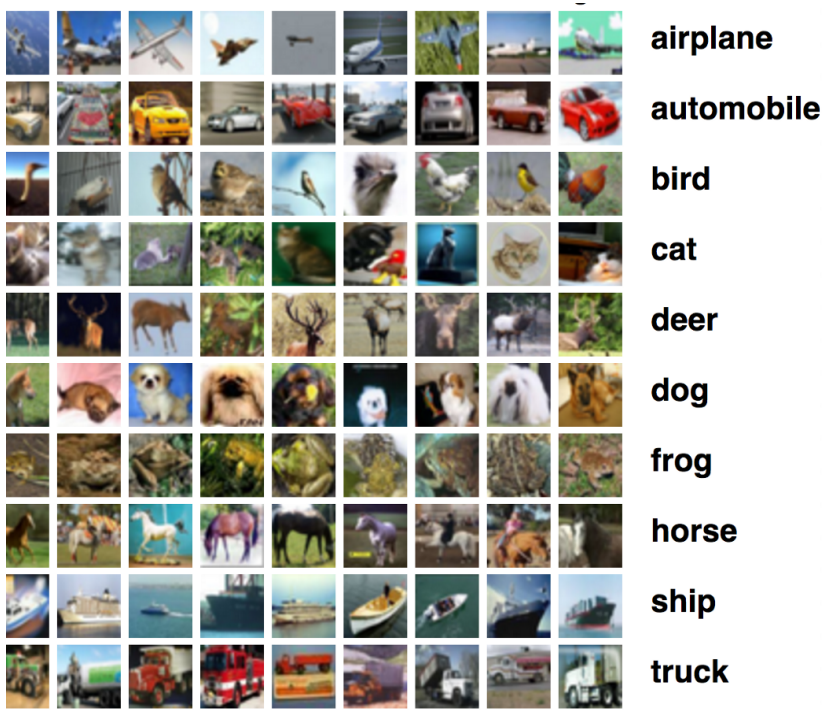
ECE 4803/8803: Fundamentals of Machine Learning (FunML) Spring 2024

Lecture 4: Classifiers



OLIVES @GeorgiaTech

Georgia Tech

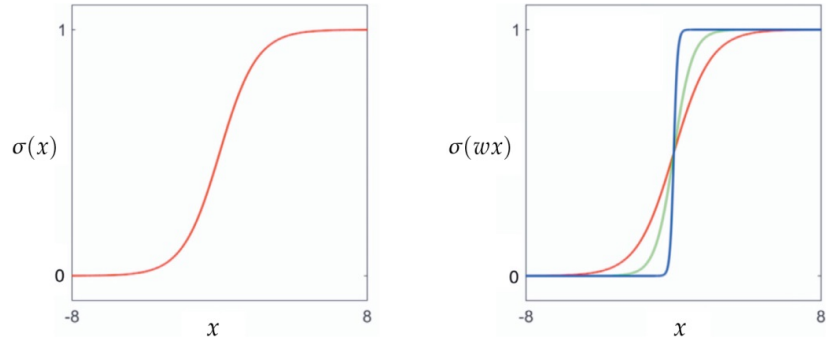
Logistic Regression

Summary

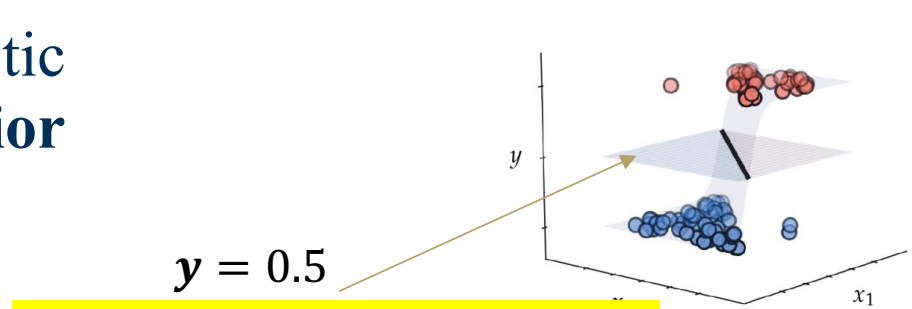
- The modeling process we just went through is called logistic regression
- Logistic Regression is a **parametric model**
- Given feature vector $\boldsymbol{x} = [x_1, \dots, x_p]^T$, logistic regression classifier estimates the **posterior** $P(y|\boldsymbol{x})$ with *sigmoid function* $\sigma(\boldsymbol{x})$:

$$P(y = 1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{x} + b)$$
$$P(y = 0|\boldsymbol{x}) = 1 - \sigma(\boldsymbol{w}^T\boldsymbol{x} + b)$$

- Where $\boldsymbol{w} = [w_1, \dots, w_p]^T$ is **weight**, vector b is **bias**
- Classification rule:
  - $y = \begin{cases} 1 & P(y = 1|\boldsymbol{x}) > 0.5 \\ 0 & otherwise \end{cases}$

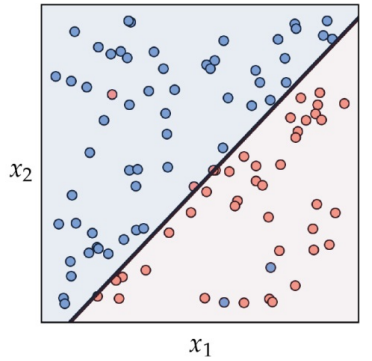


Sigmoid function $\sigma(\boldsymbol{x}) = \dfrac{1}{1 + e^{-\boldsymbol{x}}}$



$\boldsymbol{y} = 0.5$

Linear separation hyperplane



$\boldsymbol{\theta} = \{\boldsymbol{w}, b\}$

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) - \alpha \frac{\partial LL(\boldsymbol{x}, y, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$

The overall negative log-likelihood cost function:

$$LL(\boldsymbol{x}, y) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\big(\sigma(\boldsymbol{x}_i)\big) + (1 - y_i)\log\big(1 - \sigma(\boldsymbol{x}_i)\big)$$

N: number of samples, y_i: desired output ,z_i: predicted output

$LL(\boldsymbol{x}, y)$is referred to as the **Cross Entropy Cost** for logistic regression. In binary classification, this is also referred to as the **Binary Cross Entropy Cost**

Logistic Regression

Example: Sentiment Classification

Let's assume we want to predict the binary sentiment for the following movie review:

Input: "It's hokey. There are virtually no surprises, and the writing is second-rate. So why was it so enjoyable? For one thing, the cast is great. Another nice touch is the music. I was overcome with the urge to get off the couch and start dancing. It sucked me in, and it'll do the same to you."

Output: positive (1) or negative (0)

Logistic Regression

Example: Sentiment Classification

Input: "It's hokey . There are virtually no surprises , and the writing is second-rate. So why was it so enjoyable? For one thing, the cast is great. Another nice touch is the music. I was overcome with the urge to get off the couch and start dancing. It sucked me in, and it'll do the same to you."

Output: positive (1) or negative (0)

Let's assume the sentiment is represented by 6 features

$$\boldsymbol{x} = [x_1, \dots x_6]$$

Thus, we want to estimate

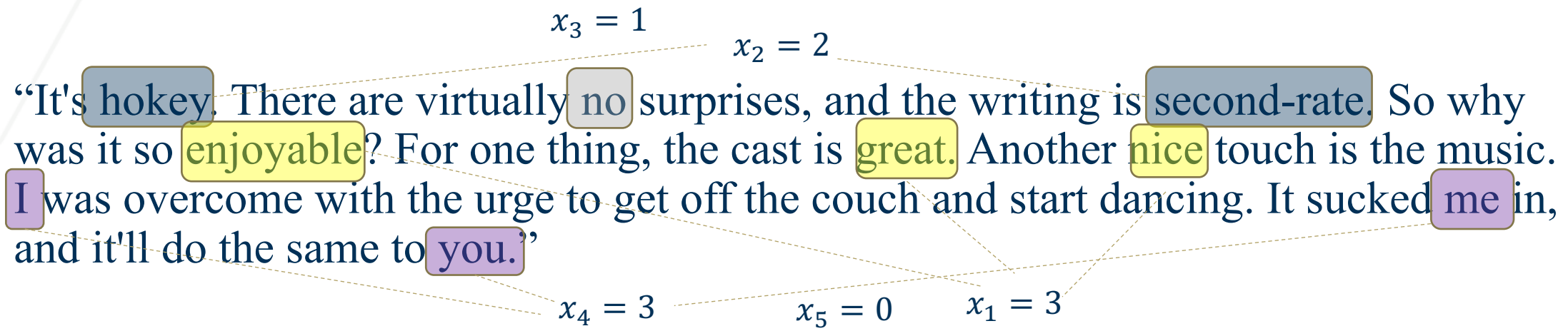
$$P(semtiment = 1|\boldsymbol{x})$$

$$P(sentiment = 0|\boldsymbol{x})$$

feature	description	value
x_1	Count positive words	
x_2	Count negative words	
x_3	$\begin{cases} 1 & if\ "no"\ in\ text \\ 0 & otherwise \end{cases}$	
x_4	Count 1st and 2nd pronouns	
x_5	$\begin{cases} 1 & if\ "!"\ in\ text \\ 0 & otherwise \end{cases}$	
x_6	$\log(word\ count)$	

Logistic Regression

Example: Sentiment Classification



feature	description	value
x_1	Count positive words	3
x_2	Count negative words	2
x_3	$\begin{cases} 1 & if\ "no"\ in\ text \\ 0 & otherwise \end{cases}$	1
x_4	Count 1^{st} and 2^{nd} pronouns	3
x_5	$\begin{cases} 1 & if\ "!"\ in\ text \\ 0 & otherwise \end{cases}$	0
x_6	$\ln(word\ count)$	ln(56)=4.025

Logistic Regression

Example: Sentiment Classification

Input: "It's hokey . There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ? For one thing , the cast is great . Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you."

Output: positive (1) or negative (0)

Let's assume the weights and bias are:

$$\boldsymbol{w} = [w_1, \dots w_6]^T = [2.5, -5.0, -1.2, 0.5, 2.0, 0.7]^T$$
$$b = 0.1$$

feature	description	value
x_1	Count positive words	3
x_2	Count negative words	2
x_3	$\begin{cases} 1 & if\ "no"\ in\ text \\ 0 & otherwise \end{cases}$	1
x_4	Count 1st and 2nd pronouns	3
x_5	$\begin{cases} 1 & if\ "!"\ in\ text \\ 0 & otherwise \end{cases}$	0
x_6	$\ln(word\ count)$	4.025

Thus

$$P(semtiment = 1|\boldsymbol{x}) = \sigma(\boldsymbol{w^T x} + b)$$
$$= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7][3, 2, 1, 3, 0, 4.025]^T + 0.1)$$
$$= 0.67$$
$$P(sentiment = 0|\boldsymbol{x}) = \boldsymbol{1 - \sigma(wx} + b) = 0.33$$

Classifier Comparison

Methods	Assumptions on Feat. Dist.	Feat. Normalization	Cost Function	Regularization	Linear Classifier	Prob. View of Prediction	Generative/Discriminative	Parametric/ Non-parametric	Overfitting
Logistic Regression	No	Required	BCE (convex)	Additional term	Linear	Yes	Discriminative	Parametric	Not often
K Nearest Neighbors	No	Required	N/A	N/A	Non-linear	N/A	Discriminative	Non-parametric	when k is too small
Decision Trees	No	Not Required	N/A	N/A	Non-linear	N/A	Discriminative	Non-parametric	with large depth
Support Vector Machines	No	Required	Hinge (convex)	C (control robustness)	Linear/ Non-linear(kernel)	N/A	Discriminative	Parametric	Not often
Naïve Bayes	Conditional independent	Not Required	N/A	N/A	Non-linear /Linear (Gaussian)	Yes	Generative	Parametric	Not often
Artificial Neural Networks	No	Required	Non-convex	Additional term	Non-linear	Yes	Discriminative/ Generative	Parametric	with many layers

Overview

In this Course..

