CSE 375 Machine Learning and Pattern Recognition

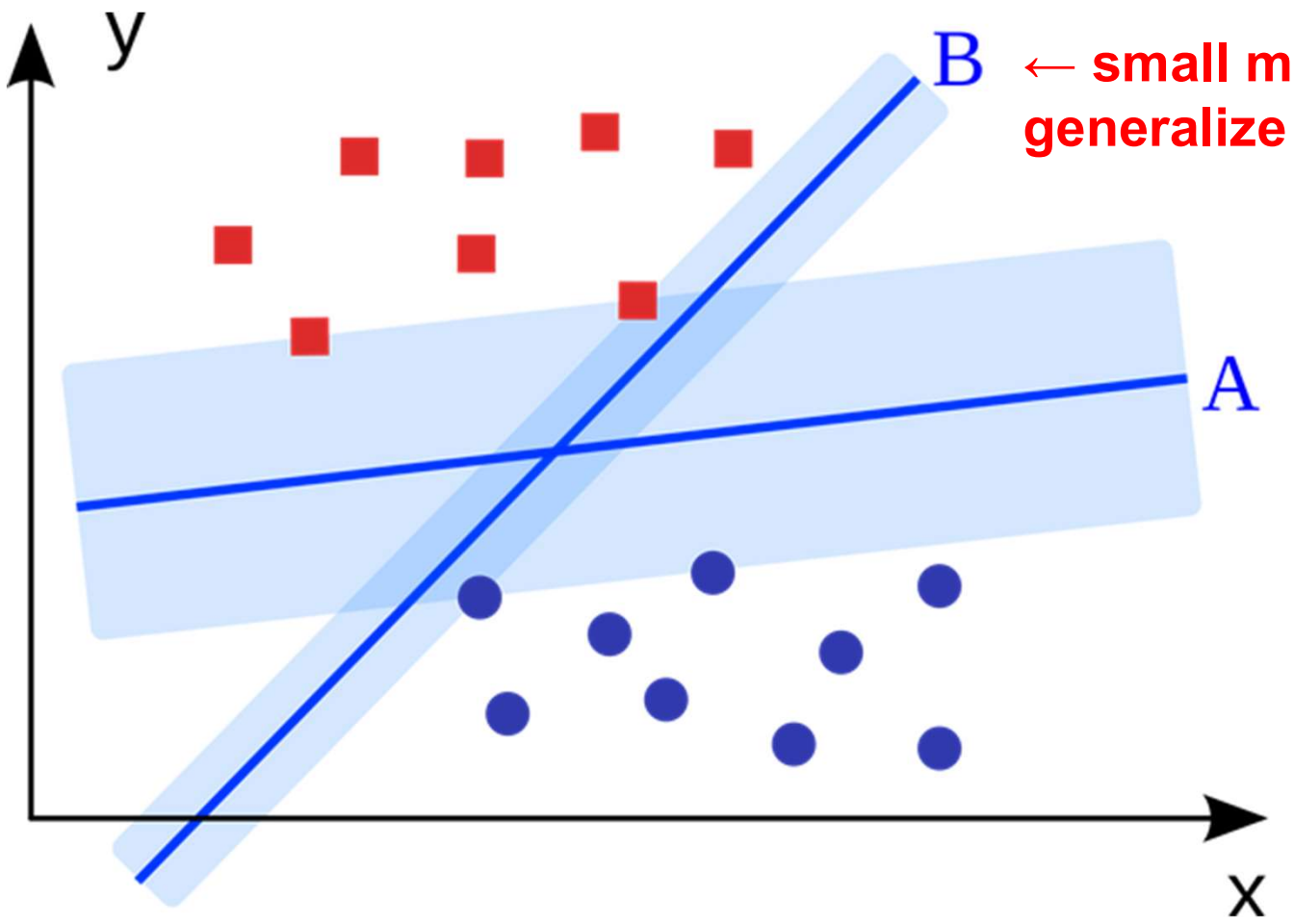
09. Support Vector Machines

Slides Credit: N. Rich Nguyen

Contents

1. Understand **large margin** classification
2. Derive **objective function** for Linear SVM
3. Handle **soft-margin** classification with Hinge Loss

