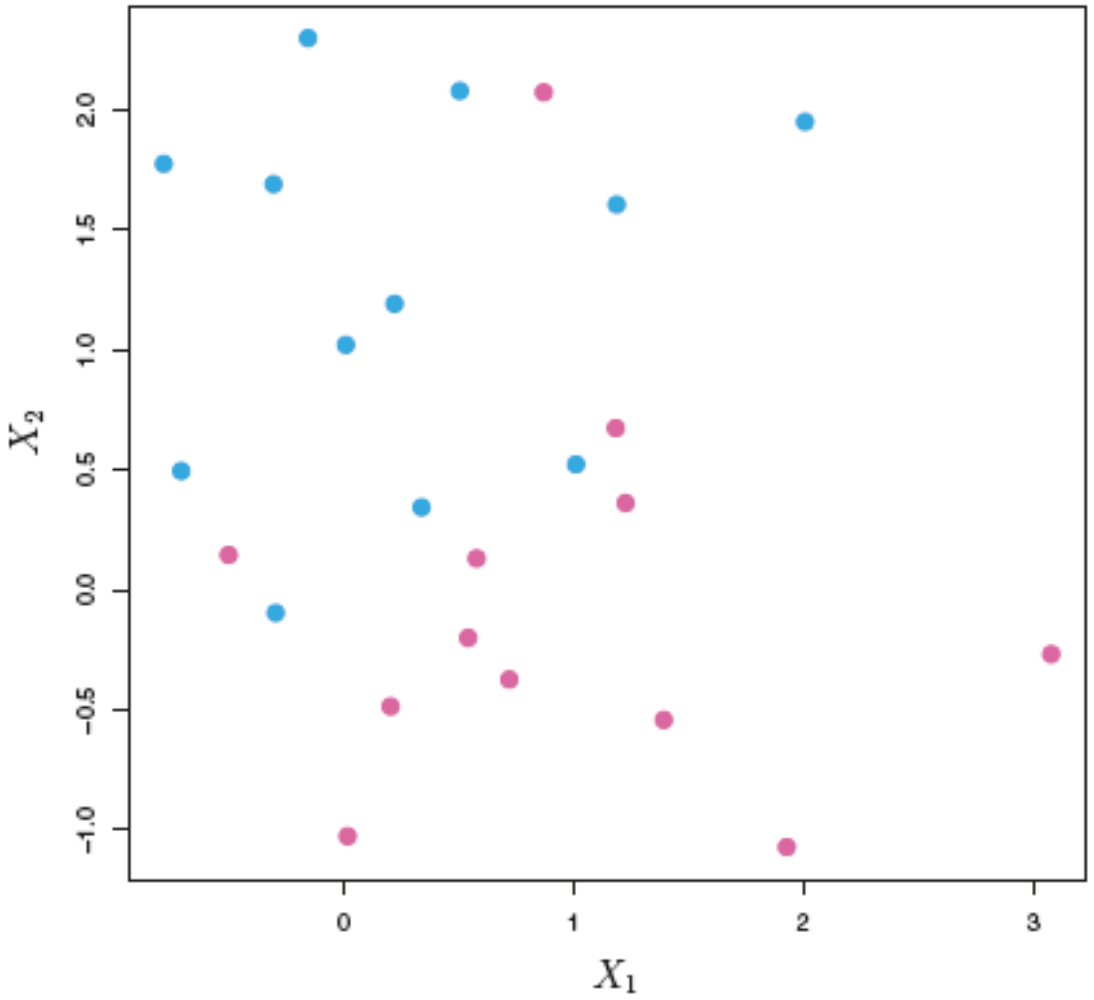
Questions?

