DS 5220

Supervised Machine Learning and Learning Theory

Alina Oprea
Associate Professor, CCIS
Northeastern University

November 25 2019

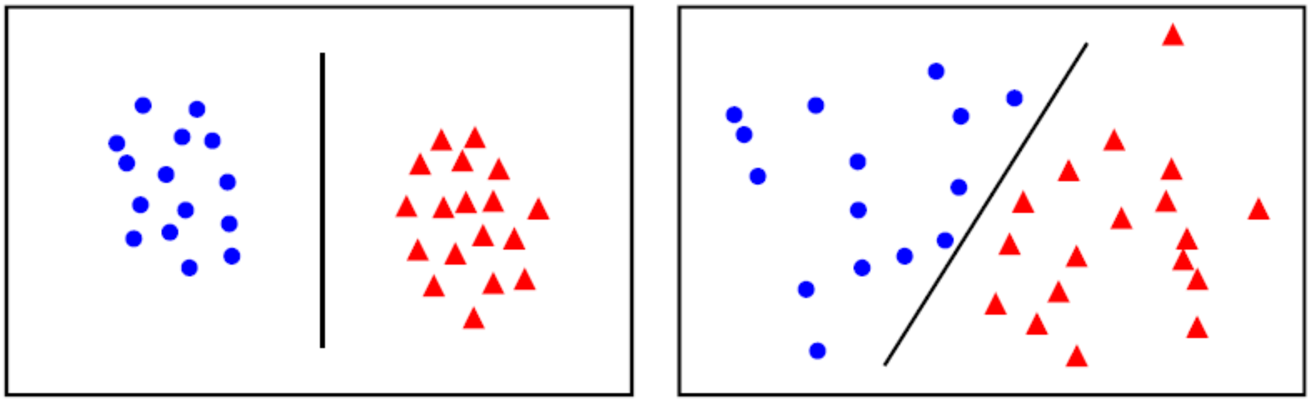
Logistics

- Project milestone
  - Feedback in Gradescope
- Homework 4 due tomorrow
- Final exam
  - Next Wednesday (Dec 4) in class, 2 hours
  - Review next Monday, Dec. 2
- Final project
  - Presentation on Monday, Dec. 9, 1-5pm, ISEC 655
  - Final report due on Tuesday, Dec. 10

Outline

- Review of linear models
  - Separating hyperplanes
- Support Vector Machines
  - Linearly separable data
    - Maximum margin classifier
  - Non-separable data
    - Support vector classifier
  - Non-linear decision boundaries
    - Kernels and Radial SVM

Linear models we've seen



Classifiers with linear decision boundary:

- Perceptron
- Logistic regression
- Linear discriminant analysis
- Today: support vector classifier

Hyperplane

- Line (2-dimensions): $\theta_0 + \theta_1 x_1 + \theta_2 x_2 = 0$
- Hyperplane (d-dimensions): $\theta_0 + \theta_1 x_1 + \cdots \theta_d x_d = 0$

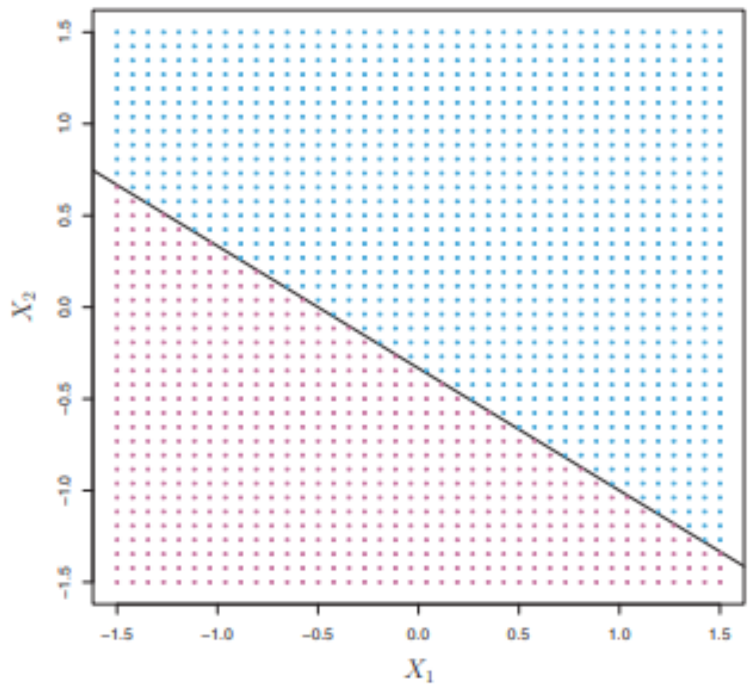


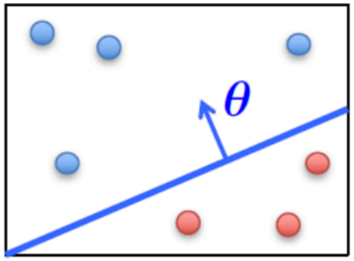
FIGURE 9.1. *The hyperplane $1 + 2X_1 + 3X_2 = 0$ is shown. The blue region is the set of points for which $1 + 2X_1 + 3X_2 > 0$, and the purple region is the set of points for which $1 + 2X_1 + 3X_2 < 0$.*

Recall:

Linear classifiers

- **Linear classifiers**: represent decision boundary by hyperplane

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \quad \boldsymbol{x}^\intercal = \begin{bmatrix} 1 & x_1 & \dots & x_d \end{bmatrix}$$



All the points x on the hyperplane satisfy: $\theta^T x = 0$

$$h(\boldsymbol{x}) = \text{sign}(\boldsymbol{\theta}^\intercal \boldsymbol{x}) \quad \text{where} \quad \text{sign}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases}$$

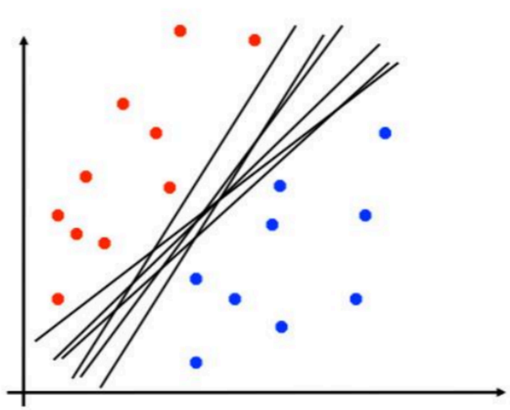
– Note that: $\boldsymbol{\theta}^\intercal \boldsymbol{x} > 0 \implies y = +1$

$\boldsymbol{\theta}^\intercal \boldsymbol{x} < 0 \implies y = -1$

Recall:

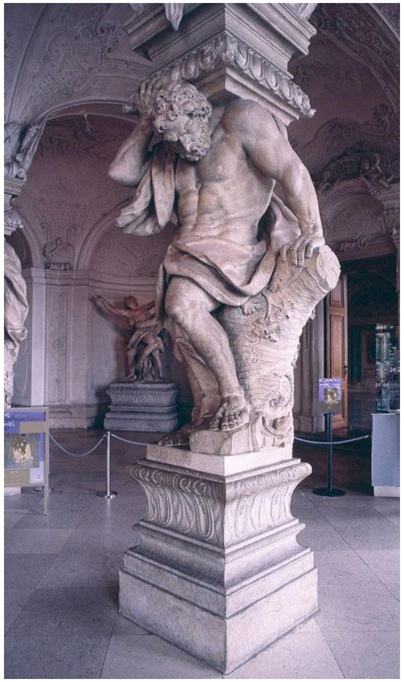
Perceptron Limitations

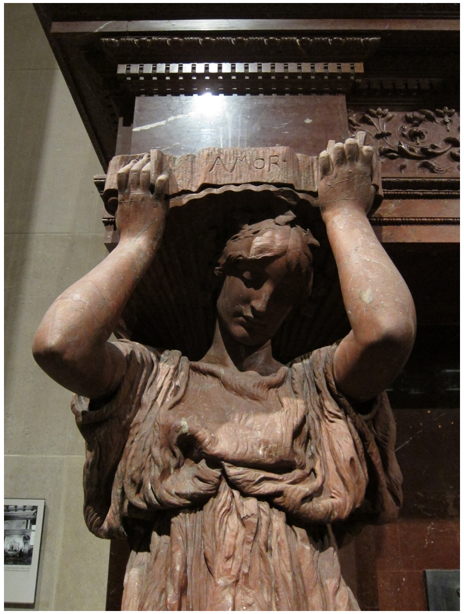
- Is dependent on starting point
- It could take many steps for convergence
- Perceptron can overfit
  - Move the decision boundary for every example



Which of this is optimal?