Large Margin Classifier

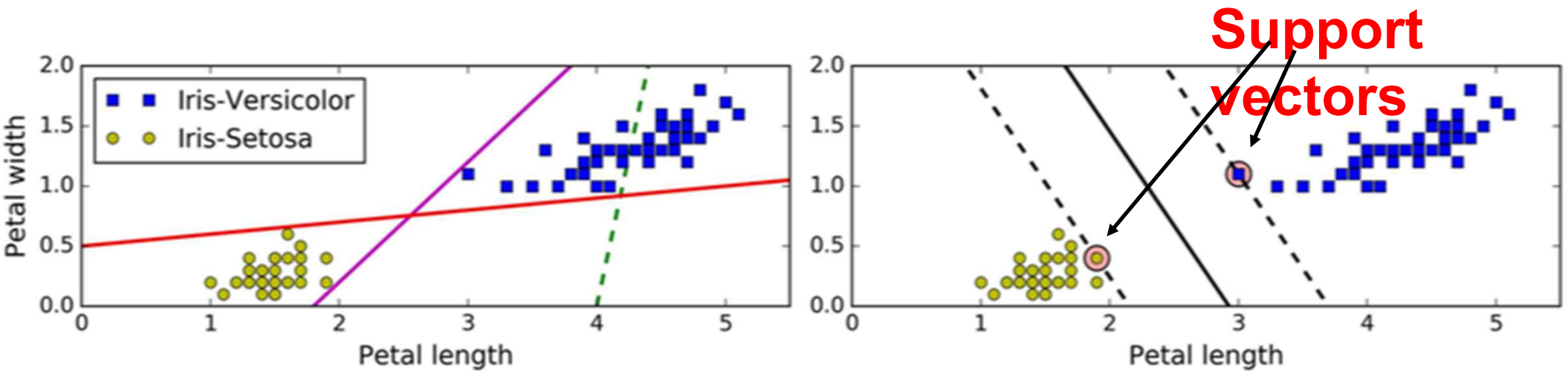


← small margin does not generalize well

← more robust classification with larger margin

Linear SVM

Linear SVM classifier separates two classes but also stays as far away from the closest training samples as possible → **maximized the margin**



Decision boundary is fully determined (or supported) by the samples located "on the edge of the street" → **support vectors**

Introducing Support Vector Machine

- A large margin classifier
- Capable of non-linear **classification**, **regression**, and **outlier detection**
- Particularly suited for classification of **complex** and **mid-sized datasets**
- Those who are interested in ML should have Support Vector Machine (SVM) in their toolbox

History of SVM

- First introduced in 1992 inspired from a statistical learning theory*
- Became popular because of its success in MNIST digit recognition (1994)
- Had lots successful applications in Computer Vision, Text Categorization, Ranking, Time Series Analysis, and BioInformatics, ect.
- Regarded as an important example of "**kernel methods**", arguably the hottest area in machine learning in the early 2000s

* B.E. Boser et al. A Training Algorithm for Optimal Margin Classifiers. Proceedings of the Fifth Annual Workshop on Computational Learning Theory 5 144-152, Pittsburgh, 1992.

The SVM Model

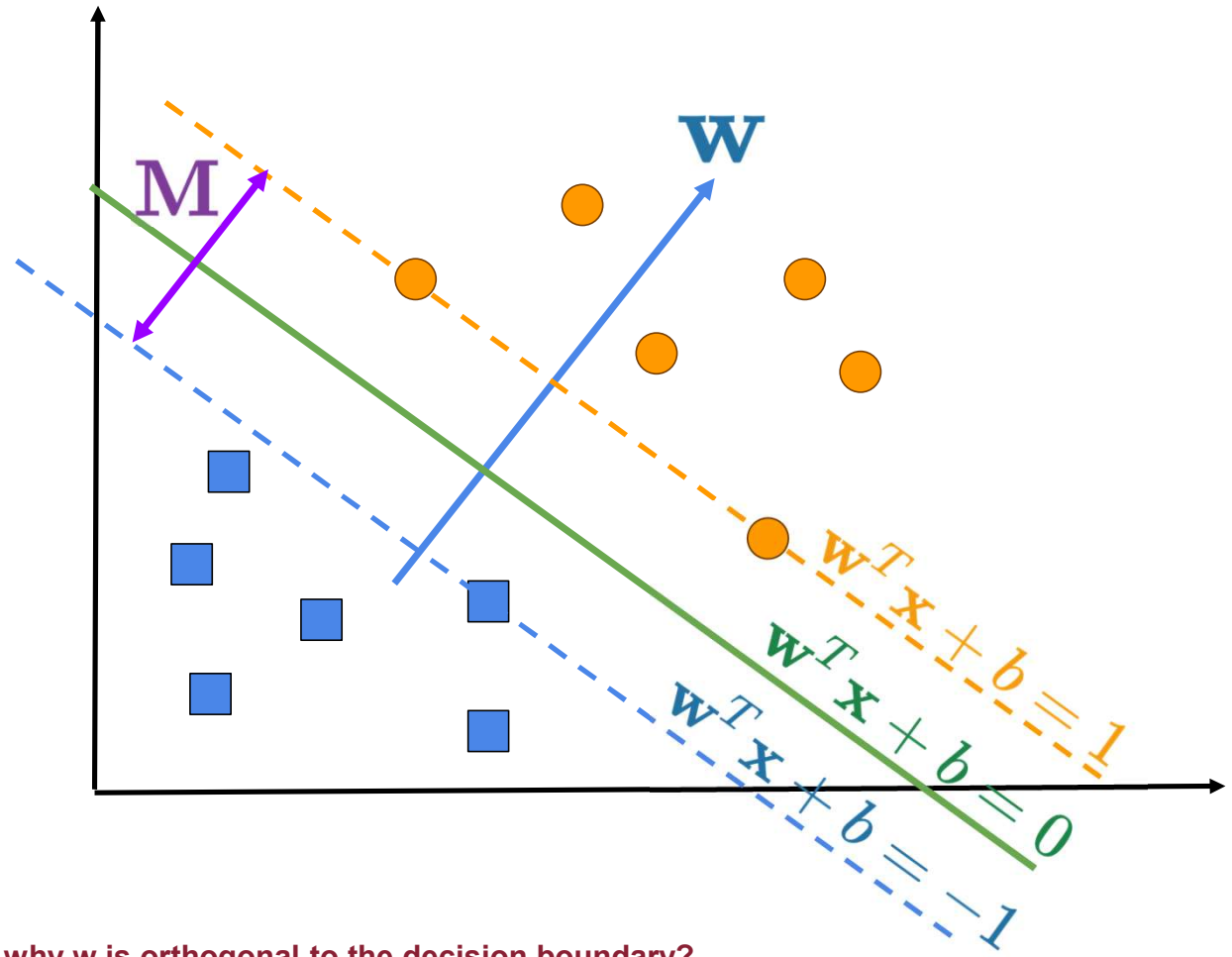
Notation **change**:

- Bias term will be called **b** (no longer $\boldsymbol{\theta}_0$)
- Feature weight vector will be call **w** (no longer $\boldsymbol{\theta}$)

Linear SVM classifier model predicts the class of sample **x** by computing:

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$$

Geometry Interpretation



Class 1

Class -1

Decision Boundary

Weight Vector

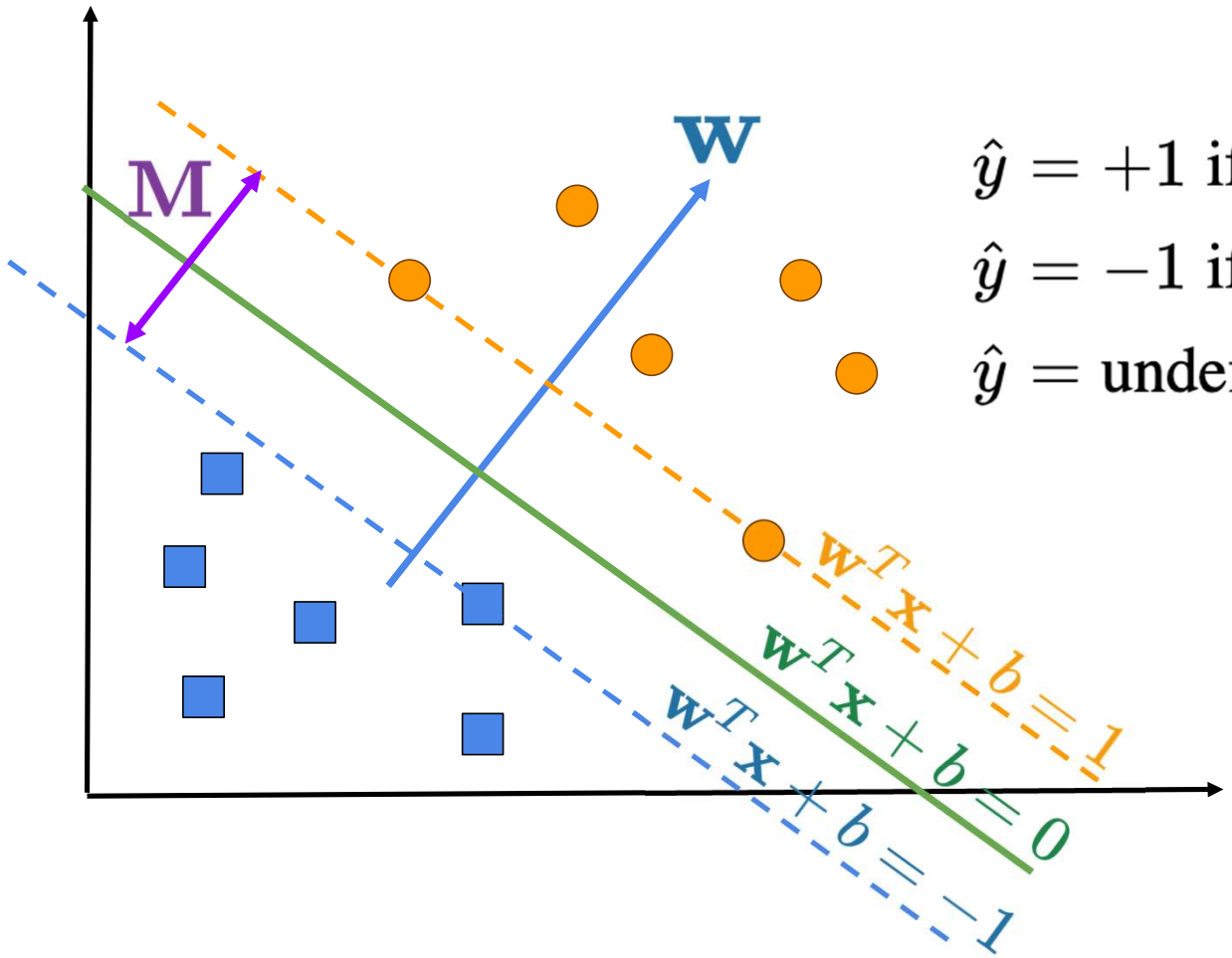
+1 Plane

-1 Plane

Margin

*See why w is orthogonal to the decision boundary?

Predicting a label



$$\hat{y} = +1 \text{ if } \mathbf{w}^\top \mathbf{x} + b \geq +1$$

$$\hat{y} = -1 \text{ if } \mathbf{w}^T \top \mathbf{x} + b \leq -1$$

$$\hat{y} = \text{undefined if } -1 \leq \mathbf{w}^\top \mathbf{x} + b \leq +1$$

Classifying new instances