Classification - I

Lecture 11

Support vector machine

Non-Separable Case



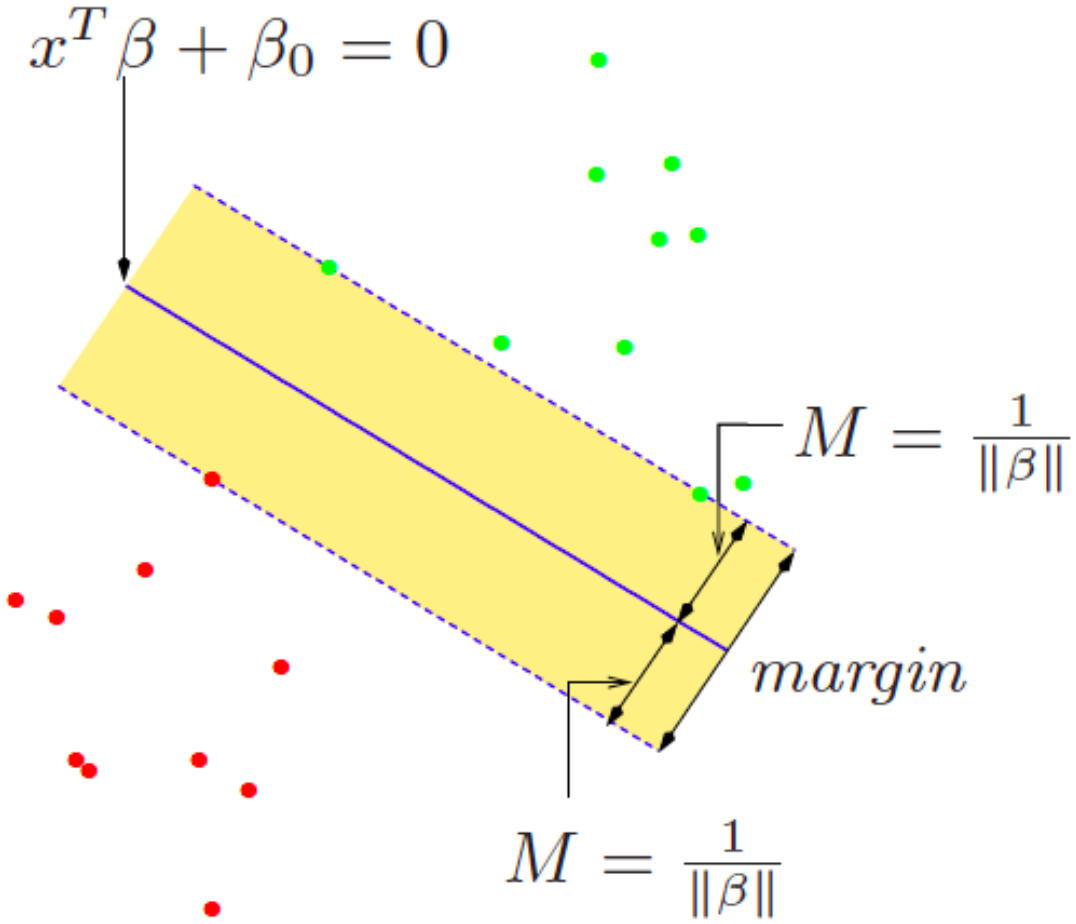
The maximal margin classifier is a very natural way to perform Classification, if a separating hyperplane exists .

In many cases there is no separating hyperplane exists!

In this case, we cannot exactly separate the two classes.(However, Notice soft margin in the following slides!)

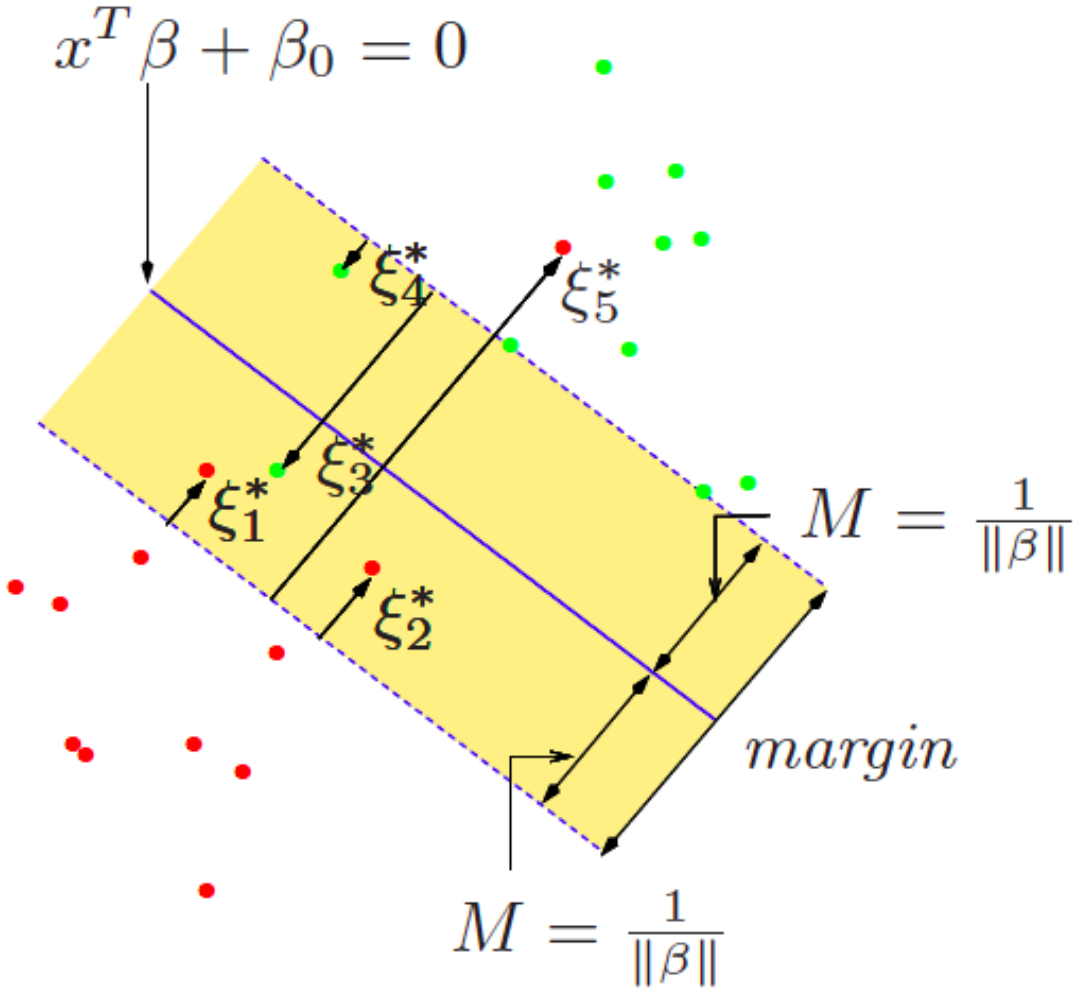
The generalization of the maximal margin classifier to the non-separable case is known as the support vector classifier .