Support vectors



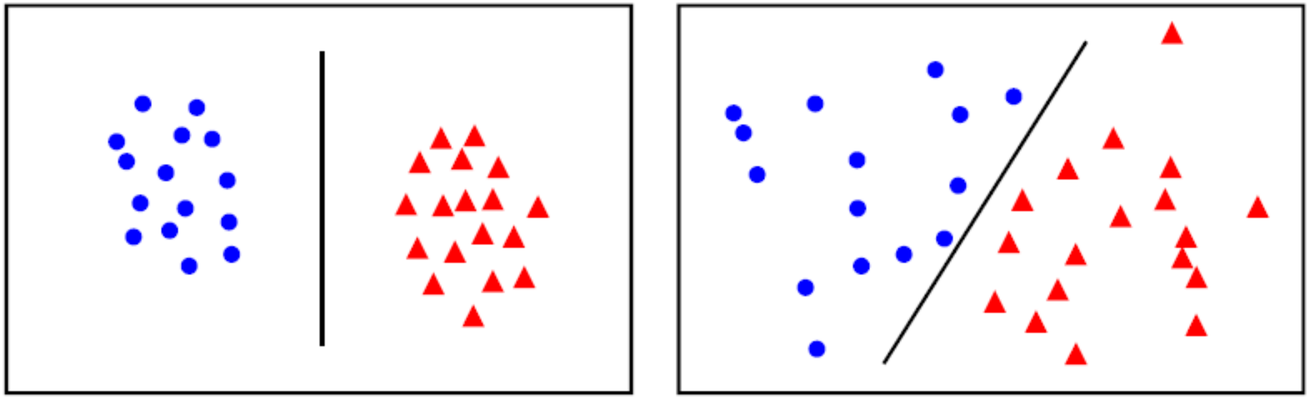


Outline

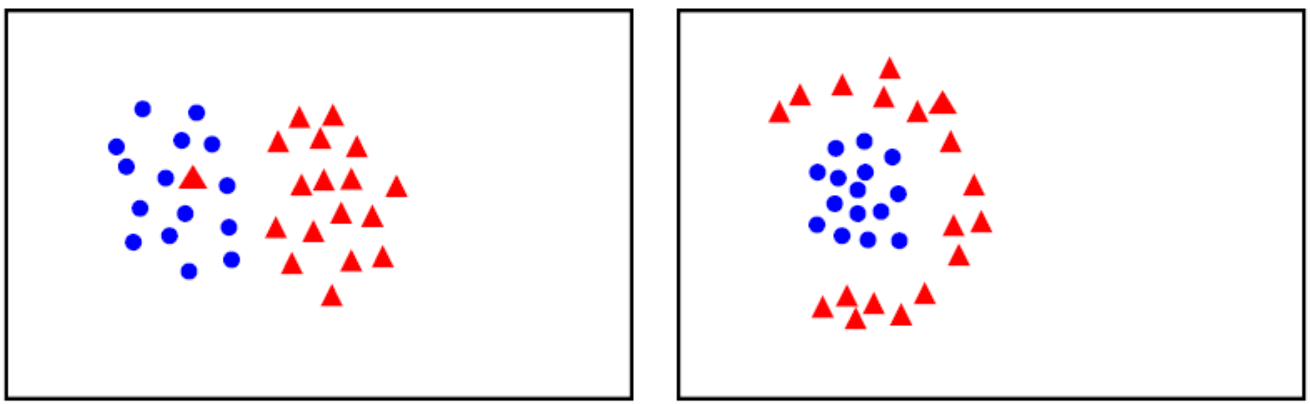
- Review of linear classifiers
  - Separating hyperplanes
- Support Vector Machines
  - Linearly separable data
    - Maximum margin classifier
  - Non-separable data
    - Support vector classifier
  - Non-linear decision boundaries
    - Kernels and Radial SVM

Linear separability

linearly separable



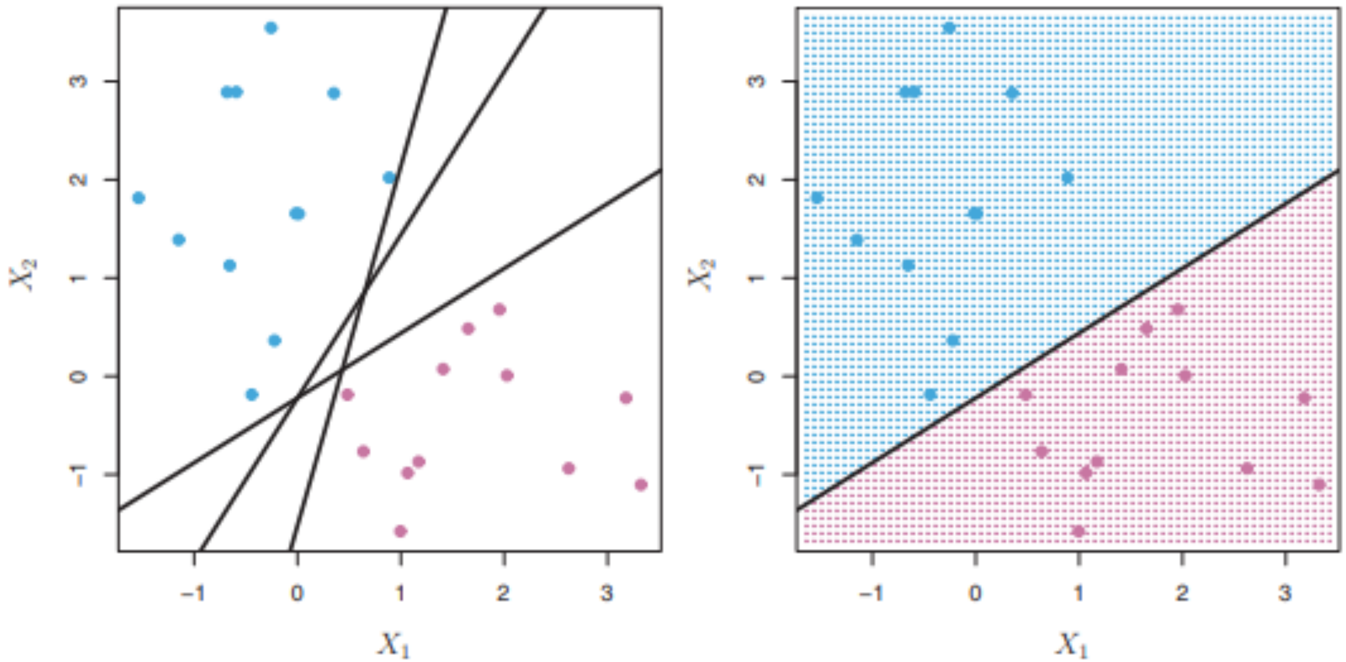
not linearly separable



Notation (supervised learning)

- Training data $x_1, \dots, x_N$ with $x_i = \left(x_{i1}, \dots, x_{id}\right)^{\mathrm{T}}$
- Labels are from 2 classes: $y_i \in \{-1,1\}$
- Goal:
  - Build a model to classify training data
  - Test it on new vector x' = $(x'_1, \dots, x'_d)^T$ to predict label y'