- In order to classify a new instance z we just have to compute

$$sign(w^T z + b)$$

We hope that a classifier that has a large maximal margin classifier margin on the training data will also have a large margin on the test data, and hence will classify the test observations correctly

An Example in Python

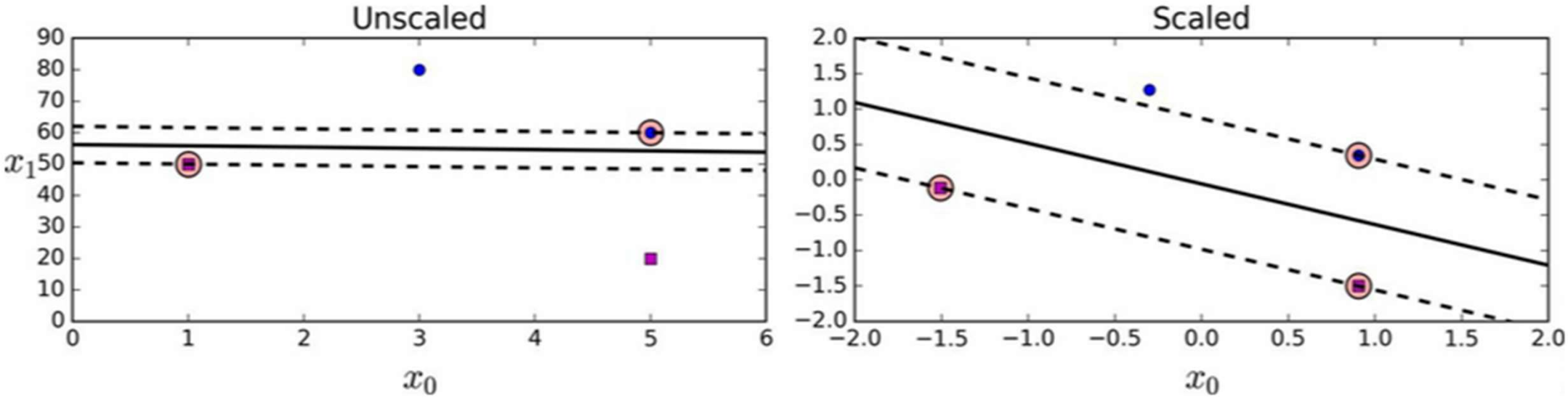
```
import numpy as np
from sklearn import datasets
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVC

iris = datasets.load_iris()
X = iris["data"][:, (2, 3)]  # petal length, petal width
y = (iris["target"] == 2).astype(np.float64)  # Iris-Virginica

svm_clf = Pipeline([
        ("scaler", StandardScaler()),
        ("linear_svc", LinearSVC(C=1, loss="hinge", random_state=42)),
    ])

svm_clf.fit(X, y)
```