Support Vector Classifier-Separable



- Separable plane is shown.
- Decision boundary is the solid line.
- Broken lines bound the Shaded maximal margin of 2M.

Support Vector Classifier-Overlap



- Overlap case is shown.
- The points labeled ξi are on the wrong side of the Margin.
- The margin is maximized subject to a total budget

$$\sum \xi_i \leq \text{ constant}$$

Soft-Margin ~ Support Vector Classifier

- Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane.

Soft-Margin ~ Support Vector Classifier

- Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane.

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

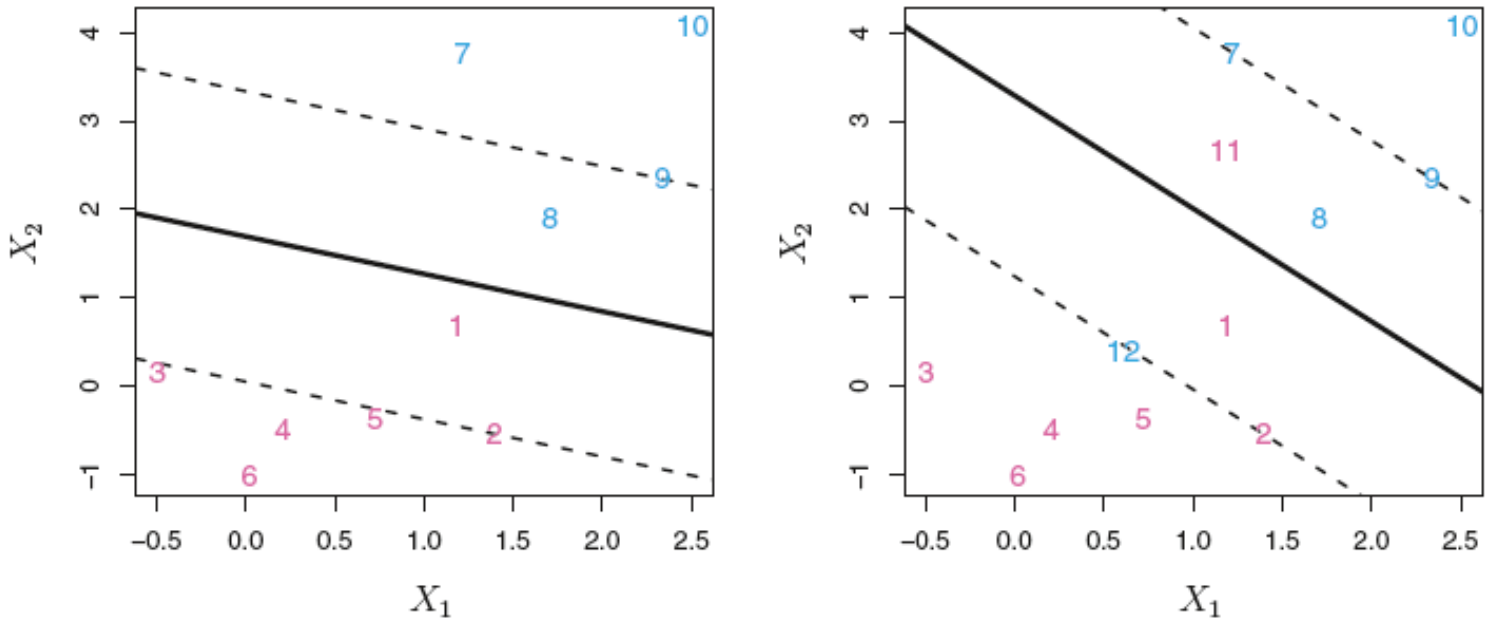
$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