Nearest Neighbor

Naïve Bayes

Logistic Regression

Decision Trees (know it exists but will not study it)

- Overview
- ID3 Algorithm
- Entropy & Information Gain
- Example

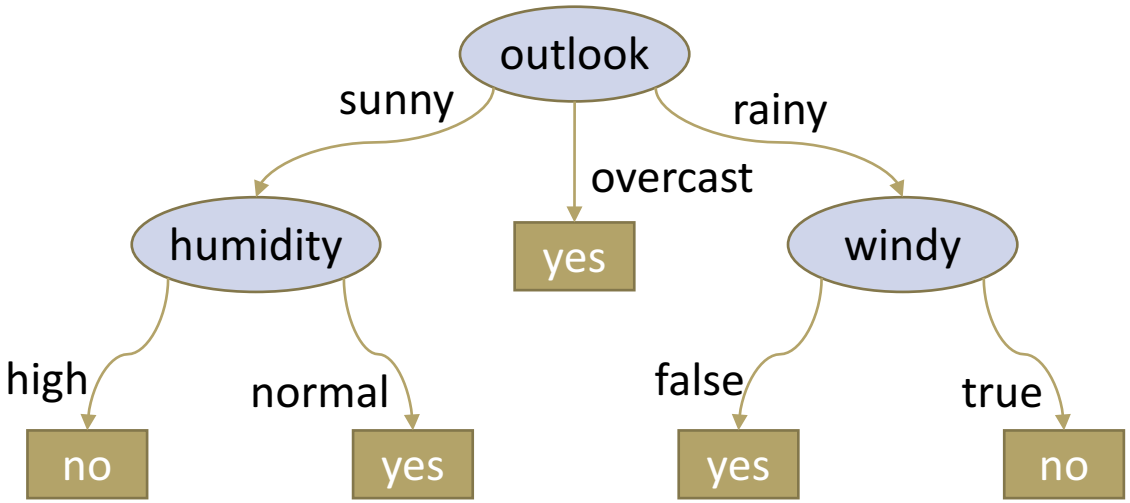
Support Vector Machines

Artificial Neural Networks

Decision Trees

Overview

- Can be used for both Classification and Regression
- The learning process builds classification model in the form of a tree structure that mimics rule-based human reasoning



Final decision tree computed based on data in table

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
Overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Overview

In this Course..

- Nearest Neighbor
- Naïve Bayes
- Logistic Regression
- Decision Trees
- Support Vector Machines
  - Linear Separators
  - Maximum Margin Classification
  - Soft Margin Classification
  - Multi-class SVM
  - Non-linear SVM and kernel tricks
- Artificial Neural Networks

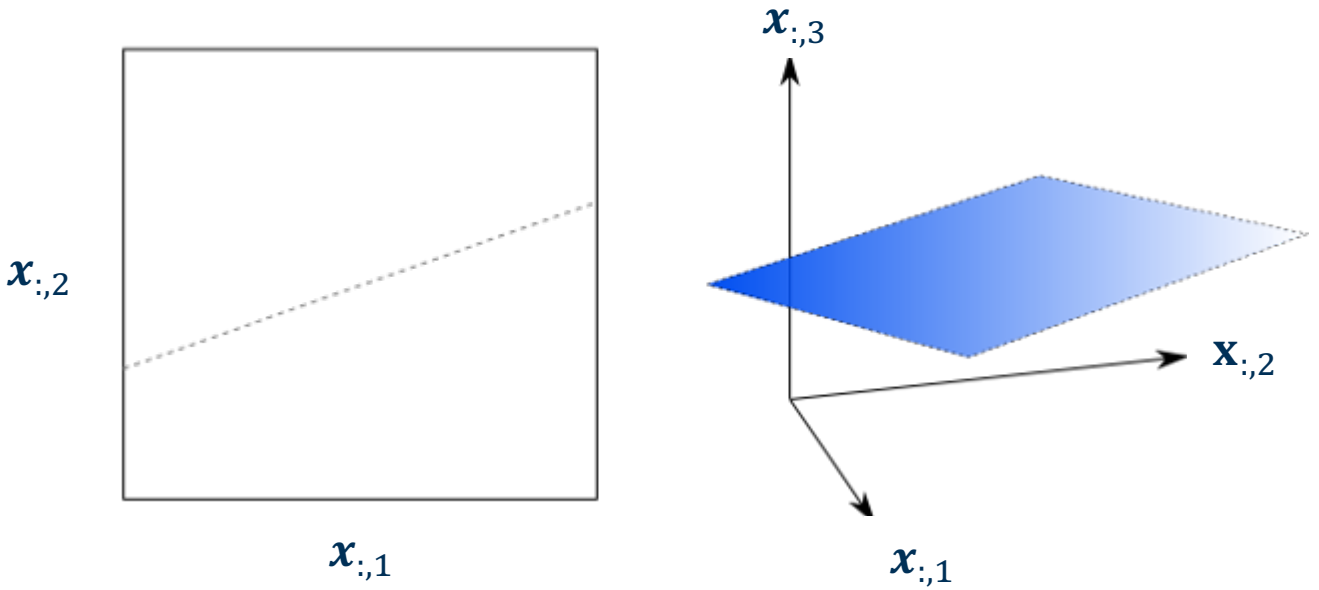
Support Vector Machines

Overview

- Linear Separating Hyperplanes
- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
- Advantages and Disadvantages

Support Vector Machines

Linear Hyperplanes



One-dimensional hyperplane Two-dimensional hyperplane

Support Vector Machines

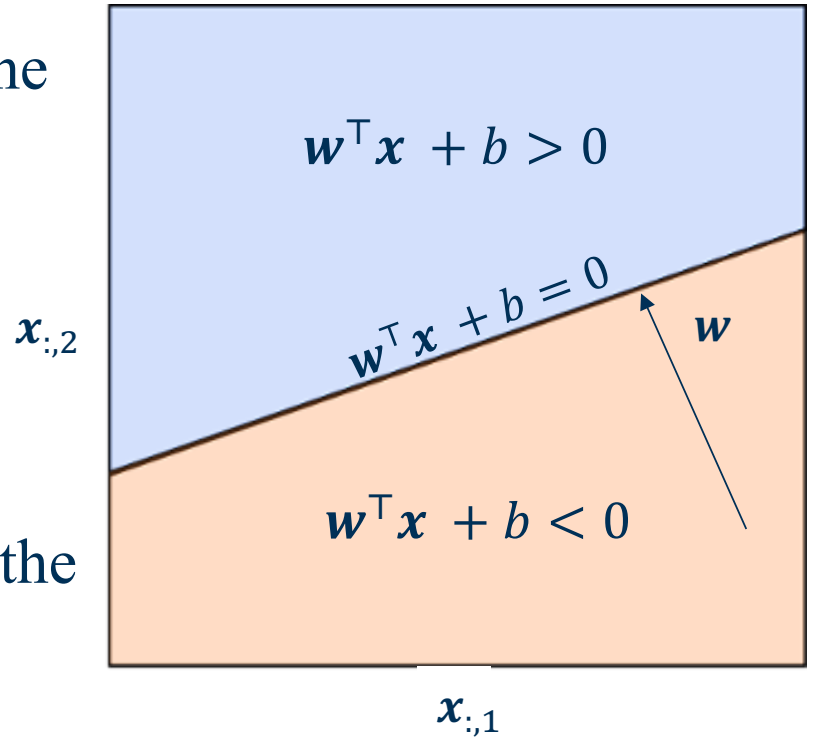
Linear Hyperplanes

- Separating P-dimensional feature space by a hyperplane
- Given a single sample $\boldsymbol{x} = [x_1,\ x_2, \ldots, x_P] \in \mathbb{R}^P$, we define a hyperplane by the following equation:

$$b + w_1 x_1 + \cdots + w_P\, x_P = 0$$

$$\text{or } \boldsymbol{w}^\top \boldsymbol{x} + b = 0$$

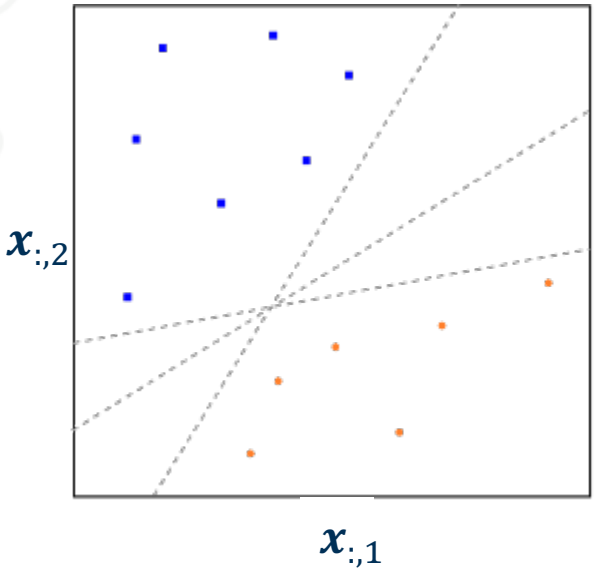
- $b \neq 0$ gives us an affine plane (*i.e.* it does not pass throug the origin)
- Note: the vector $\boldsymbol{w}$ is ANY vector that is perpendicular to the hyperplane. However, for simplicity and without loss of generality, we will assume $\boldsymbol{w}$ is a unit norm vector.