Separating hyperplane



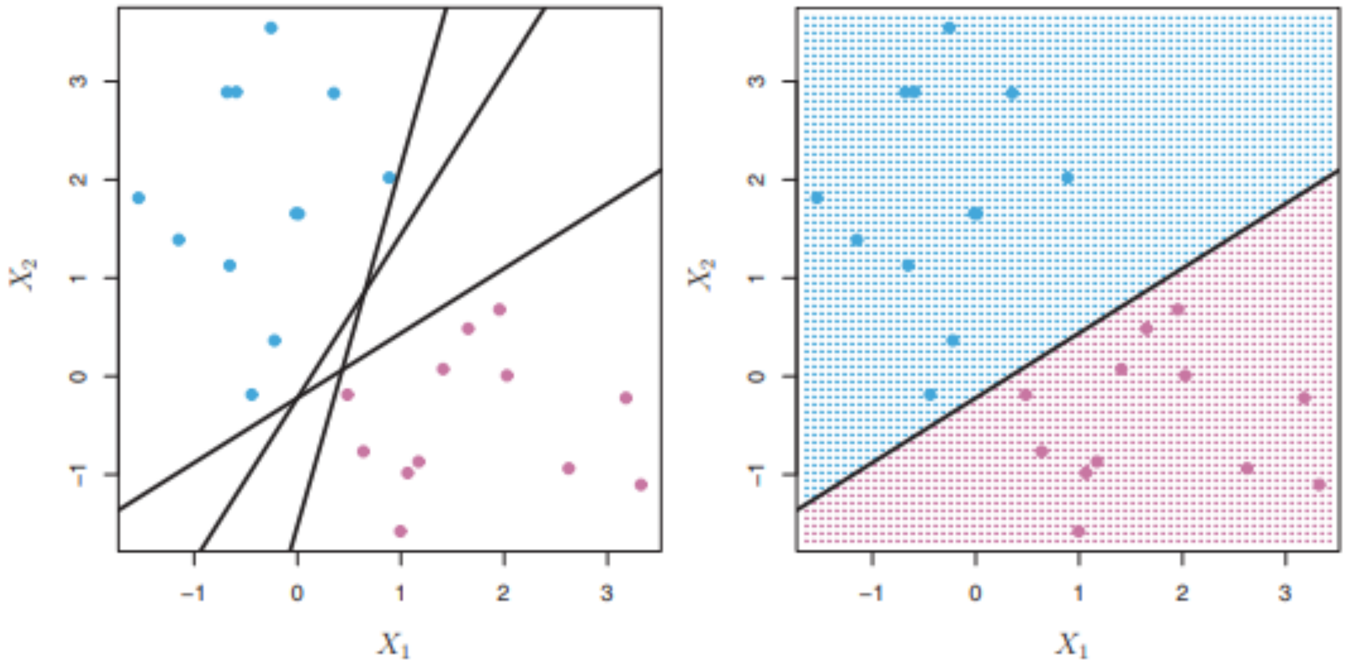
$$\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id} > 0 \text{ if } y_i = 1$$

$$\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id} < 0 \text{ if } y_i = -1$$

For all training data x_i, y_i
$i \in \{1, \ldots, N\}$

Perfect separation between the 2 classes

Separating hyperplane



$$y_i(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}) > 0$$

For all training data x_i, y_i, $i \in \{1, \ldots, N\}$

From separating hyperplane to classifier

- Training data $x_1, \dots, x_N$ with $x_i = \left(x_{i1}, \dots, x_{id}\right)^{\mathrm{T}}$
- Labels are from 2 classes: $y_i \in \{-1,1\}$
- Let $\theta_0, \dots, \theta_d$ (will be learned) such that:

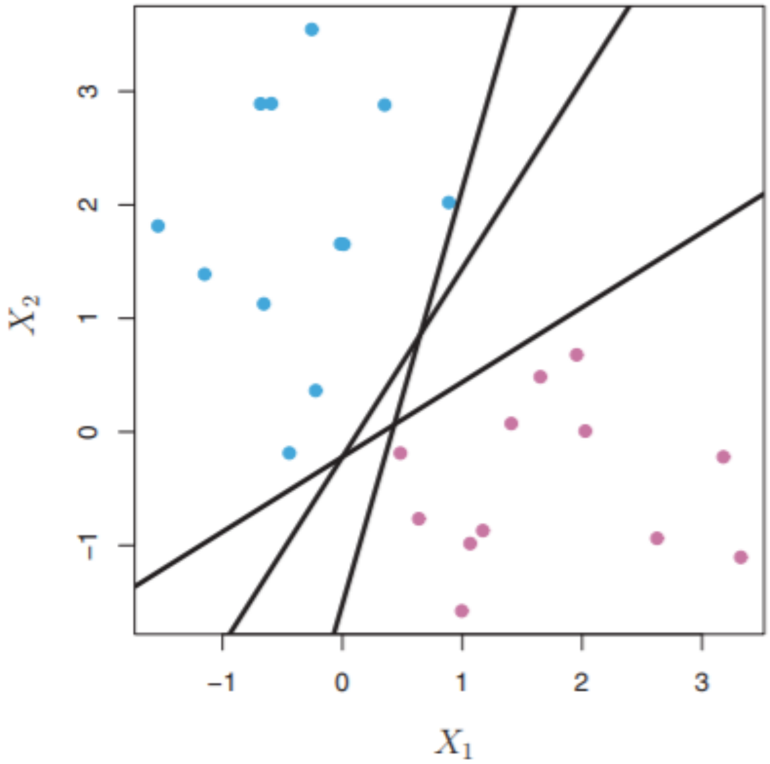
$$y_i(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}) > 0$$

- Classifier

$$f(z) = \text{sign}(\theta_0 + \theta_1 z_1 + \cdots \theta_d z_d) = \text{sign}(\theta^T \text{z})$$

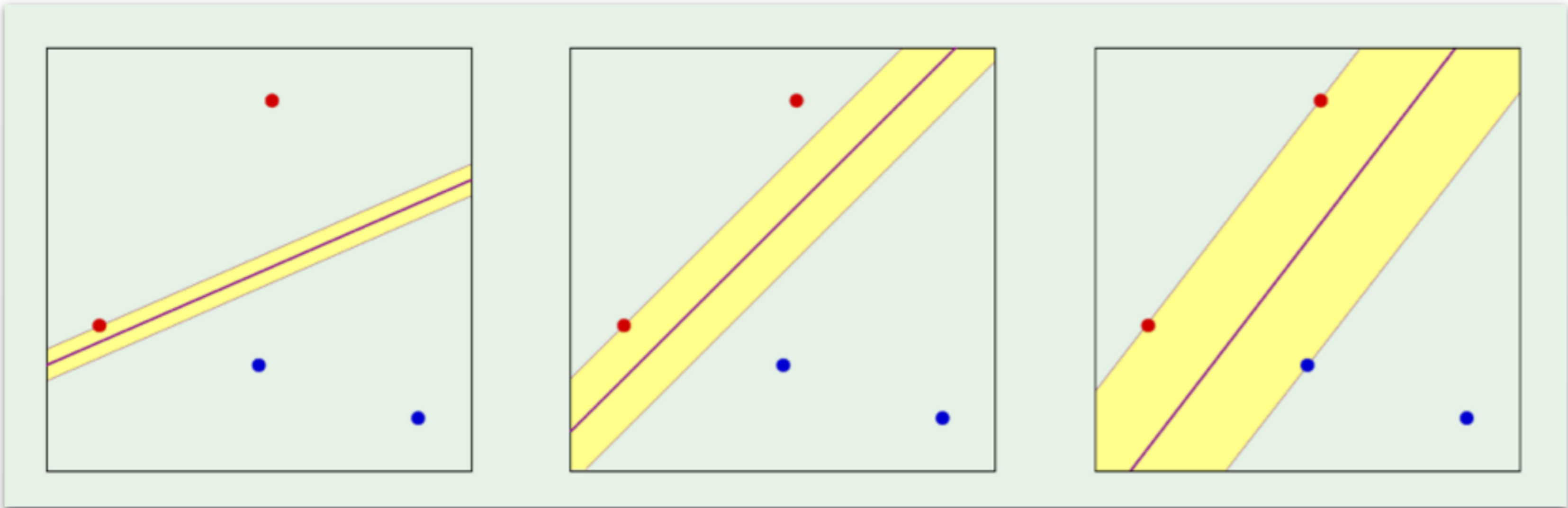
- Classify new test point x'
  - If $f(x') > 0$ predict y'$= 1$
  - Otherwise predict y'$= -1$

Separating hyperplane



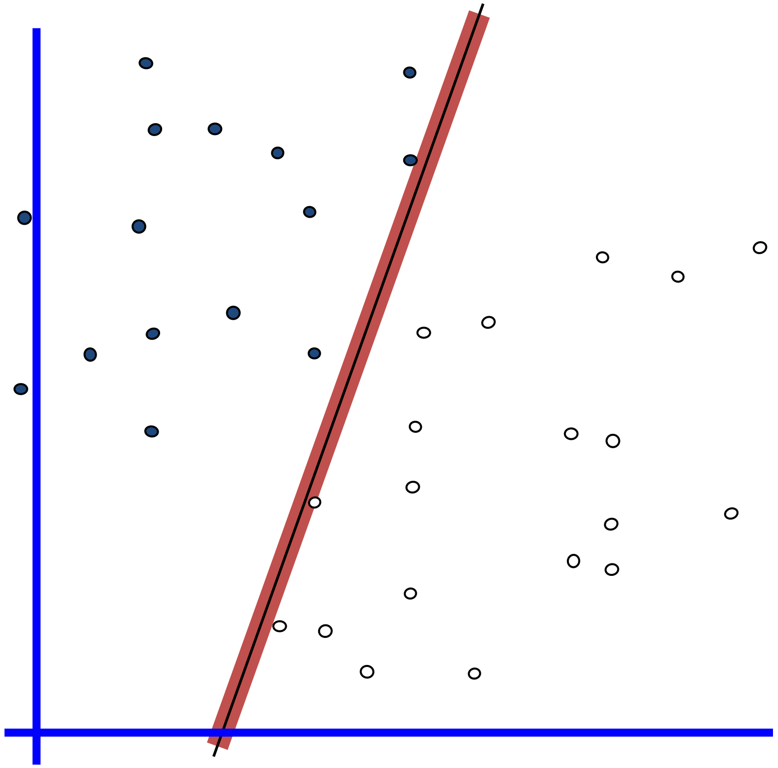
- If a separating hyperplane exists, there are infinitely many
- Which one should we choose?