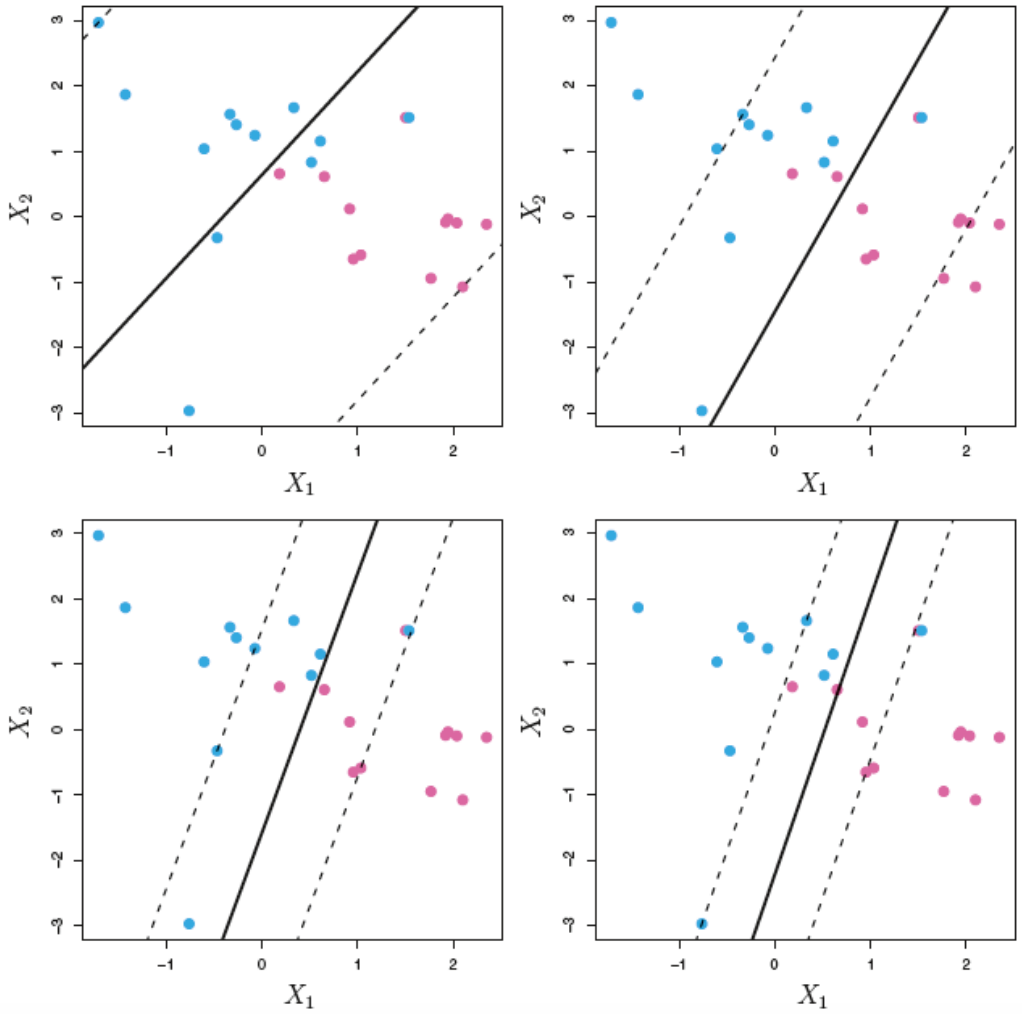
- C: nonnegative tuning params
- eps: slack variables allowing obs. to be on the wrong side of the margin.

SV Classifier



Left: A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines. Purple observations: Observations 3, 4, 5, and 6 are on the correct side of the margin, observation 2 is on the margin, and observation 1 is on the wrong side of the margin. Blue observations: Observations 7 and 10 are on the correct side of the margin, observation 9 is on the margin, and observation 8 is on the wrong side
of the margin. No observations are on the wrong side of the hyperplane. Right: Two additional points, 11 and 12 are added.