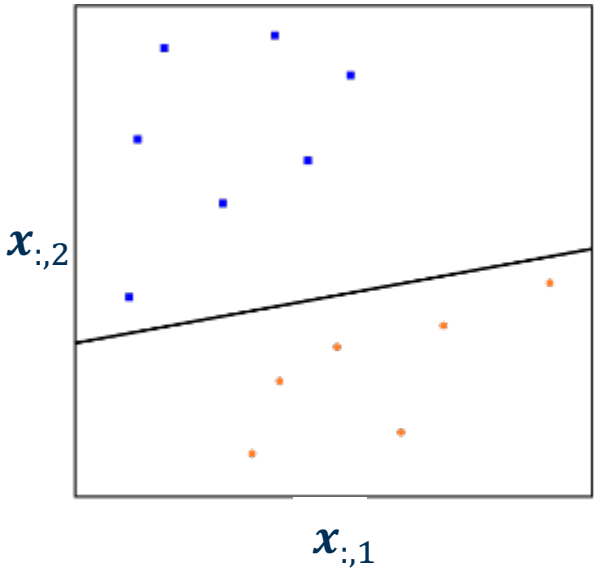
Separation of two-dimensional featu space by a hyperplane

Support Vector Machines

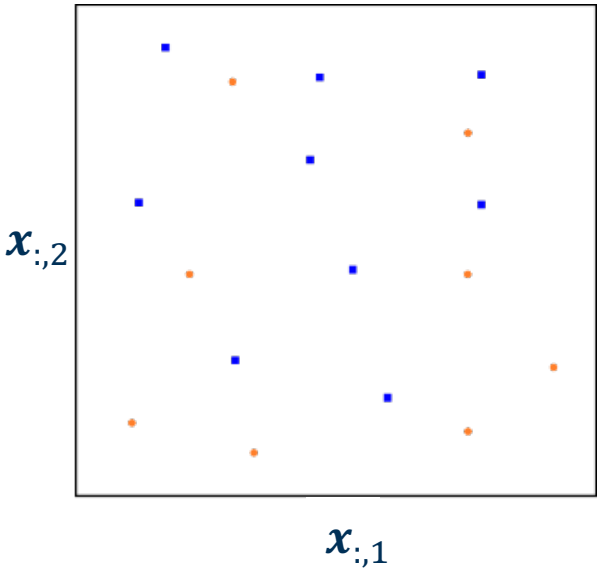
Linear Hyperplanes



Multiple separating hyperplanes



Perfect separation of class data



No possibility of a true separating hyperplane

Support Vector Machines

Maximal Margin Classifiers

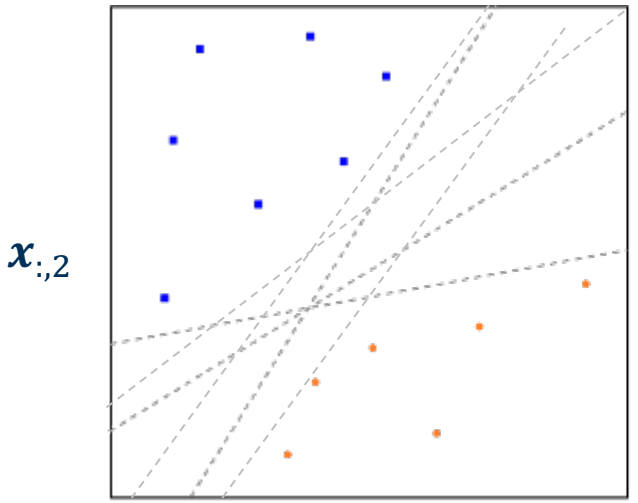
- Our goal is to develop a classifier based on training data that will correctly classify test data using only their feature values.
- We will initially suppose that it is possible, via a means yet to be determined, to construct a hyperplane that separates training data *perfectly* according to their class labels.
- The following diagram is only showing $P = 2$. However, for real examples (like spam email filtering) $P > 10^6$.

$$\boldsymbol{w}^\top \boldsymbol{x} + b > 0, \text{ if } y = 1 \text{ (class 1)}$$

and

$$\boldsymbol{w}^\top \boldsymbol{x} + b < 0, \text{ if } y = -1 \text{ (class 2)}$$

- However, based on the training data, it's possible to find many hyperplanes that satisfy the above. How do we choose the "**best**" ?



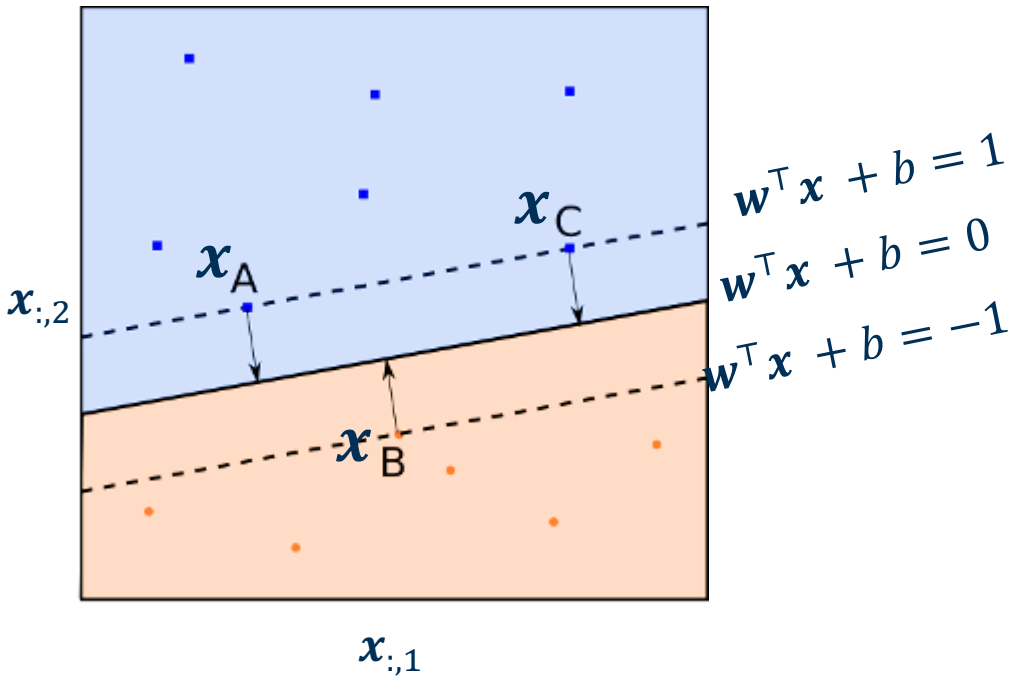
Multiple separating hyperplanes

Support Vector Machines

Maximal Margin Classifiers

- The **maximal margin hyperplane** (MMH) is the separating hyperplane that is farthest from any training data.
- The location of the MMH only depends on the training observations that lie directly on the margin boundary ($\boldsymbol{x}_A$, $\boldsymbol{x}_B$ and $\boldsymbol{x}_C$)
- ($\boldsymbol{x}_A$, $\boldsymbol{x}_B$ and $\boldsymbol{x}_C$) are known as the **support vectors**.
- Thus, the goal is:

finding an algorithm that can produce the best MMH.



Best = Highest width and highest accuracy

Support Vector Machines

Width of Maximal Margin Classifiers

- Let's try to write the width of the margin as a function of something we know.
- Let's change the equation as follows, to ensure no samples are within the margin M
  - $\boldsymbol{w}^\top \boldsymbol{x}^+ + b \geq 1$
  - $\boldsymbol{w}^\top \boldsymbol{x}^- + b \leq -1$

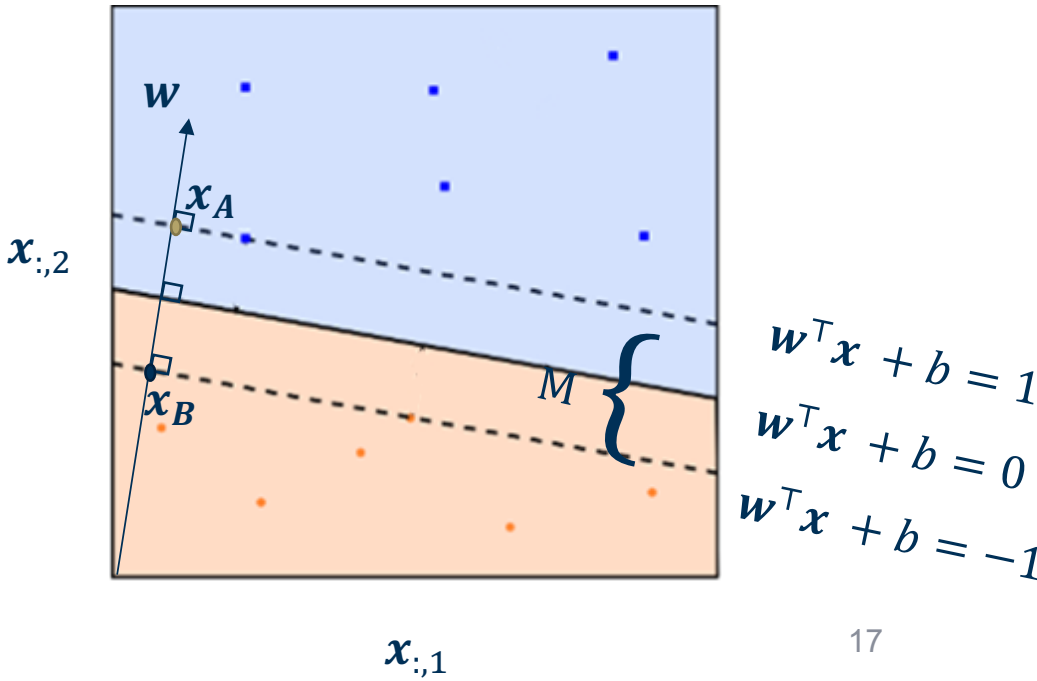
$\mathbf{x}^+$ is a class 1 example
$\mathbf{x}^-$ is a class 2 example

- Note the following from the figure:

$$(\boldsymbol{x}_A - \boldsymbol{x}_B)^\top \boldsymbol{w} = \mathbf{2}$$

But the left-hand side is the inner product and can Be written as: $\left|\left|\boldsymbol{x}_A - \boldsymbol{x}_B\right|\right| \left|\left|\boldsymbol{w}\right|\right| \boldsymbol{cos}(\mathbf{0}) = \mathbf{2}$

Thus, $M = \frac{\mathbf{2}}{\|\boldsymbol{w}\|} = \left|\left|\boldsymbol{x}_A - \boldsymbol{x}_B\right|\right|$



17

Support Vector Machines

Width of Maximal Margin Classifiers

- Let's try to write the width of the margin as a function of something we know.
- Let's change the equation as follows, to ensure no samples are within the margin M
  - $\boldsymbol{w}^\top \boldsymbol{x}^+ + b \geq 1$
  - $\boldsymbol{w}^\top \boldsymbol{x}^- + b \leq -1$

$\mathbf{x}^+$ is a class 1 example
$\mathbf{x}^-$ is a class 2 example

- Note the following from the figure:

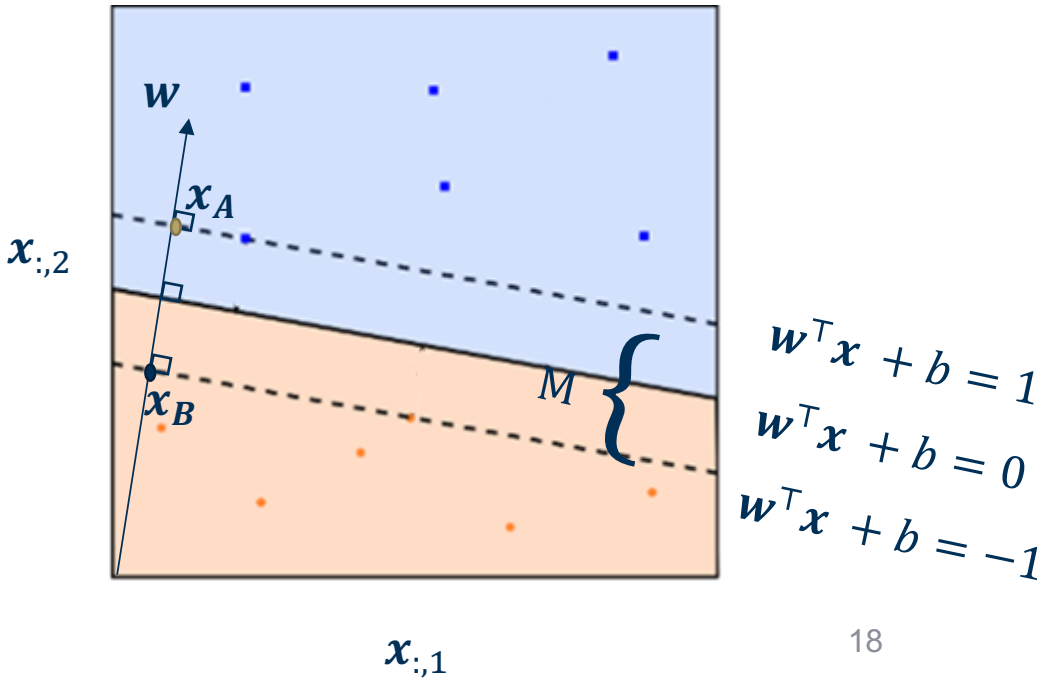
$$(\boldsymbol{x}_A - \boldsymbol{x}_B)^\top \boldsymbol{w} = \mathbf{2}$$