Intuition



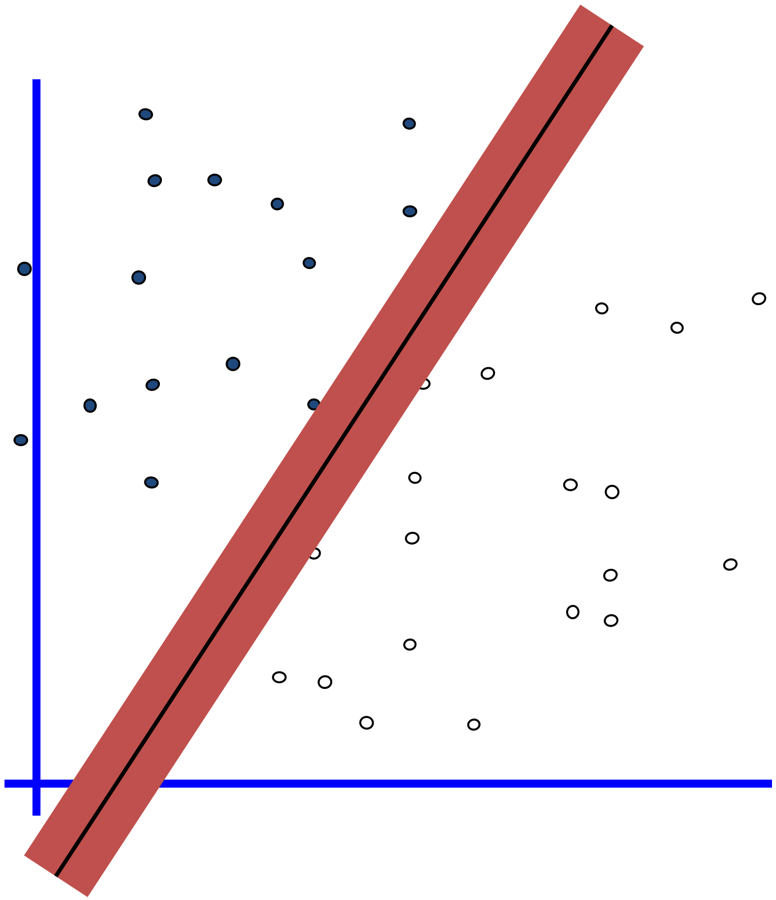
Which of these linear classifiers is the best?

Classifier Margin



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

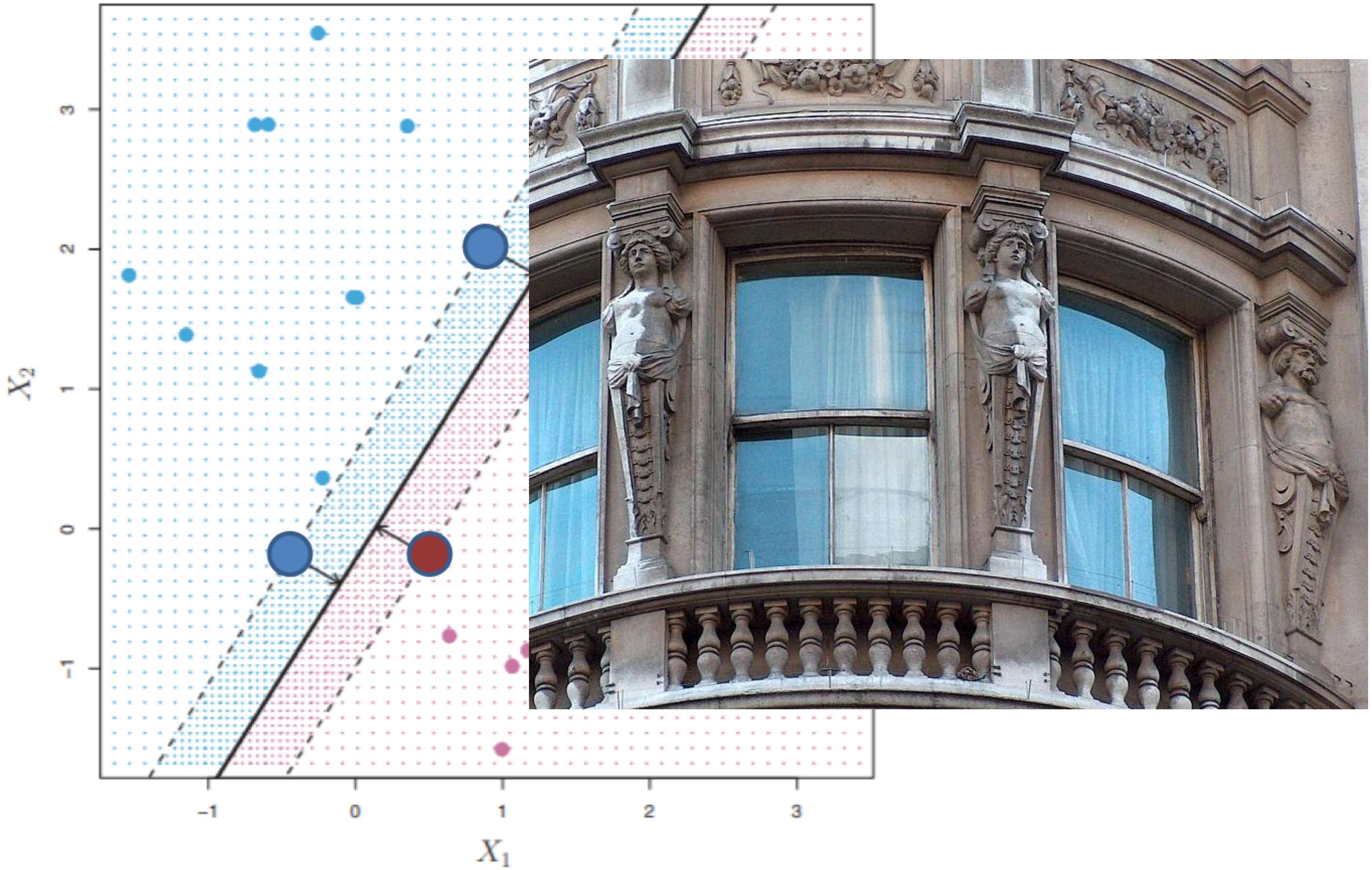
Maximum Margin



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

Choose the maximum margin linear classifier: the linear classifier with the maximum margin.

Support Vectors (informally)



- Support vectors = points “closest” to hyperplane
- If support vectors change, classifier changes
- If other points change, no effect on classifier

Finding the maximum margin classifier

- Training data $x_1, \dots, x_N$ with $x_i = \left(x_{i1}, \dots, x_{id}\right)^{\mathrm{T}}$
- Labels are from 2 classes: $y_i \in \{-1,1\}$

maximize M

$$y_i\left(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}\right) \geq M \ \forall i$$

$$\left|\left|\theta\right|\right|_2 = 1$$

Normalization constraint (to have unique solution)

Each point is on the right side of hyper-plane at distance $\geq M$

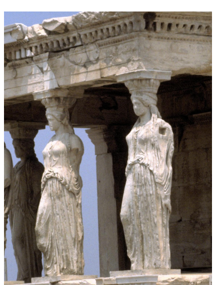
Equivalent formulation

- Min $||\theta||^2$
- $y_i(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}) \geq 1 \; \forall i$

- Maximum margin classifier – given by solution θ to this optimization problem
- Can be solved with quadratic optimization techniques. Easier to solve via its dual problem.