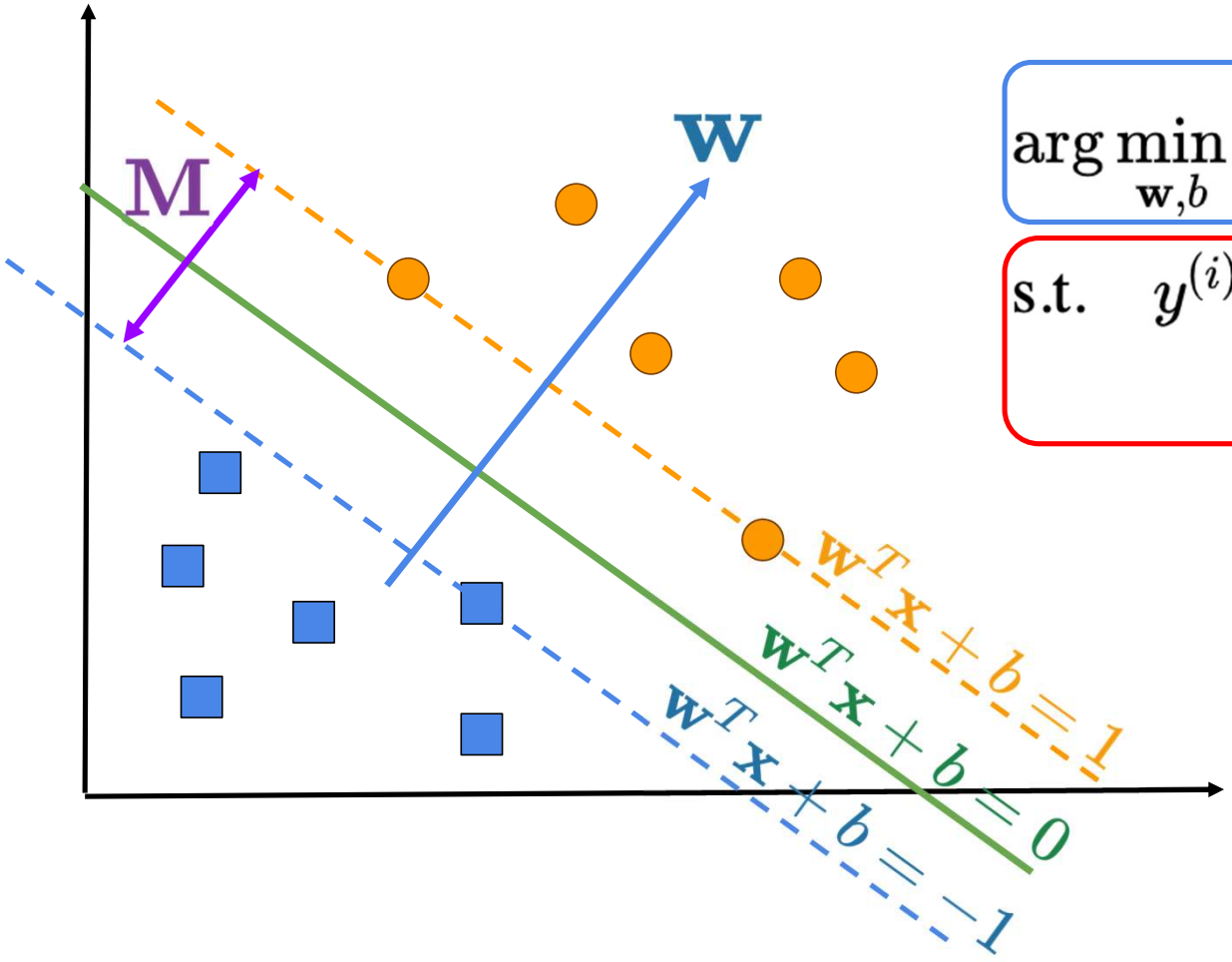
SVM margin is sensitive to feature scales



Make sure to use feature scaling (with StandardScaler)

Formulating SVM Objective Function

SVM Objective Function (aka how to find w, b)



$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$ SVM Objective

$$\text{s.t.} \quad y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1 \text{ for } i = 1, \ldots, m$$