SV Classifier

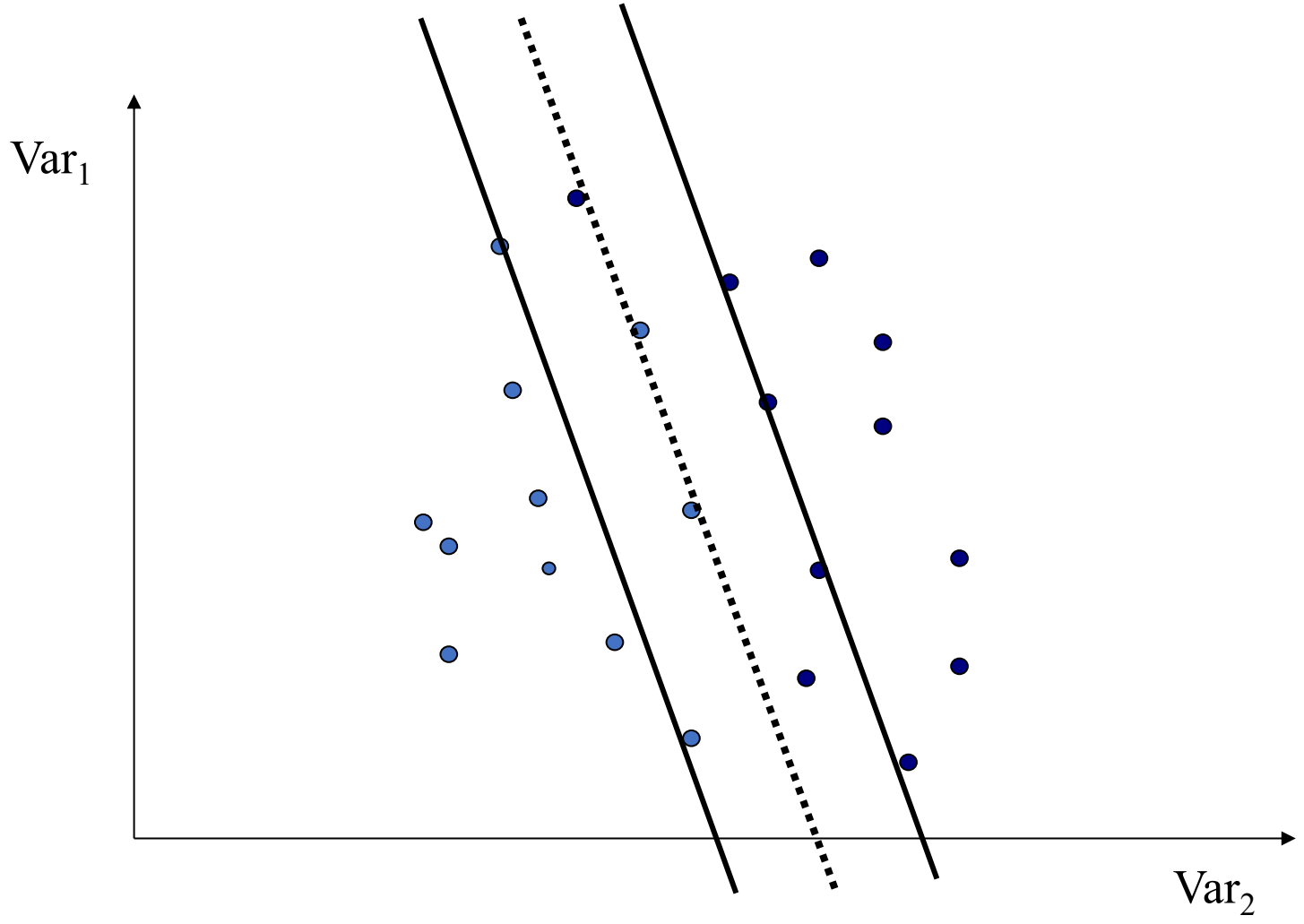


Four different tuning param C
Were used to fit SVM into
Small number of data.