But the left-hand side is the inner product and can Be written as: $\boldsymbol{||x_A - x_B||\,||w||cos(0) = 2}$

Thus, $M = \frac{\mathbf{2}}{\|\boldsymbol{w}\|} = \boldsymbol{||x_A - x_B||}$

Therefore, we want to:

- $\mathbf{max}\frac{\mathbf{1}}{\|\boldsymbol{w}\|}$ **or min** $\|\boldsymbol{w}\|$ **or min** $\frac{\mathbf{1}}{\mathbf{2}}\|\boldsymbol{w}\|^{\mathbf{2}}$



Support Vector Machines

Accuracy of Maximal Margin Classifiers

- Perceptron with a linear boundary $\boldsymbol{w^T x + b}$=0:

$$\begin{cases} \boldsymbol{w^T x + b} > 0, & if\ y = +1 \\ \boldsymbol{w^T x + b} < 0, & if\ y = -1 \end{cases}$$

- Suppose that all samples could be separated further by $\boldsymbol{w^T x + b} = \pm\mathbf{1}$, the above could be written as following:

$$\begin{cases} \boldsymbol{w^T x + b} \geq 1, & if\ y = +1 \\ \boldsymbol{w^T x + b} \leq -1, & if\ y = -1 \end{cases}$$

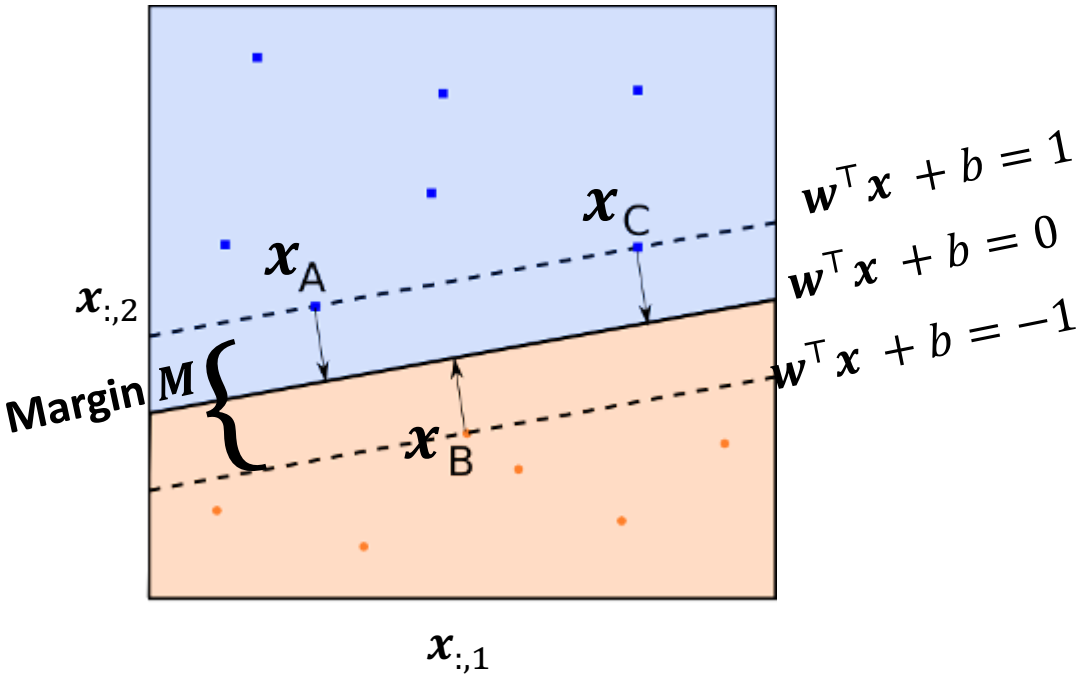
- The above formula could be written as:

$$y(\boldsymbol{w}^\top \boldsymbol{x}\ + b) - 1\ \geq 0$$

- Or equivalently as **Margin-Perceptron cost**,

$$L(\boldsymbol{x_i}, y_i, \boldsymbol{\theta}) = \max\left(0, 1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i}\ + b)\right) = 0$$

This additional 1 prevents the trivial zero solution with the original perceptron



Support Vector Machines

Accuracy of Maximal Margin Classifiers

- We want our classifier to be correct. Hence, we need to satisfy:
  - $\boldsymbol{w}^\top \boldsymbol{x}^+ + b \geq 1$
  - $\boldsymbol{w}^\top \boldsymbol{x}^- + b \leq -1$
- Let's multiply both inequalities by y_i.
- Since $y_i = 1$ for $\boldsymbol{x}_i^+$ and $y_i = -1$ for $\boldsymbol{x}_i^-$, we get
  - $y_i(\boldsymbol{w}^\top \boldsymbol{x}_i^+ + b) \geq y_i \rightarrow y_i(\boldsymbol{w}^\top \boldsymbol{x}_i^+ + b) \geq 1$
  - $y_i\,(\boldsymbol{w}^\top \boldsymbol{x}_i^- + b) \geq -y_i \rightarrow y_i(\boldsymbol{w}^\top \boldsymbol{x}_i^- + b) \geq 1$
- So, the two inequalities become identical, and our optimization objectives become:

$$minimize\ \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$Subject\ to\ y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) - 1 \geq 0$$

Support Vector Machines

Summary

- $y(\boldsymbol{w}^\top \boldsymbol{x} + b) - 1 \geq 0$
- The optimization objectives become:

$$minimize \frac{1}{2} \|\boldsymbol{w}\|^2$$

$$Subject\ to\ y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) - 1 \geq 0$$

Margin-Perceptron cost

- Or equivalently,

$$minimize \frac{1}{2} \|\boldsymbol{w}\|^2$$

$$\text{Subject to } \max\left(0, 1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b)\right) = 0$$

Support Vector Machines

Optimization

$$minimize \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$Subject\ to\ y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) - 1 \geq 0$$

- Since we have a constrained optimization objective, we use Lagrange multiplier to find our cost function.

$$L = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) - 1]$$

- We can solve this by taking the derivative and equating it to 0

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i\, y_i \boldsymbol{x}_i = 0 \rightarrow \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i\, y_i \boldsymbol{x}_i$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N} \alpha_i\, y_i = 0 \rightarrow \sum_{i=1}^{N} \alpha_i\, y_i = \boldsymbol{0}$$

- Next, let's plug those values back in the cost function

Support Vector Machines

Optimization

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i \, y_i \boldsymbol{x}_i$$

$$\sum_{i=1}^{N} \alpha_i \, y_i = \mathbf{0}$$

$$L = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) - 1]$$

$$= \frac{1}{2}\left(\sum_{i=1}^{N} \alpha_i \, y_i \boldsymbol{x}_i\right)^\top \left(\sum_{j=1}^{N} \alpha_j \, y_j \boldsymbol{x}_j\right) - \sum_{i=1}^{N} \alpha_i \left[y_i \left(\left(\sum_{j=1}^{N} \alpha_j \, y_j \boldsymbol{x}_j\right)^\top \boldsymbol{x}_i + b \right) - 1 \right]$$

$$= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x_j} - \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x_j} - b\cancel{\sum_{i=1}^{N} \alpha_i \, y_i} + \sum_{i=1}^{N} \alpha_i$$

$$L = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x_j} + \sum_{i=1}^{N} \alpha_i$$

Note that the loss (cost) function is only a sum of the dot products of pairs of samples ($\mathbf{x}_i^\top \mathbf{x}_j$)

Support Vector Machines

Optimization

- **Final Objective Function:**

$$\text{maximize}\left(L(\alpha_i) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\top}\boldsymbol{x_j} + \sum_{i=1}^{N}\alpha_i \right)$$

$$\text{Subject to: } \sum_{i=1}^{N}\alpha_i\, y_i = \mathbf{0} \text{ and } \alpha_i \geq 0$$

Note that loss (cost) function is only a sum of the dot products of pairs of samples ($\mathbf{x}_i^{\top}\mathbf{x_j}$)

- Can intuitively think of dot product as measure of *similarity*.
- **Case 1:** If two features x_i , x_j are completely dissimilar, their dot product is 0, and they don't contribute to L.
- **Case 2:** If two, features x_i , x_j are completely alike, their dot product is 1.
  - **Subcase 1:** x_i ,and x_j predict the same output value y_i (either +1 or –1). Then $y_i\ y_j\ {x_i}^T\ x_j$ is always 1, and the value of $\alpha_i\ \alpha_i\ y_i\ y_j\ {x_i}^T\ x_j$ will be positive. But this would decrease the value of L. So, the objective function downgrades similar feature vectors that make the same prediction.
  - **Subcase 2:** x_i ,and x_j make opposite predictions about the output value y_i (*i.e.*, one is +1, the other –1), but are otherwise very closely similar: then the product $\alpha_i\ \alpha_i\ y_i\ y_j\ {x_i}^T\ x_j$ is negative and we are subtracting it, so this adds to the sum, maximizing it. This is precisely the examples we are looking for: the critical ones that tell the two classes apart.