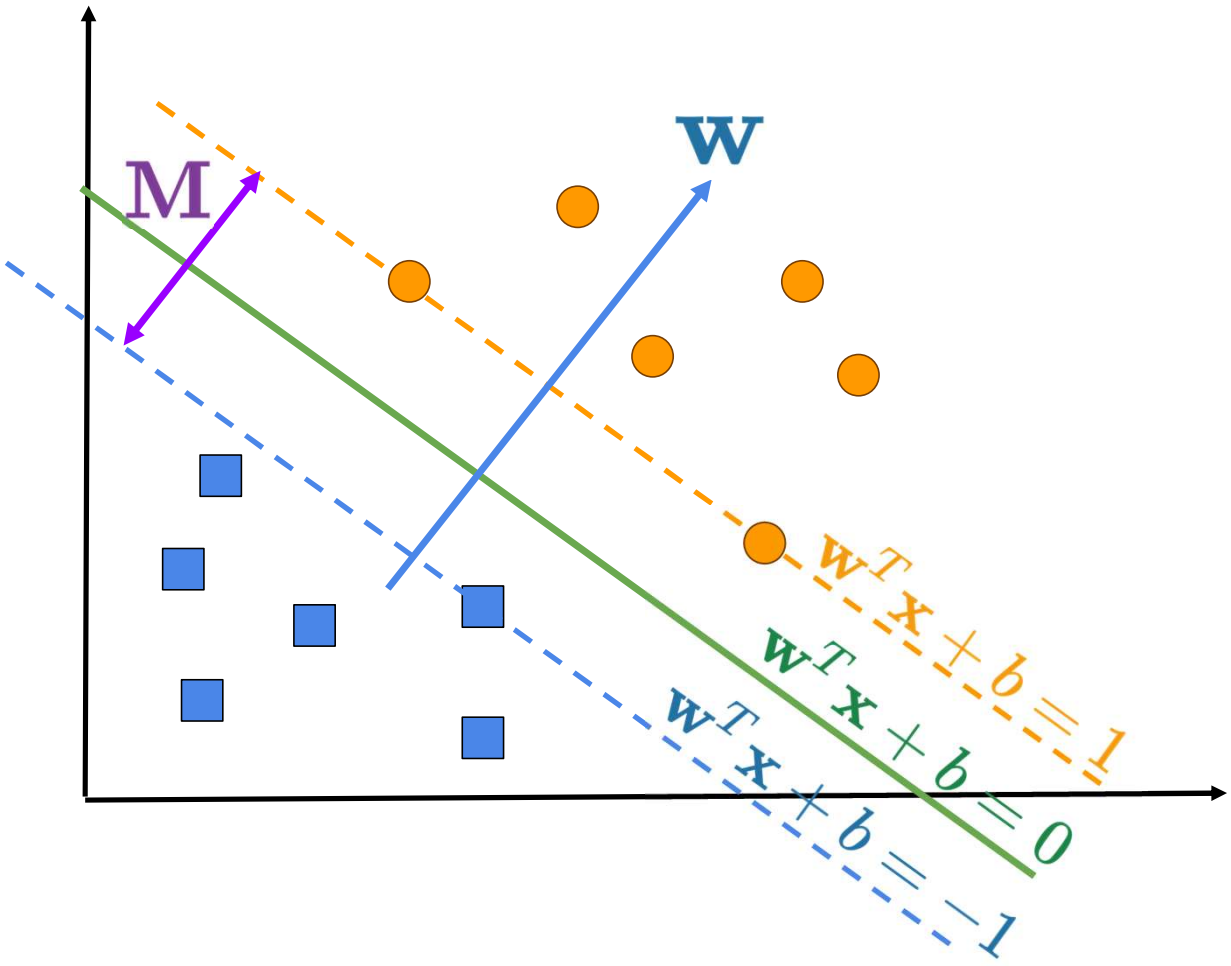
SVM Constraints

***In English**, this means trying to maximize the margin while correctly classifying all training examples.*

SVM Constraints

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$

$$\text{s.t.} \quad y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 \text{ for } i = 1, \ldots, m$$



For all x in positive class:

$$y = +1 \text{ if } \mathbf{w}^\top \mathbf{x} + b \geq +1$$

$$\Rightarrow y(\mathbf{w}^\top \mathbf{x} + b) \geq 1$$

For all x in negative class:

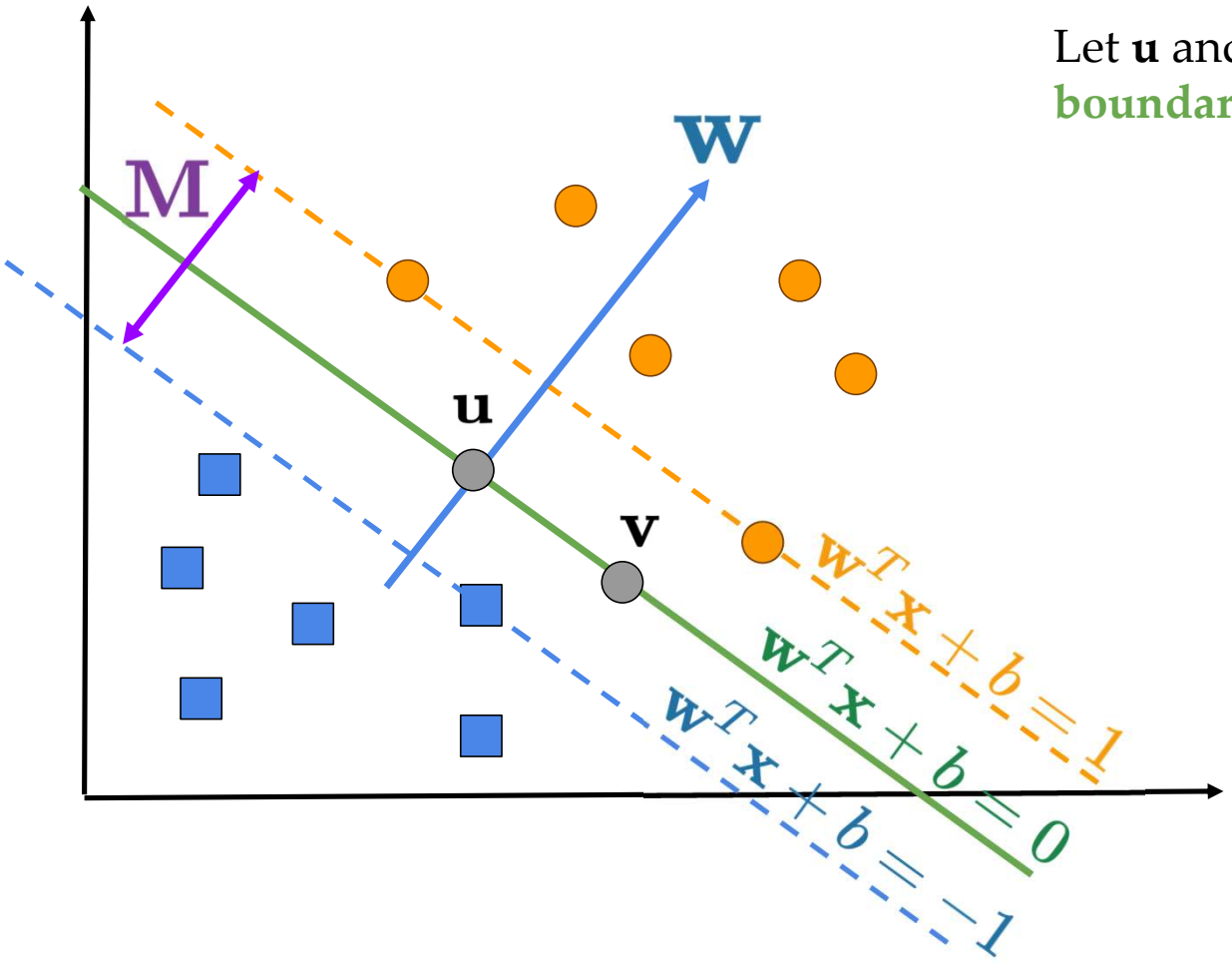
$$y = -1 \text{ if } \mathbf{w}^\top \mathbf{x} + b \leq -1$$

$$\Rightarrow y(\mathbf{w}^\top \mathbf{x} + b) \geq 1$$

Therefore, for all $x^{(i)}$, $y^{(i)}$ in train set:

$$y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1$$

Why w is orthogonal to the decision boundary?

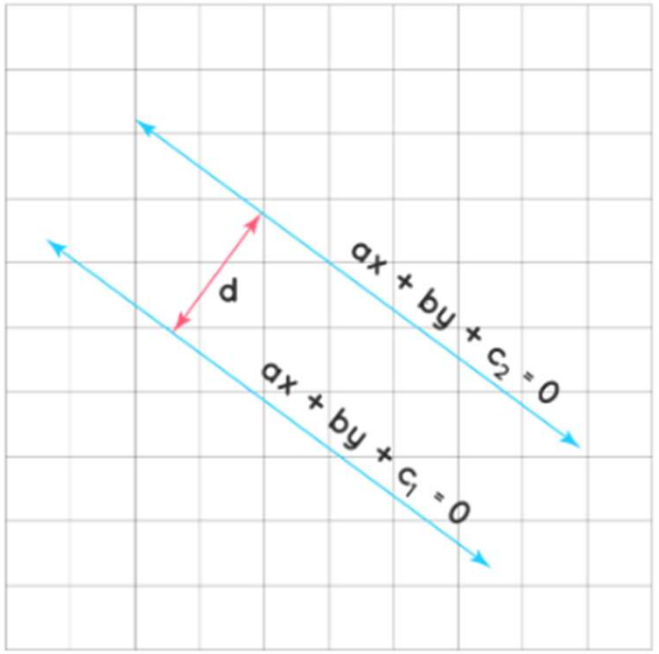


Let **u** and **v** be two points on the **decision boundary**, then:

$$\mathbf{w}^\top \mathbf{u} + b = 0$$
$$\mathbf{w}^\top \mathbf{v} + b = 0$$
$$\mathbf{w}^\top (\mathbf{u} - \mathbf{v}) = 0$$
$$\rightarrow \mathbf{w} \perp (\mathbf{u} - \mathbf{v})$$