Properties of solution

- The solution to the (dual) optimization provides a convenient way to rewrite the decision function using new variables α_i
  - Originally: $f(z) = \text{sign}(\theta_0 + \theta_1 z_1 + \cdots \theta_d z_d) = \text{sign}(\theta^T \text{z})$
  - Equivalent to: $f(z) = \theta_0 + \sum_i \alpha_i < z, x_i >$
    - For test point *z*, the inner product $< z, x_i >$ = $z^T x_i$ with each training instance x_i in turn.
    - And $\alpha_i \neq 0$ only for support vectors! For all other training points $\alpha_i = 0$.

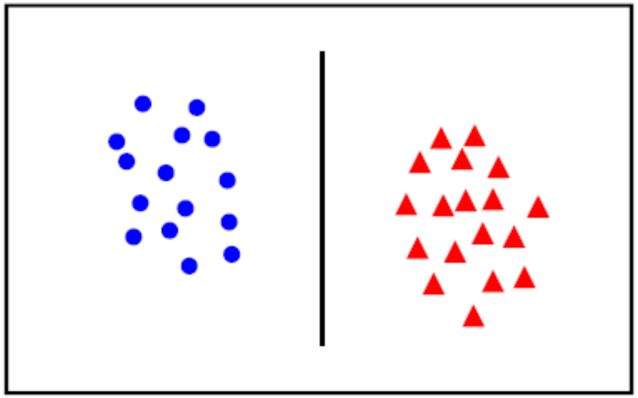


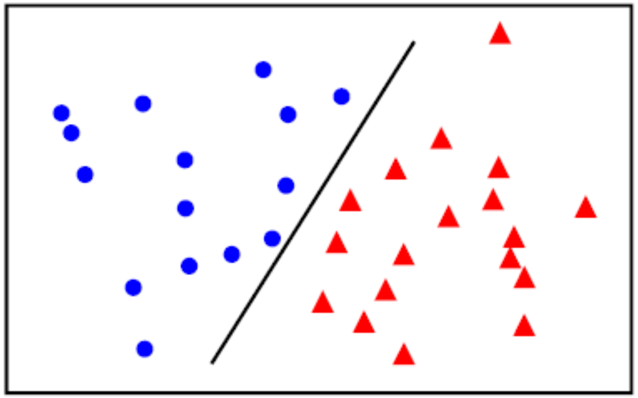
Outline

- Review of linear models
  - Separating hyperplanes
- Support Vector Machines
  - Linearly separable data
    - Maximum margin classifier
  - Non-separable data
    - Support vector classifier
  - Non-linear decision boundaries
    - Kernels and Radial SVM

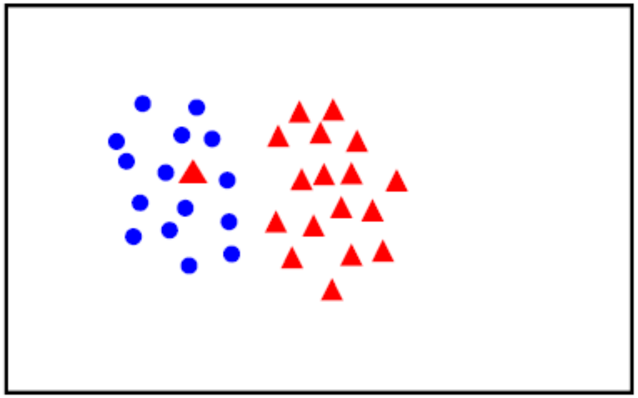
Linear separability

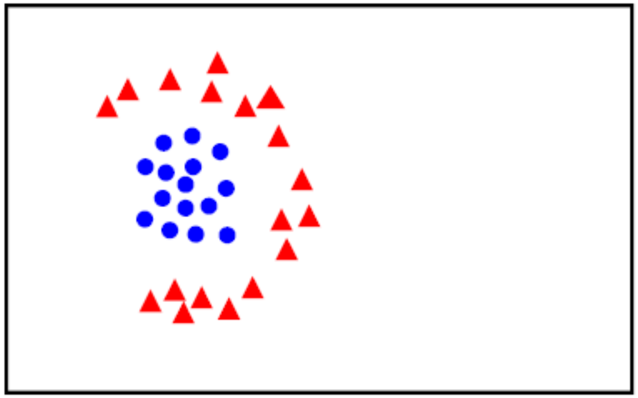
linearly
separable



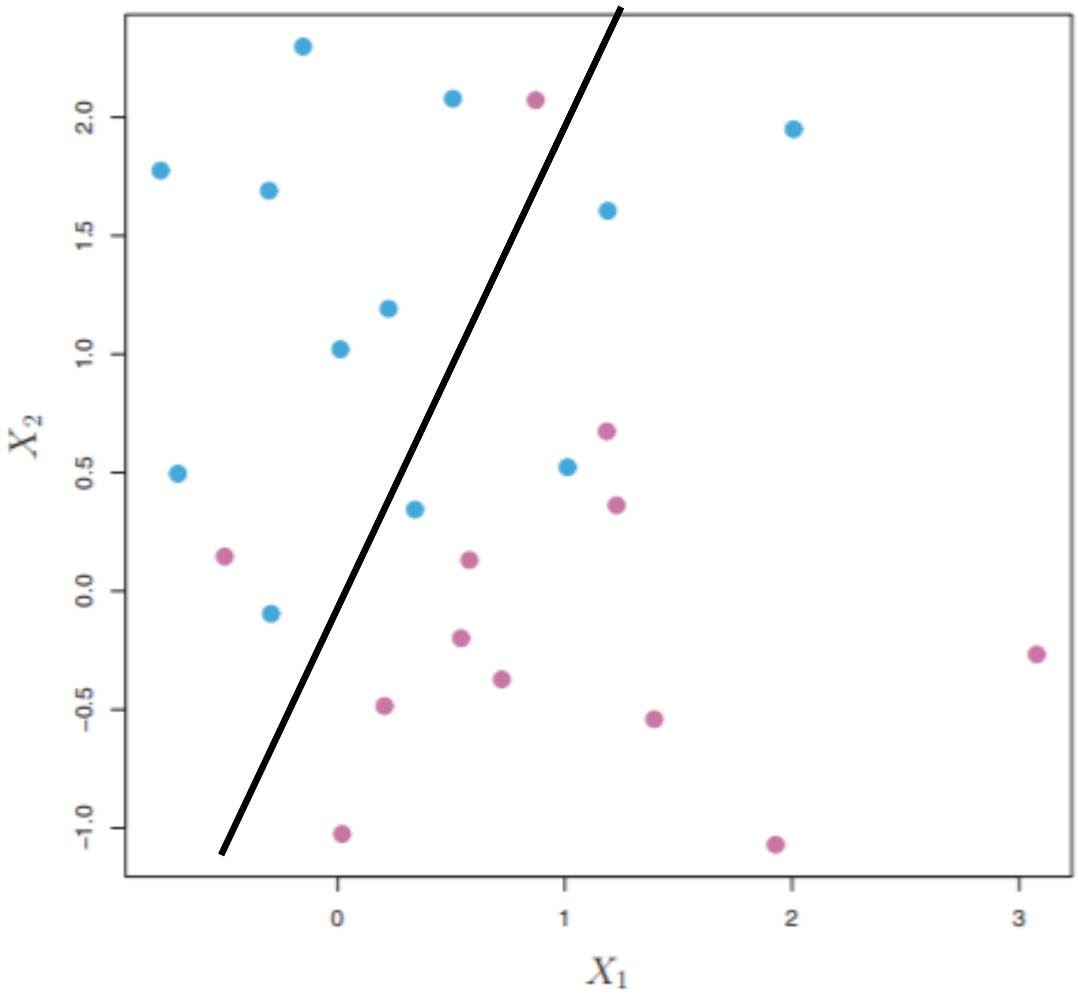


not
linearly
separable
(but almost)





Non-separable case



Optimization problem has no solution!