Support Vector Machines

Optimization

$$L = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\top}\boldsymbol{x}_j + \sum_{i=1}^{N}\alpha_i$$

- This optimization problem can be solved using any solver to find the values of α's and hence, the values of w_i and b
- $\boldsymbol{w} = \sum_{i=1}^{N}\alpha_i\, y_i \boldsymbol{x}_i$
- After finding the optimal weights, when we get a new unseen sample $(\boldsymbol{u})$ to classify, we can simply compute:

$$\boldsymbol{w}^{\top}\boldsymbol{u} + b$$

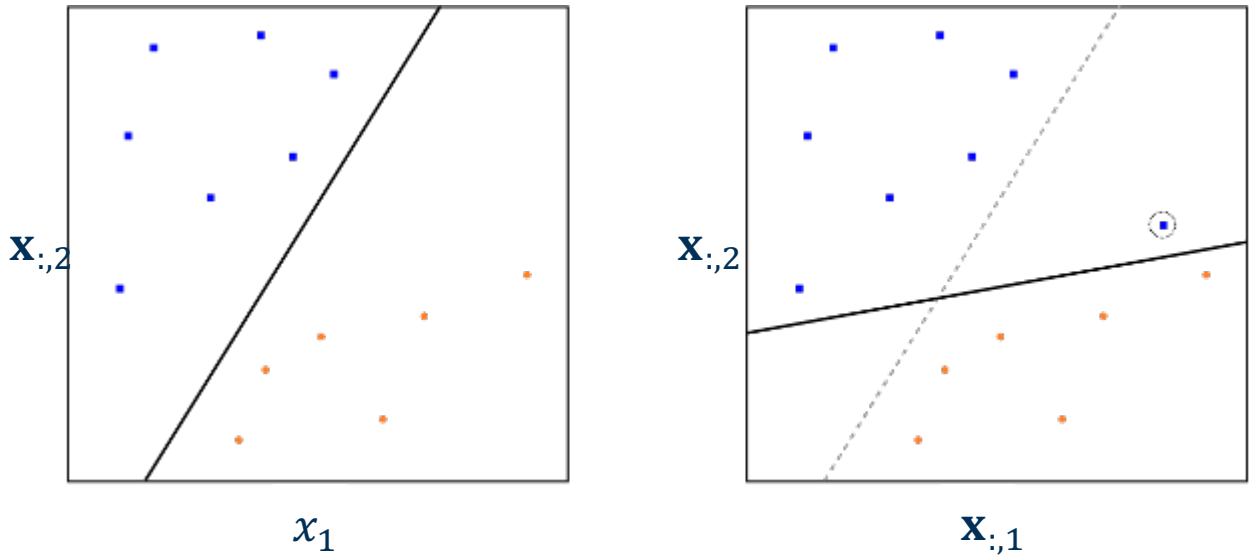
$$\left(\sum_{i=1}^{N}\alpha_i\, y_i \boldsymbol{x}_i\right)\cdot \boldsymbol{u} + \boldsymbol{b}$$

  - If $\boldsymbol{w}^{\top}\boldsymbol{u} + b > 0$, then $\boldsymbol{u}$ is classified as class 1 (+1)
  - If $\boldsymbol{w}^{\top}\boldsymbol{u} + b < 0$, then $\boldsymbol{u}$ is classified as class 2 (-1)
- Or simply, class = sgn $(\boldsymbol{w}^{\top}\boldsymbol{u} + b)$

Support Vector Machines

Classifiers

- Addition of a single point dramatically changes the MMH line



Support Vector Machines

Classifiers

- Clearly, the case of perfect separability is an ideal one. Most "real world" datasets will not have such perfect separability via a linear hyperplane
- However, if there is no separability then we are unable to construct a MMC by the optimization procedure above. So, how do we create a form of separating hyperplane?
- Essentially, we must relax the requirement that a separating hyperplane will perfectly separate every training observation on the correct side of the line (*i.e.* guarantee that it is associated with its true class label), using what is called a **soft margin**. This motivates the concept of a **support vector classifier** (SVC).

Support Vector Machines

Soft Margin Classifiers

- Maximum Margin Classifiers (MMC) can be extremely sensitive to the addition of new training observations.
- We could consider a classifier based on a separating hyperplane that:
  - Does have a greater robustness to the addition of *new* individual observations
  - Has a better classification on *most* of the training data.
- This comes at the expense of some misclassification of a few training data.

$$minimize \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{Subject to } y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) \geq 1 - \xi_i$$

Recall

$$minimize \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{Subject to } y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) - 1 \geq 0$$

Support Vector Machines

Soft Margin Classifiers

- Maximum Margin Classifiers (MMC) can be extremely sensitive to the addition of new training observations.
- We could consider a soft margin classifier based on a separating hyperplane that:
  - Does have a greater robustness to the addition of *new* individual observations
  - Has a better classification on *most* of the training data.
- This comes at the expense of some misclassification of a few training data.

$$minimize \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

regularization parameter

$$\text{Subject to } y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

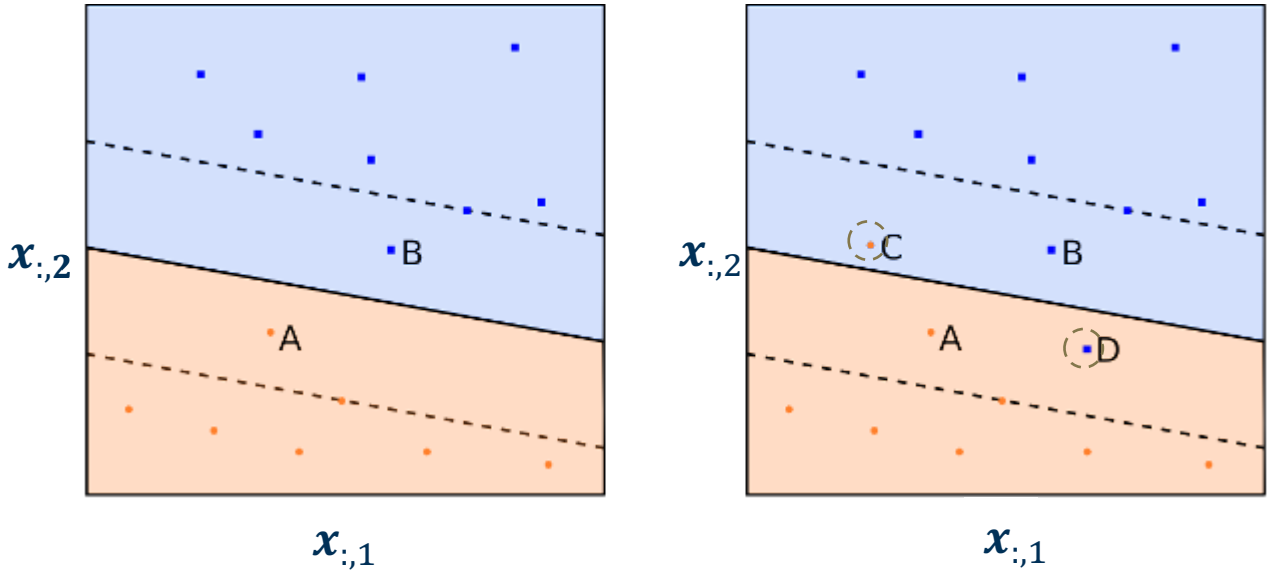
- The above objective could be written as following:

L_2 norm regularization $\longrightarrow$ $minimize \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\max\left(0, 1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} + b)\right)$ ← ***Margin-Perceptron cost***

Support Vector Machines

Soft Margin Classifiers

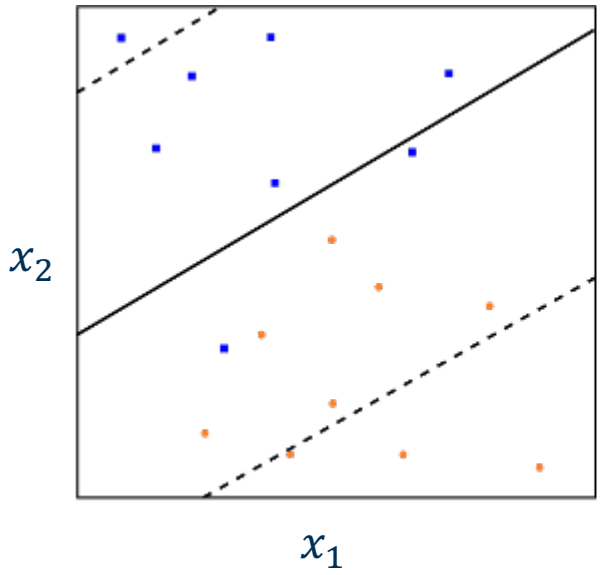
- A *Soft Margin Classifier* allows some observations to be on the incorrect side of the margin (or hyperplane), hence it provides a "soft" separation.



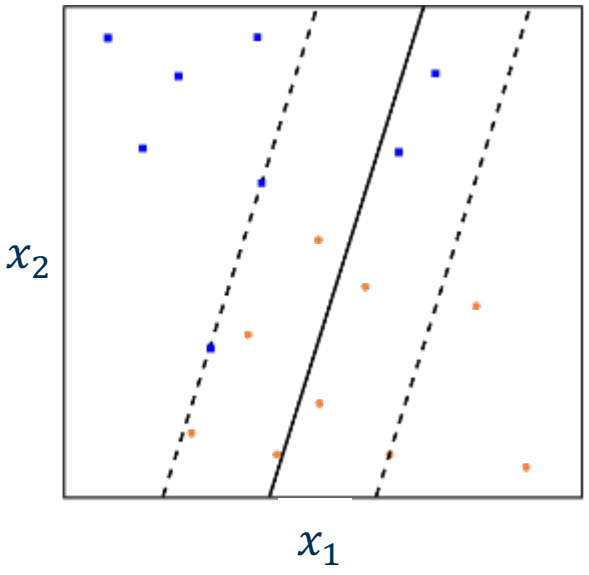
Support Vector Machines

Soft Margin Classifiers

- Different values of the tuning parameter C



smaller C



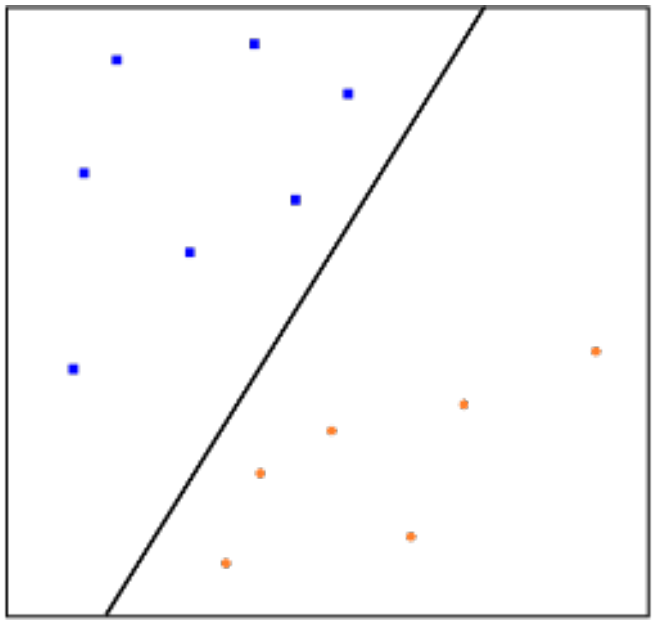
larger C

Larger C = Higher penalty for misclassification, results in narrower margin

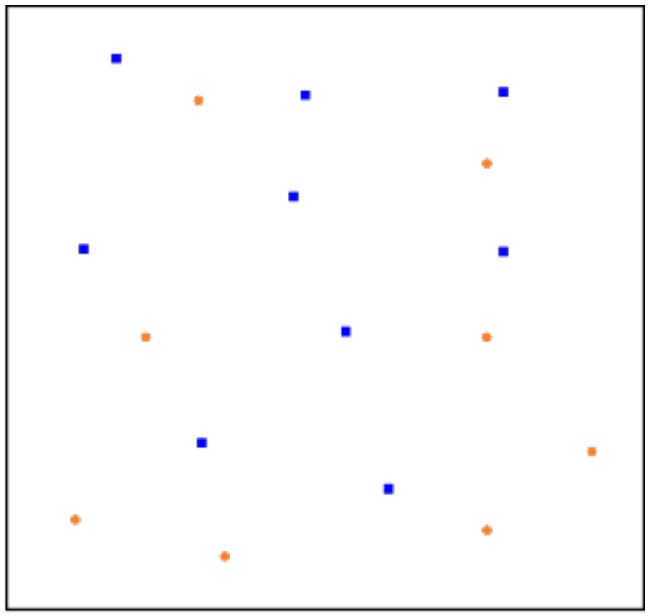
Support Vector Machines

Non-Linear Soft Margin Classifiers

- What if our data samples are not linearly separable?