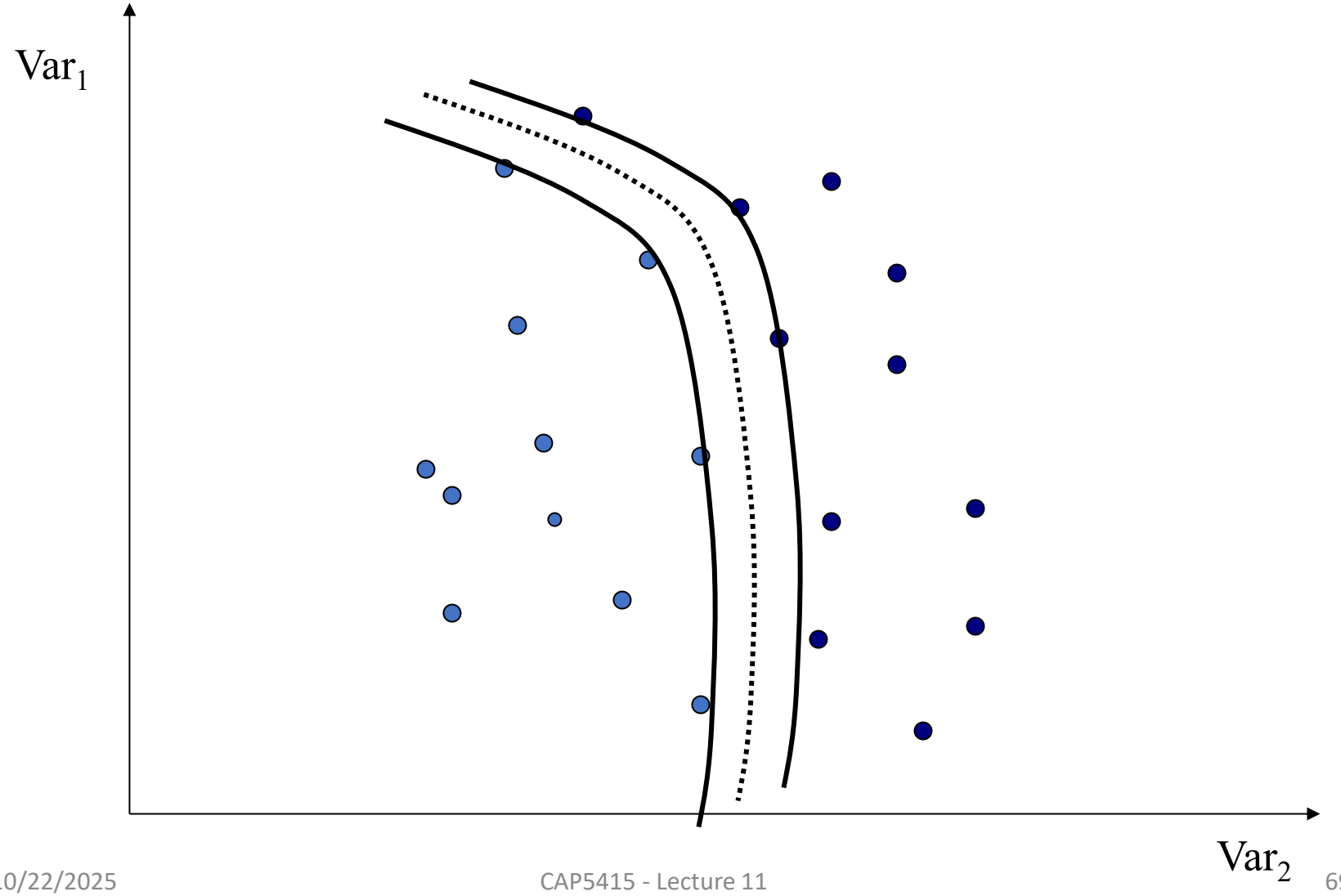
The largest value of C was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels.

When C is large, then there is a high tolerance for observations Being on the wrong side of the margin, and so the margin will be large. As C decreases, the tolerance for observations being on the wrong side of the margin decreases, and the margin narrows.

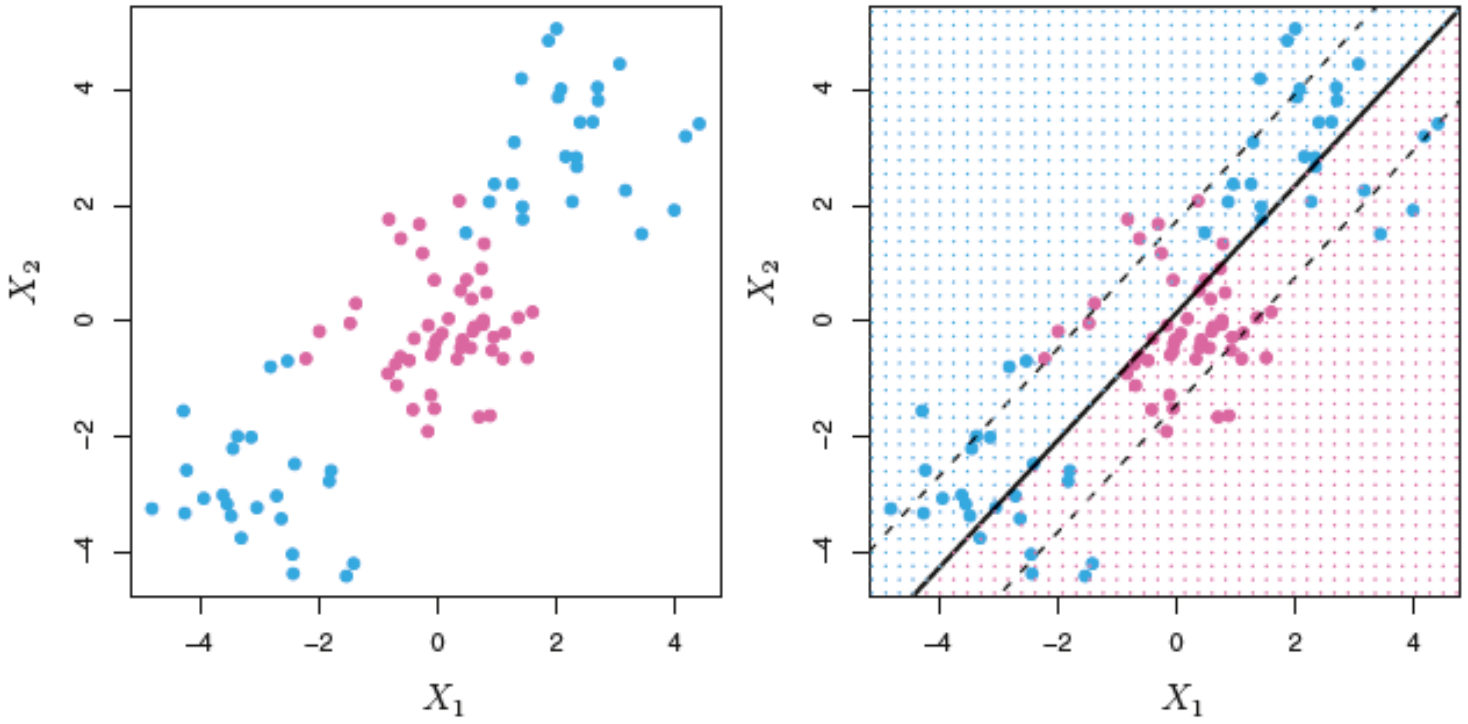
Disadvantages of Linear Decision Surfaces