Support vector classifier

- Allow for small number of mistakes on training data
- Obtain a more robust model

max M

$$y_i(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}) \geq M(1 - \epsilon_i)\forall i$$

$$||\theta||_2 = 1$$

$$\epsilon_i \geq 0, \sum_i \epsilon_i = C$$

Slack

Error Budget (Hyper-parameter)

max M

$$y_i(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}) \geq M(1 - \epsilon_i\ )\ \forall i$$

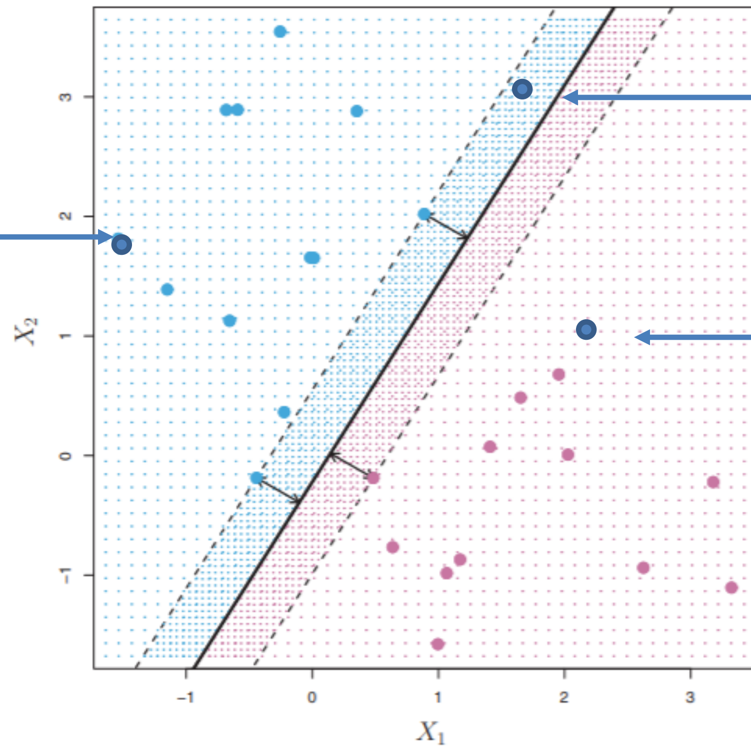
$$||\theta||_2 = 1$$

$$\epsilon_i \geq 0, \sum_i \epsilon_i = C$$

→ Error Budget

↓ Slack

$\epsilon_i = 0$
Correct side of margin



$0 < \ \epsilon_i < 1$
Violates margin
Correct label

$\epsilon_i > 1$
Incorrect label
At most C data points

Equivalent formulation

- Min $||\theta||^2 + C\sum_i \epsilon_i$
- $y_i(\theta_0 + \theta_1 x_{i1} + \cdots \theta_d x_{id}) \geq 1 - \epsilon_i \ \forall i$
- $\epsilon_i \geq 0$

- Just like in separable case, gives solution of the form:

$$f(z) = \theta_0 + \sum_i \alpha_i < z, x_i >$$

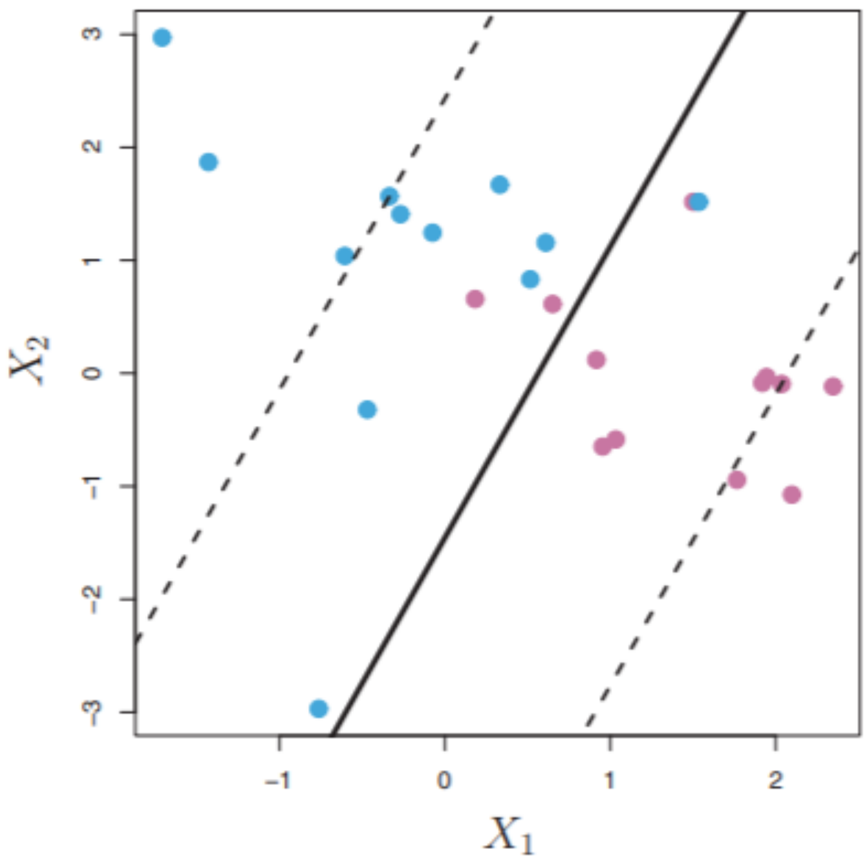
Where $\alpha_i \neq 0$ for support vectors (and $\alpha_i = 0$ for all other training points)

- This model is called Support Vector Classifier, also Linear SVM, also soft-margin classifier

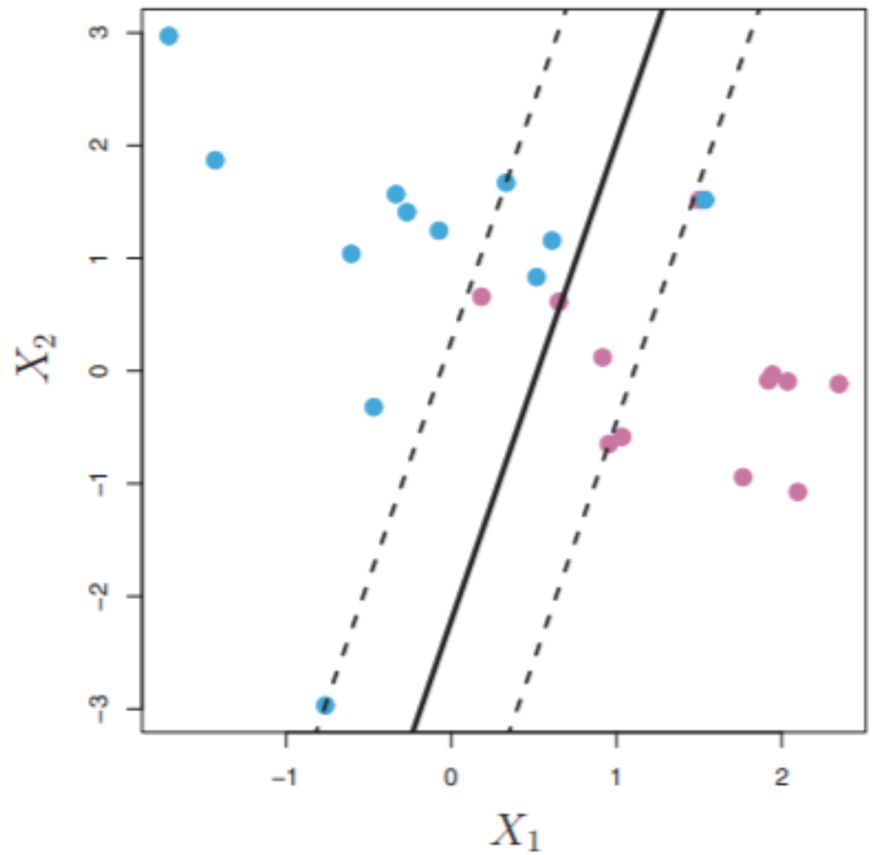
Properties

- Maximum margin classifier
  - Classifier of maximum margin
  - For linearly separable data
- Support vector classifier
  - Allows some slack and sets a total error budget (hyper-parameter)
- For both, final classifier on a point is a linear combination of inner product of point with support vectors
  - Efficient to evaluate