Linearly Separable

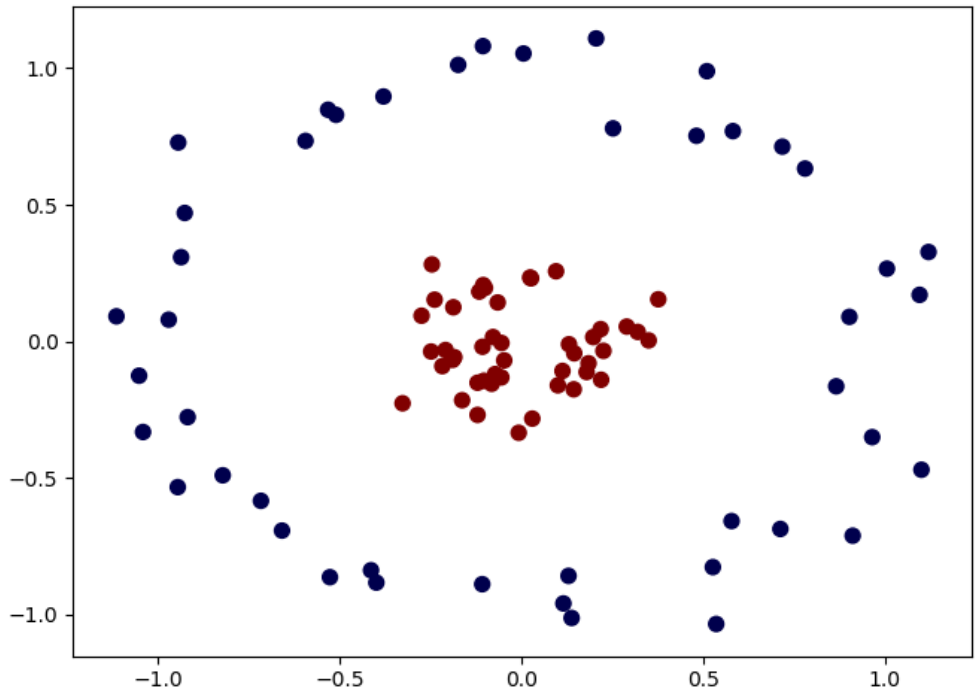


NOT Linearly Separable

Support Vector Machines

Non-Linear Soft Margin Classifiers

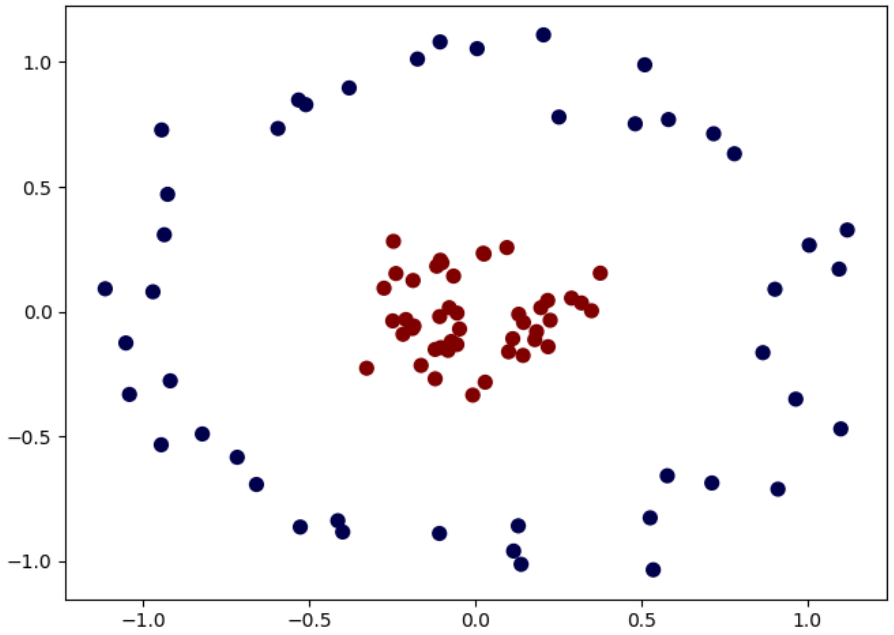
- There is no single hyperplane that can correctly separate the classes

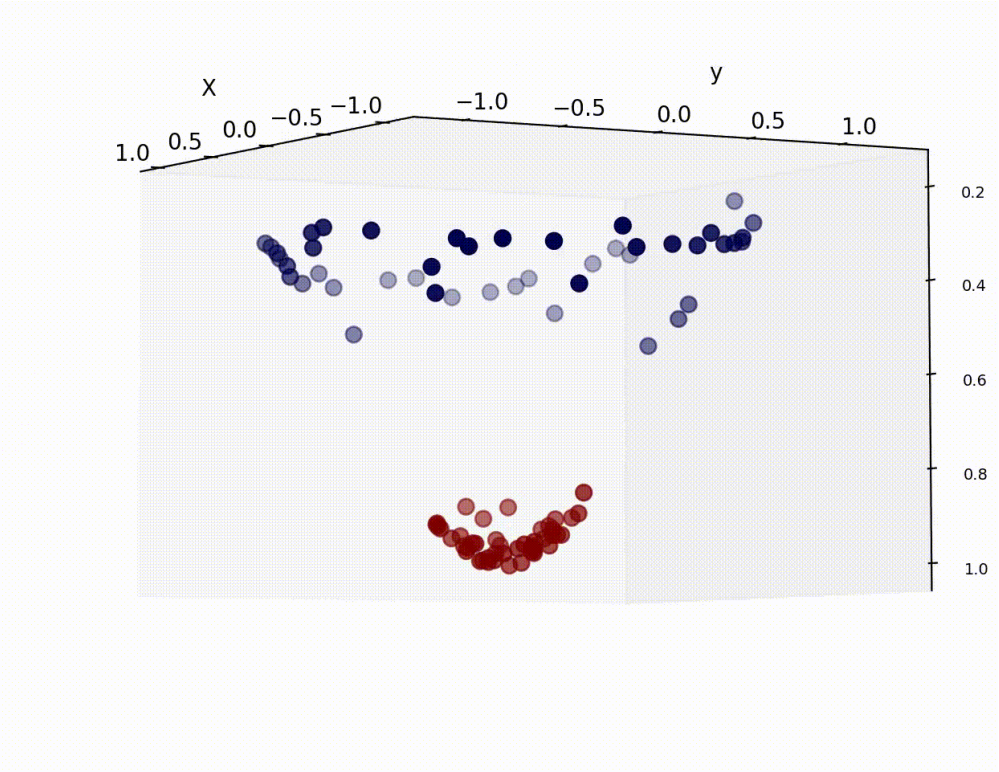


Support Vector Machines

Non-Linear Soft Margin Classifiers

- But we can transform to another subspace where the data is linearly separable.

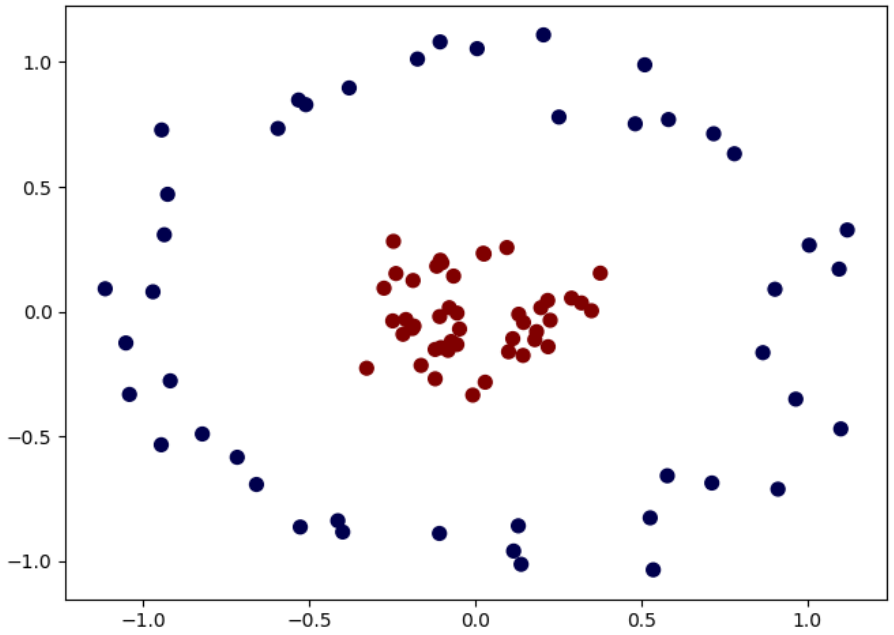


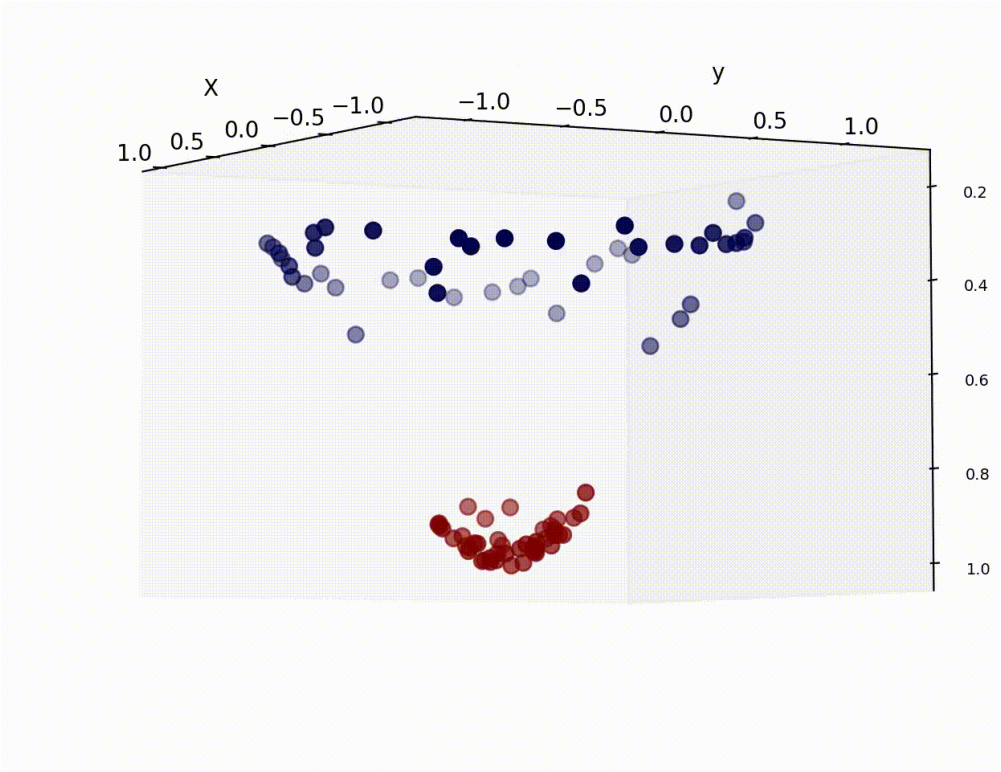


Support Vector Machines

Kernel Trick

- We can use kernel tricks $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$ to transform linearly non-separable data to another subspace where the data is linearly separable.