Advantages of non-linear Decision Surfaces



SV Classifier



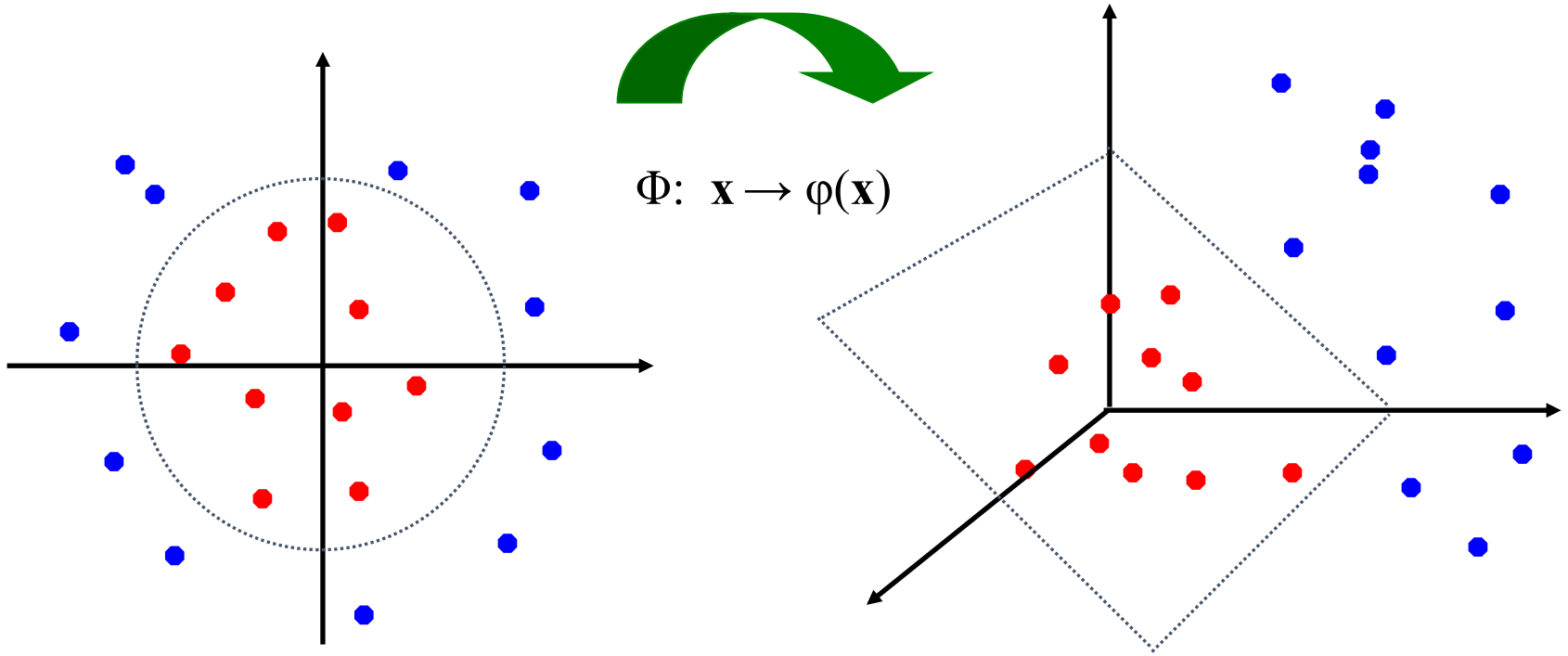
In practice we are sometimes faced with non-linear class boundaries. Support vector classifier or any linear classifier will perform poorly here

Left: The observations fall into two classes, with a non-linear boundary between them.