Error Budget and Margin



Larger C
Low variance



Smaller C
Over-fitting

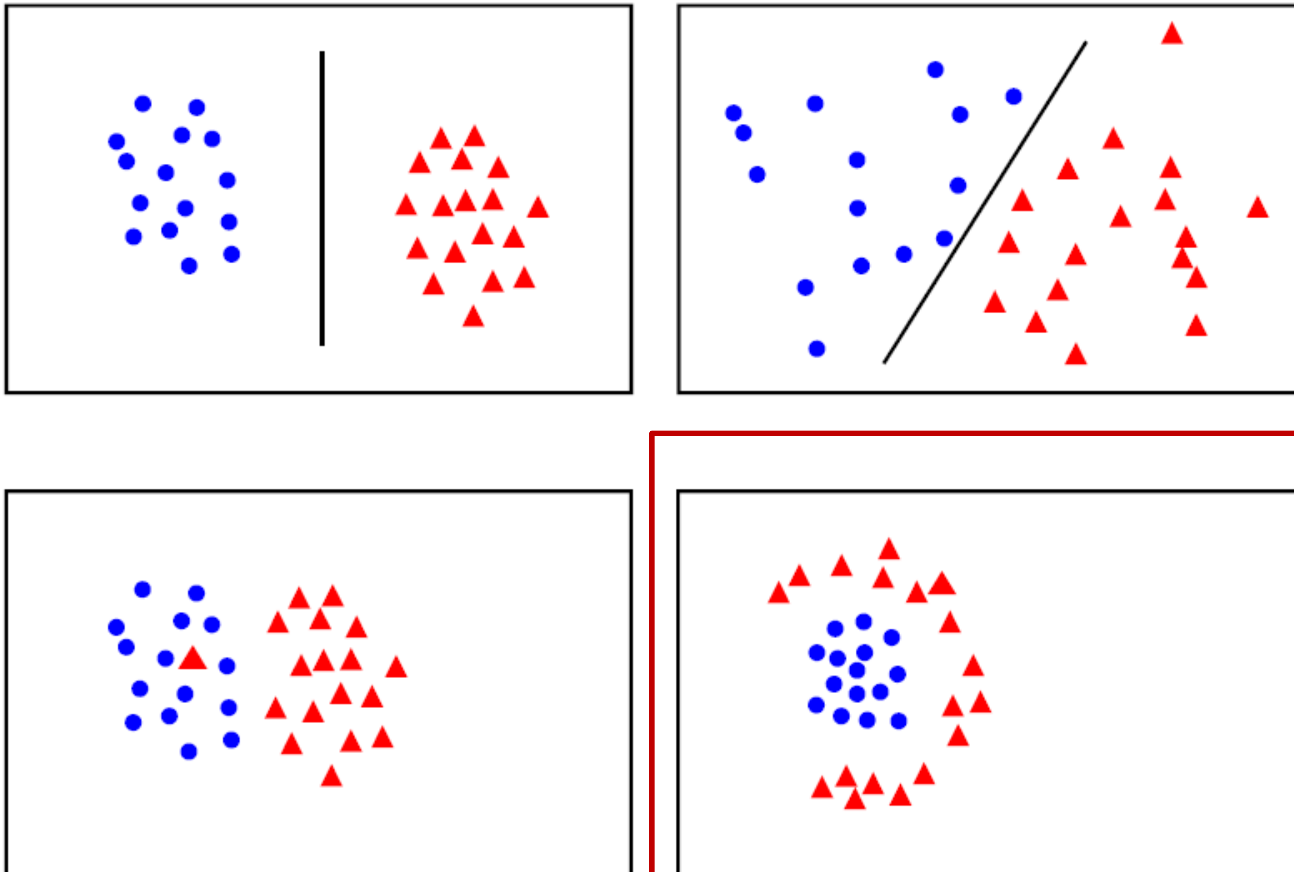
Find best hyper-parameter C by cross-validation

Outline

- Review of linear models
  - Separating hyperplanes
- Support Vector Machines
  - Linearly separable data
    - Maximum margin classifier
  - Non-separable data
    - Support vector classifier
  - Non-linear decision boundaries
    - Kernels and Radial SVM

Linear separability

linearly
separable



not
linearly
separable

(but almost)

(not even close!)

Non-linear decision

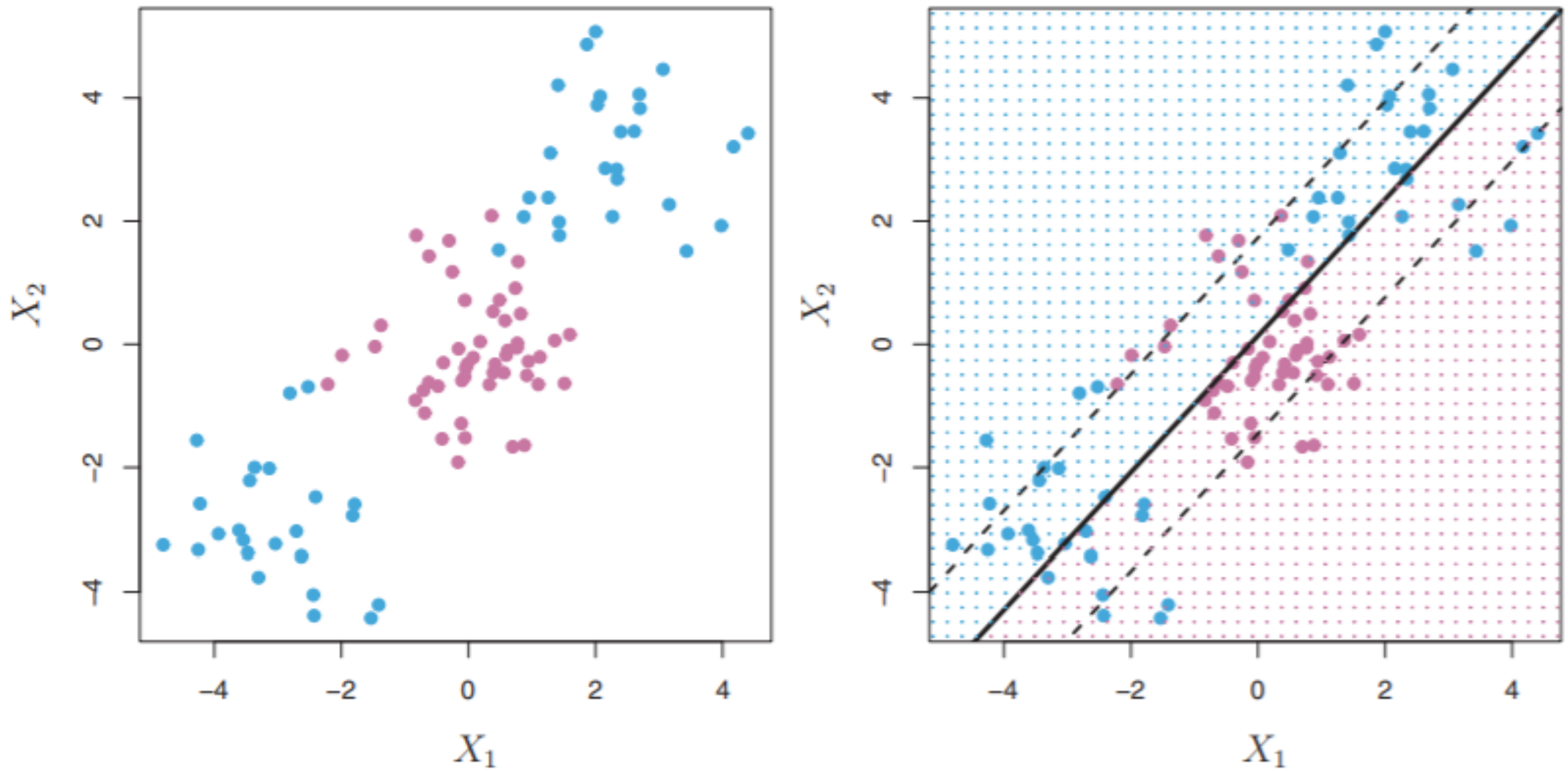
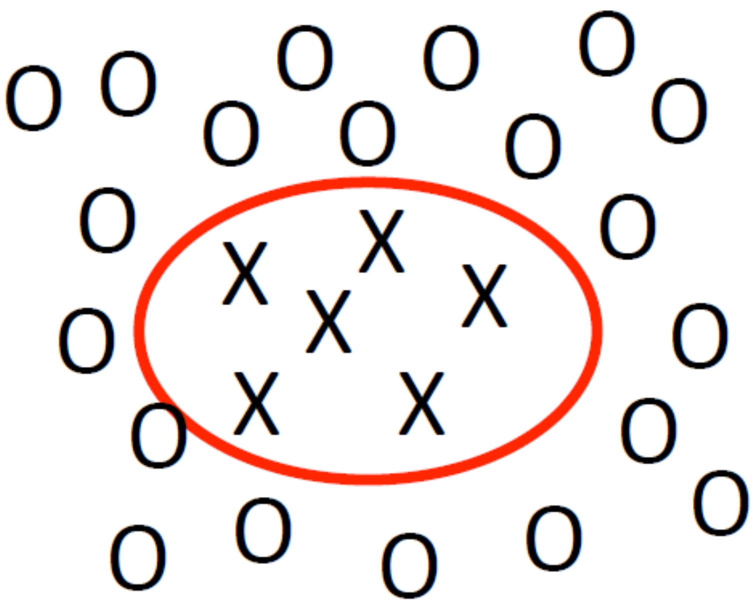