Support Vector Machines

Kernel Trick

- Recall the loss function of the MMC

$$L = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^{\top}\mathbf{x_j} + \sum_{i=1}^{N}\alpha_i$$

- Let us define the transform $\phi(\mathbf{x}_i)$

$$L = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \phi(\mathbf{x}_i)^{\top}\phi(\mathbf{x}_j) + \sum_{i=1}^{N}\alpha_i$$

- Let us define the kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^{\top}\phi(\mathbf{x}_j)$$

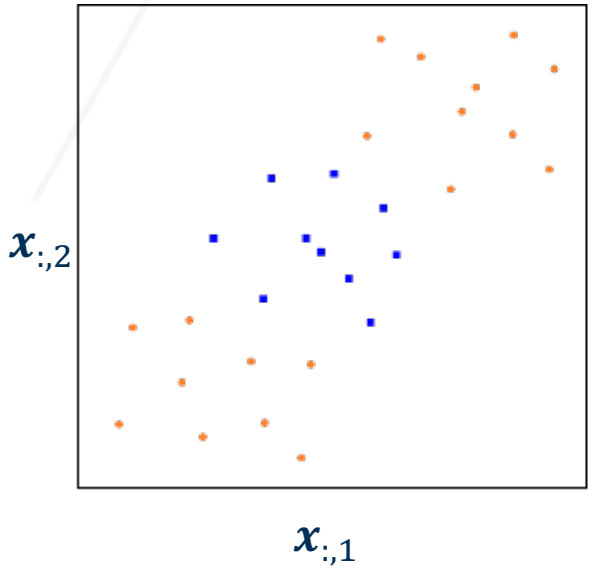
- Now, we don't need to know the transform ϕ, we can simply use a kernel K which is much easier.

$$L = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^{N}\alpha_i$$

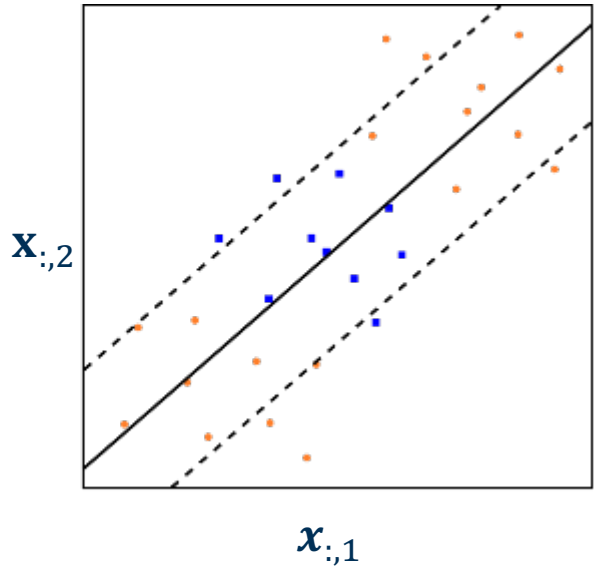
Support Vector Machines

Kernel Trick

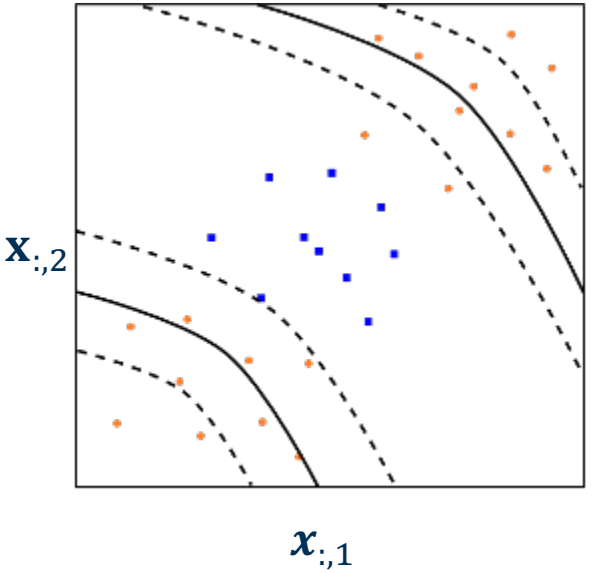
- Example kernels
  - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^\top \boldsymbol{x}_j + 1)^p$; polynomial kernel
  - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\frac{1}{2\sigma^2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)}$; Gaussian kernel
  - $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)}$; Radian Basis Function (RBF)



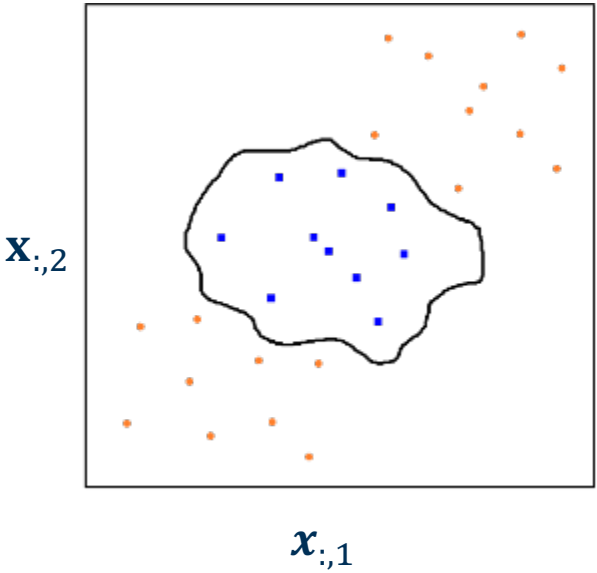
data



Linear SVM



Polynomial Kernel

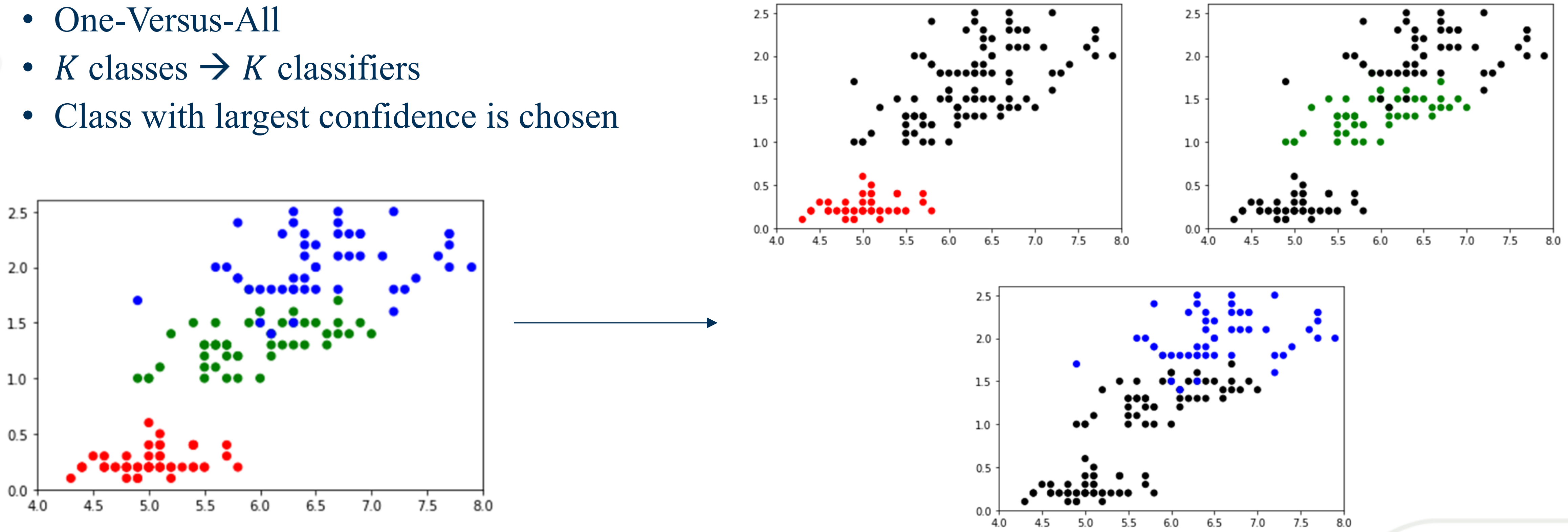


RBF Kernel

Support Vector Machines

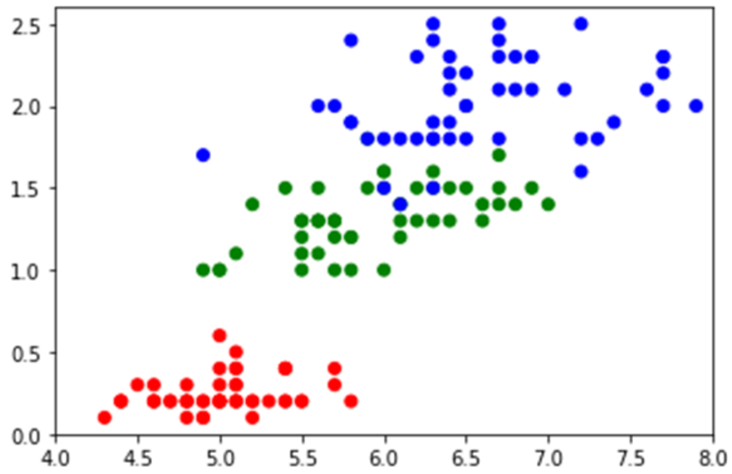
Multi-Class SVM

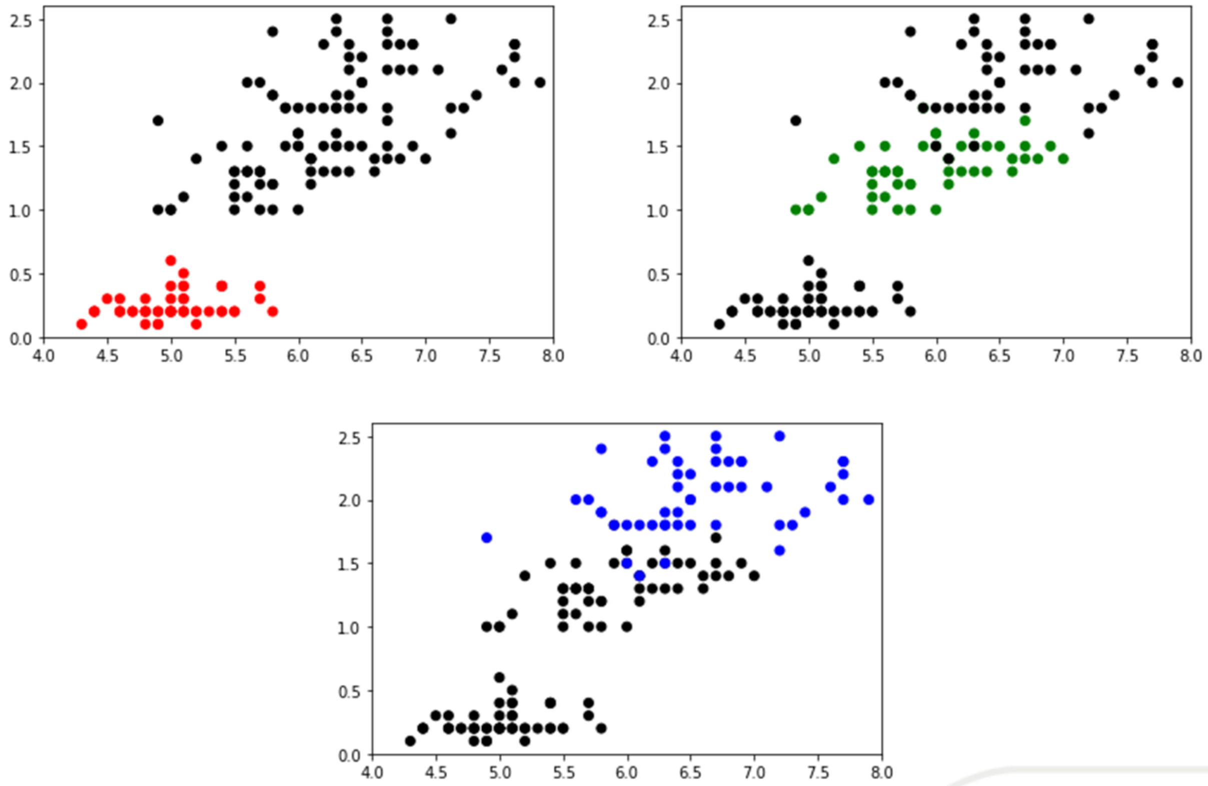
- SVMs are by default binary. However, they can be used to perform multi-class classification.
- Strategy 1:
  - One-Versus-All
  - K classes → K classifiers
  - Class with largest confidence is chosen