Right: The support vector classifier seeks a linear boundary, and consequently performs very poorly.

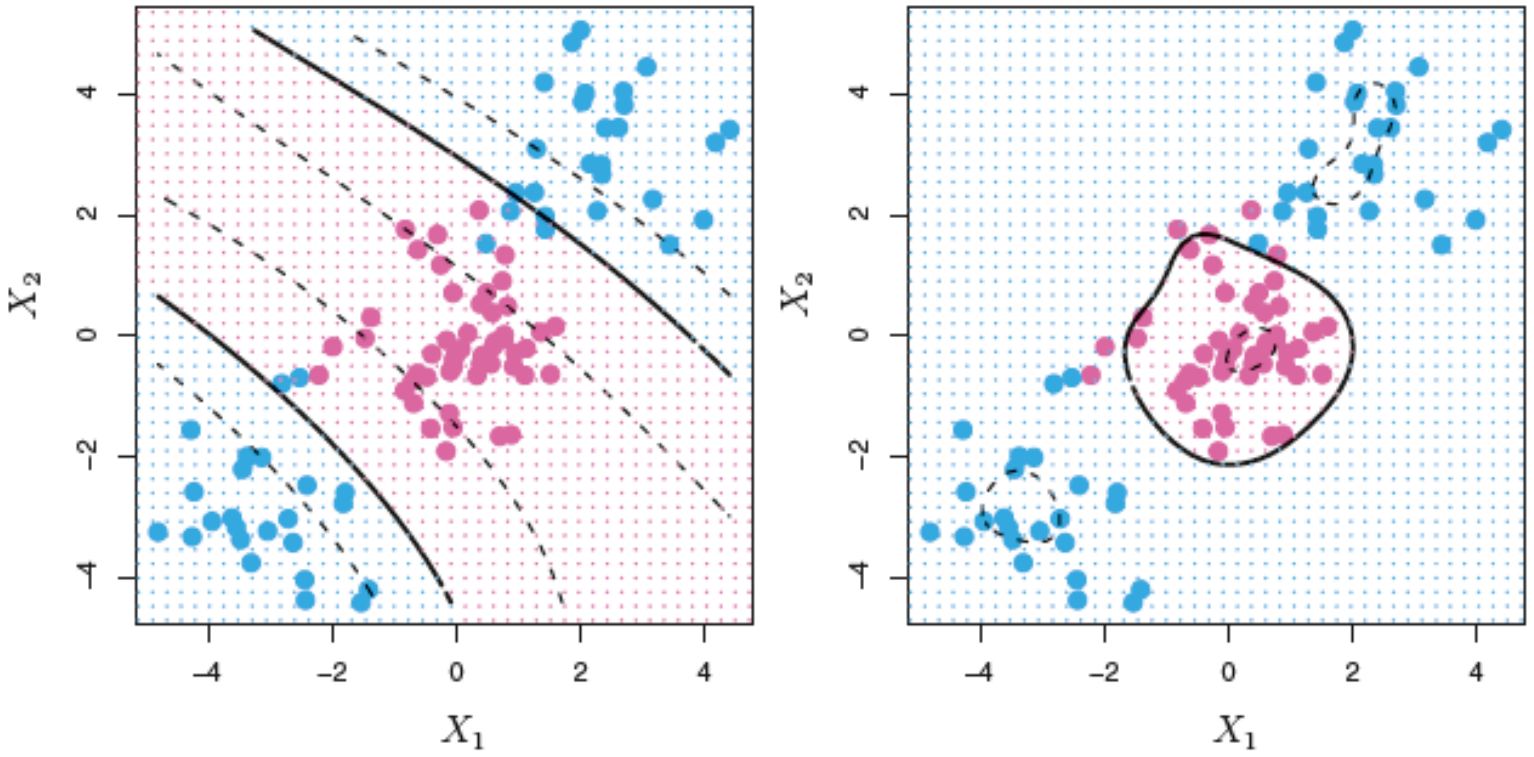
Nonlinear SVMs

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



Slide credit: Andrew Moore

SVM Classifier



Left: An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from previous slides, resulting in a far more appropriate decision rule.

Right: An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.

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

Questions?

Sources for this lecture include materials from works by Abhijit Mahalanobis, James Tompkin, and Ulas Bagci