FIGURE 9.8. Left: *The observations fall into two classes, with a non-linear boundary between them.* Right: *The support vector classifier seeks a linear boundary, and consequently performs very poorly.*

More examples



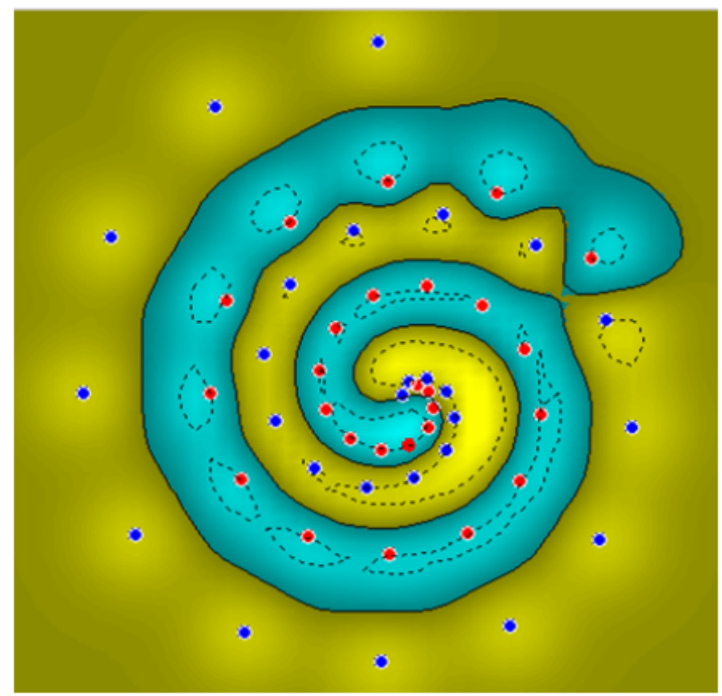


Image from http://www.atrandomresearch.com/iclass/

Kernels

- Support vector classifier
  - $h(z) = \theta_0 + \sum_{i \in S} \alpha_i < z, x_i >$
    $= \theta_0 + \sum_{i \in S} \alpha_i \sum_{j=1} z_j x_{ij}$
  - S is set of support vectors
  - Replace with $h(z) = \theta_0 + \sum_{i \in S} \alpha_i K(z, x_i)$

Any kernel function!

- What is a kernel?
  - Function that characterizes similarity between 2 observations
  - $K(a, b) = < a, b > = \sum_j a_j b_j$ linear kernel!
  - The closer the points, the larger the kernel
- Intuition
  - The closest support vectors to the point play larger role in classification

The Kernel Trick

"Given an algorithm which is formulated in terms of a positive definite kernel K_1, one can construct an alternative algorithm by replacing K_1 with another positive definite kernel K_2"

- SVMs can use the kernel trick

- Enlarge feature space
- Shape of the kernel changes the decision boundary

Kernels

- Linear kernels
  - $K(a, b) =< a, b > = \sum_i a_i b_i$
- Polynomial kernel of degree *m*
  - $K(a, b) = \left(1 + \sum_{i=0}^{d} a_i b_i\right)^m$
- Radial Basis Function (RBF) kernel (or Gaussian)
  - $K(a, b) = \exp\left(-\gamma \sum_{i=0}^{d} (a_i - b_i)^2\right)$
- Support vector machine classifier
  - $\mathrm{h}(z) = \theta_0 + \sum_{i \in S} \alpha_i K(z, x_i)$

General SVM classifier

- S = set of support vectors
- SVM with polynomial kernel
  - $h(z) = \theta_0 + \sum_{i \in S} \alpha_i \left(1 + \sum_{j=0}^{d} z_j x_{ij}\right)^m$
  - Hyper-parameter m (degree of polynomial)
- SVM with radial kernel
  - $h(z) = \theta_0 + \sum_{i \in S} \alpha_i \exp\left(-\gamma \sum_{j=0}^{d} (z_j - x_{ij})^2\right)$
  - Hyper-parameter γ (increase for non-linear data)
  - As testing point z is closer to support vector, kernel is close to 1
  - Local behavior: points far away have negligible impact on prediction

Kernel Example

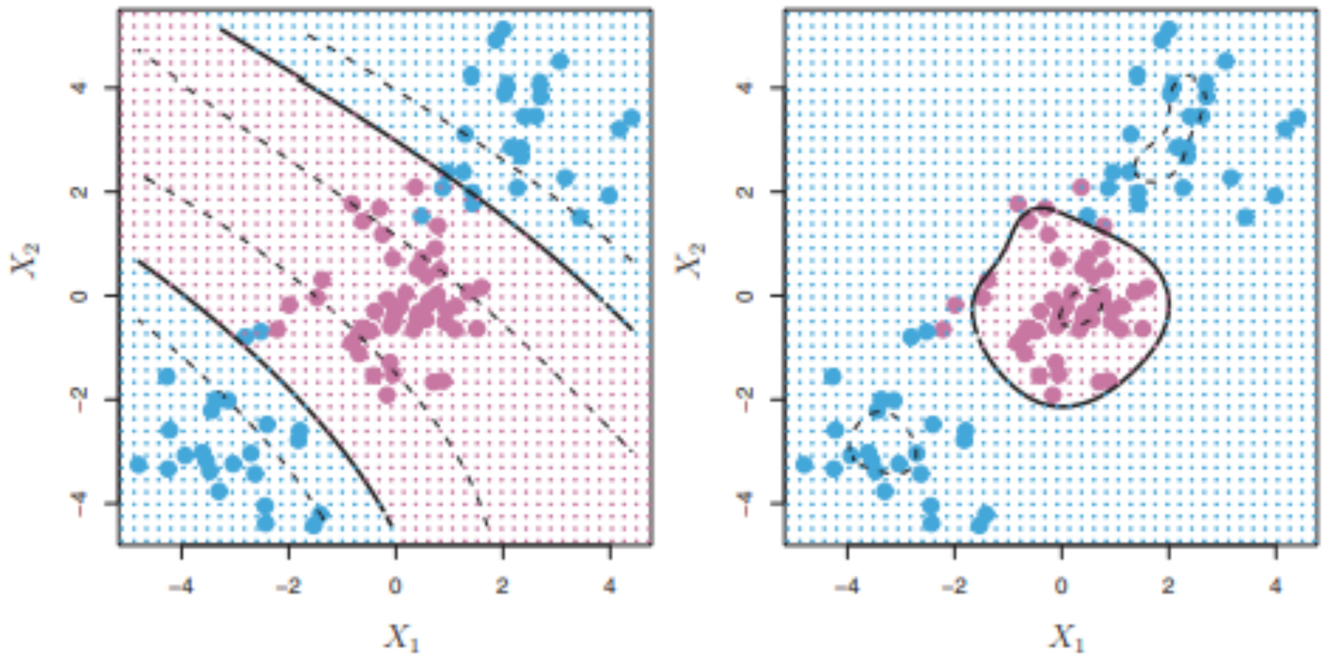


FIGURE 9.9. Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

Advantages of Kernels

- Generate non-linear features
- More flexibility in decision boundary
- Generate a family of SVM classifiers
- Testing is computationally efficient
  - Cost depends only on support vectors and kernel operation

- Disadvantages
  - Kernels need to be tuned (additional hyper-parameters)

When to use different kernels?

- If data is (close to) linearly separable, use linear SVM
- Radial or polynomial kernels preferred for non-linear data
- Training radial or polynomial kernels takes longer than linear SVM
- Other kernels
  - Sigmoid
  - Hyperbolic Tangent

Review SVM

- SVMs find optimal linear separator
- The kernel trick makes SVMs learn non-linear decision surfaces

- Strength of SVMs:
  - Good theoretical and empirical performance
  - Supports many types of kernels

- Disadvantages of SVMs:
  - "Slow" to train/predict for huge data sets (but relatively fast!)
  - Need to choose the kernel (and tune its parameters)

Comparing SVM with other classifiers

- SVM is resilient to outliers
  - Similar to Logistic Regression
  - LDA or kNN are not
- SVM can be trained with Gradient Descent
  - Hinge loss cost function
- Supports regularization
  - Can add penalty term (ridge or Lasso) to cost function
- Linear SVM is most similar to Logistic Regression

Acknowledgements

- Slides made using resources from:
  - Andrew Ng
  - Eric Eaton
  - David Sontag
  - Andrew Moore
- Thanks!