Support Vector Machines
Multi-Class SVM

- One-Versus-All
- K classes → K classifiers
- Issue:
  - Unbalanced data

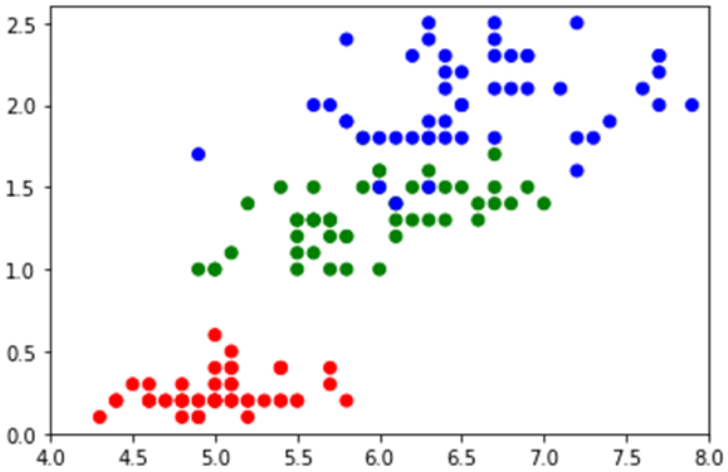


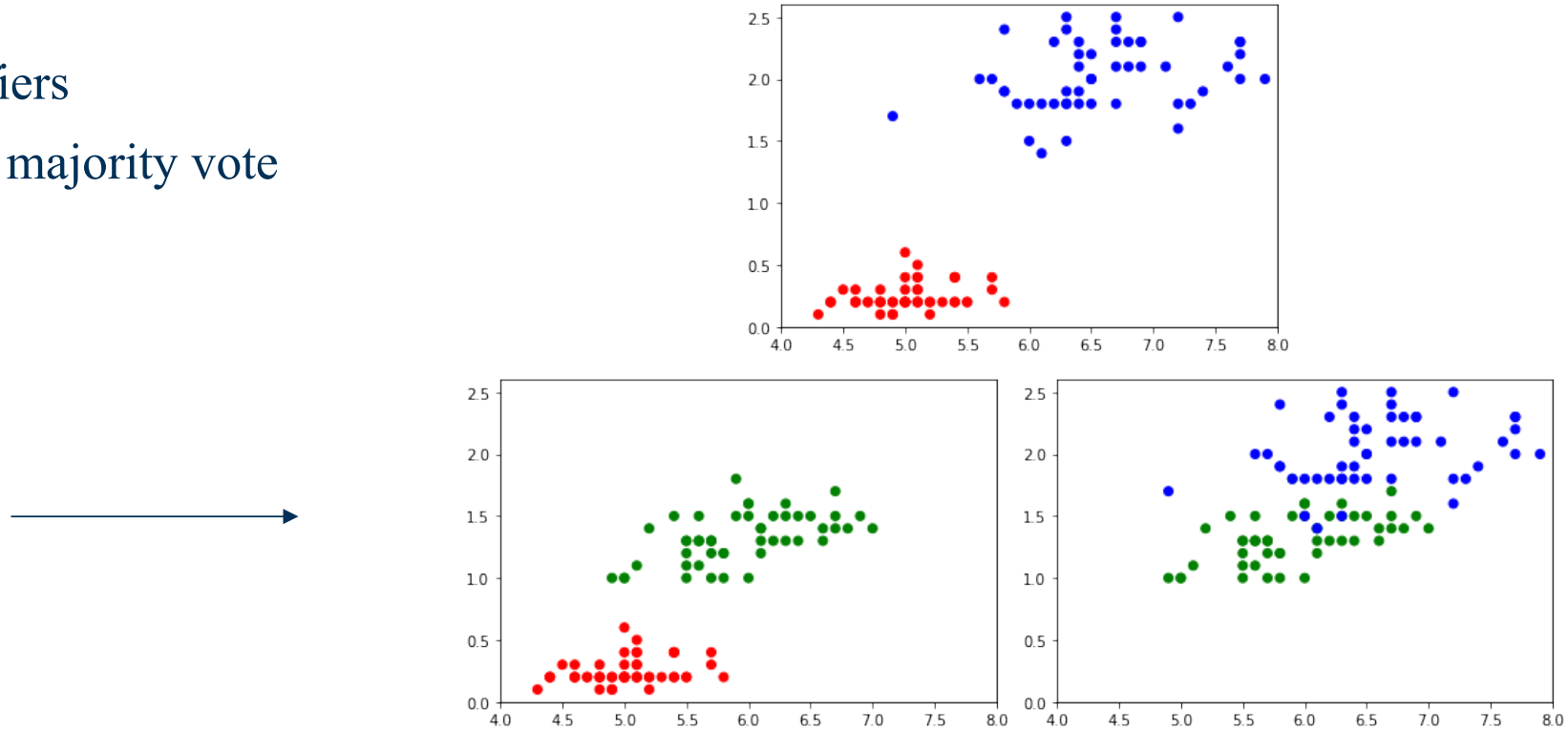


Support Vector Machines

Multi-Class SVM

- Strategy 2:
  - One-Versus-One
  - K classes → $\frac{K(K-1)}{2}$ classifiers
  - Then class is chosen using majority vote

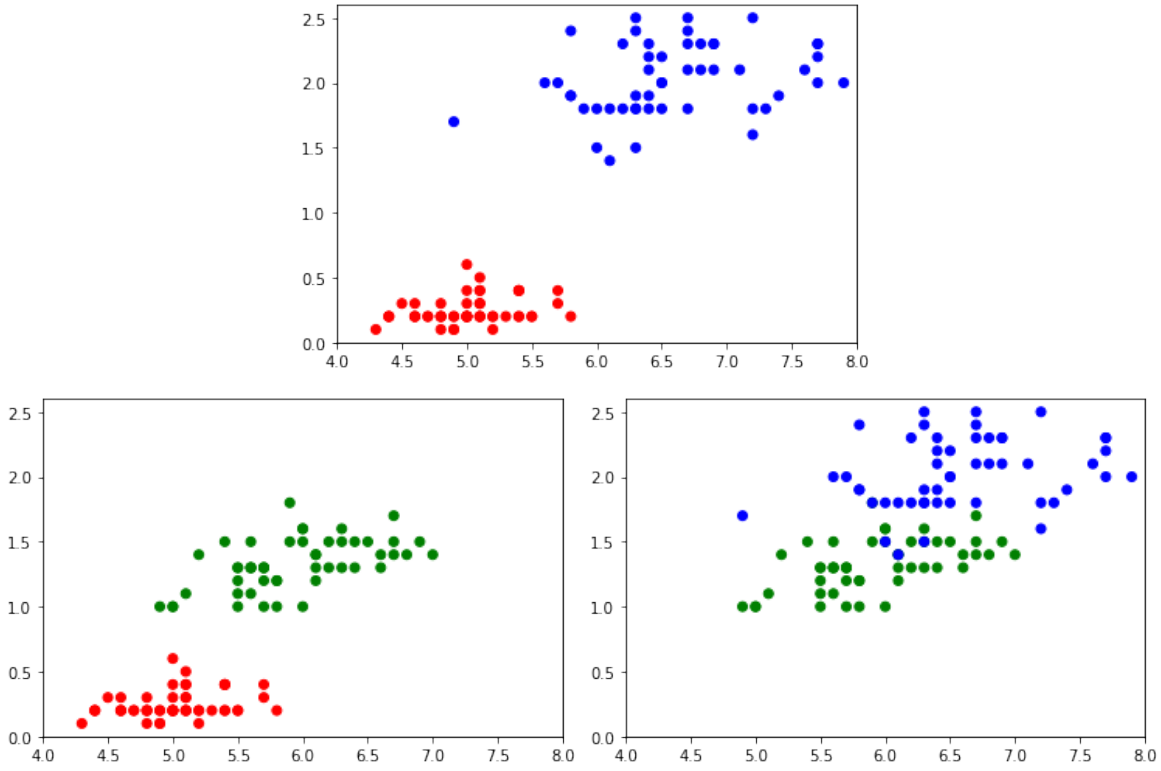




Support Vector Machines

Multi-Class SVM

- One-Versus-One
- K classes → $\frac{K(K-1)}{2}$ classifiers
- Issue:
  - Computationally expensive



Support Vector Machines

Pros and Cons

- Advantages
  - High-Dimensionality
  - Memory Efficiency
  - Versatility
- Disadvantages
  - Feature dimension $P >>$ the number of training samples N
  - Non-probabilistic

Classifier Comparison

Methods	Assumptions on Feat. Dist.	Feat. Normalization	Cost Function	Regularization	Linear Classifier	Prob. View of Prediction	Generative/Discriminative	Parametric/ Non-parametric	Overfitting
Logistic Regression	No	Required	BCE (convex)	Additional term	Linear	Yes	Discriminative	Parametric	Not often
K Nearest Neighbors	No	Required	N/A	N/A	Non-linear	N/A	Discriminative	Non-parametric	when k is too small
Decision Trees	No	Not Required	N/A	N/A	Non-linear	N/A	Discriminative	Non-parametric	with large depth
Support Vector Machines	No	Required	Hinge (convex)	C (control robustness)	Linear/ Non-linear(kernel)	N/A	Discriminative	Parametric	Not often
Naïve Bayes	Conditional independent	Not Required	N/A	N/A	Non-linear /Linear (Gaussian)	Yes	Generative	Parametric	Not often
Artificial Neural Networks	No	Required	Non-convex	Additional term	Non-linear	Yes	Discriminative/ Generative	Parametric	with many layers

Appendix A: Notations

- x_i: a single feature
- $\boldsymbol{x}_i$: feature vector (data sample)
- $\boldsymbol{X}$: matrix of feature vectors (dataset)
- N: number of data samples
- m: degree of polynomial
- P: number of features in a feature vector
- θ_i: a single model coefficient (parameter)
- $\boldsymbol{\theta}$: coefficient vector
- ε: error margin
- α: learning rate
- γ: bias factor

- Bold letter/symbol: vector
- Bold capital letters/symbol: matrix