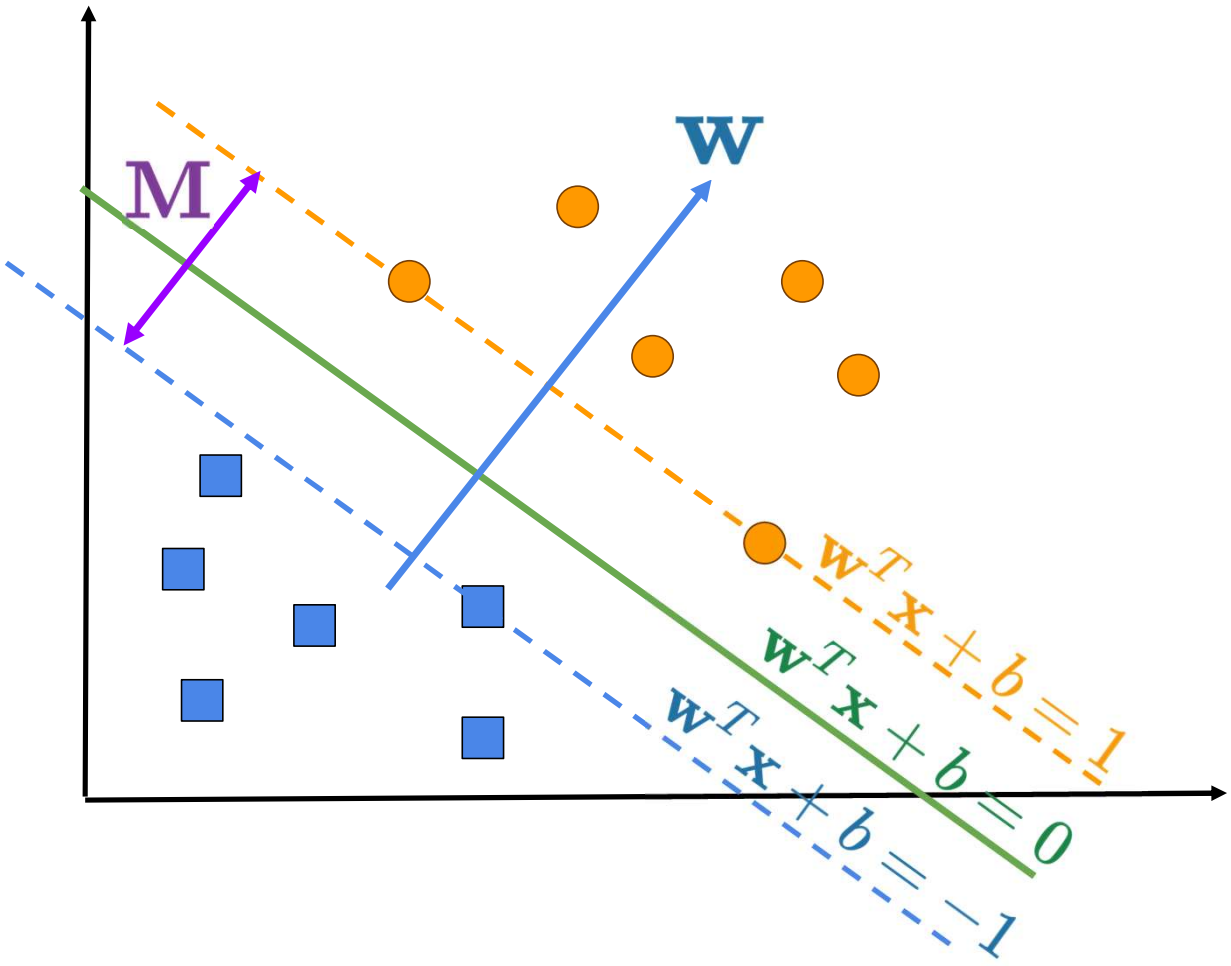
Distance between two parallel lines

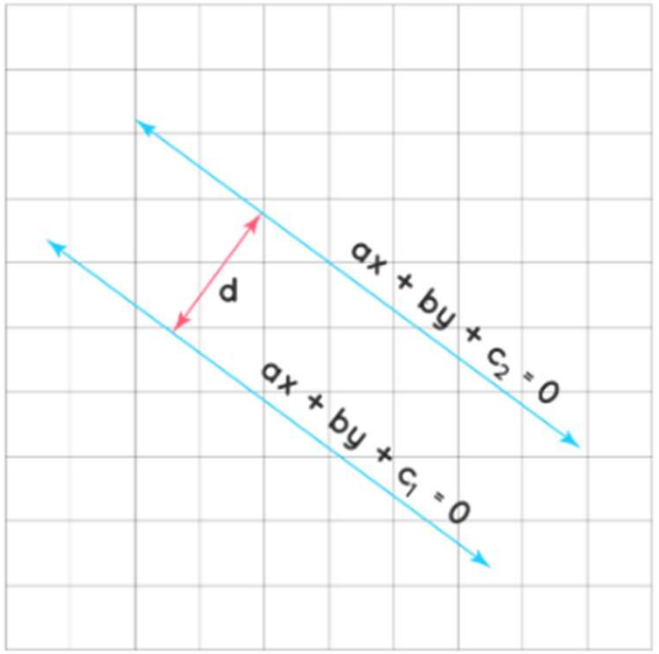
$$d = \frac{|c_2 - c_1|}{\sqrt{a^2 + b^2}}$$



$$d = \frac{|c_2 - c_1|}{\sqrt{a^2 + b^2}}$$

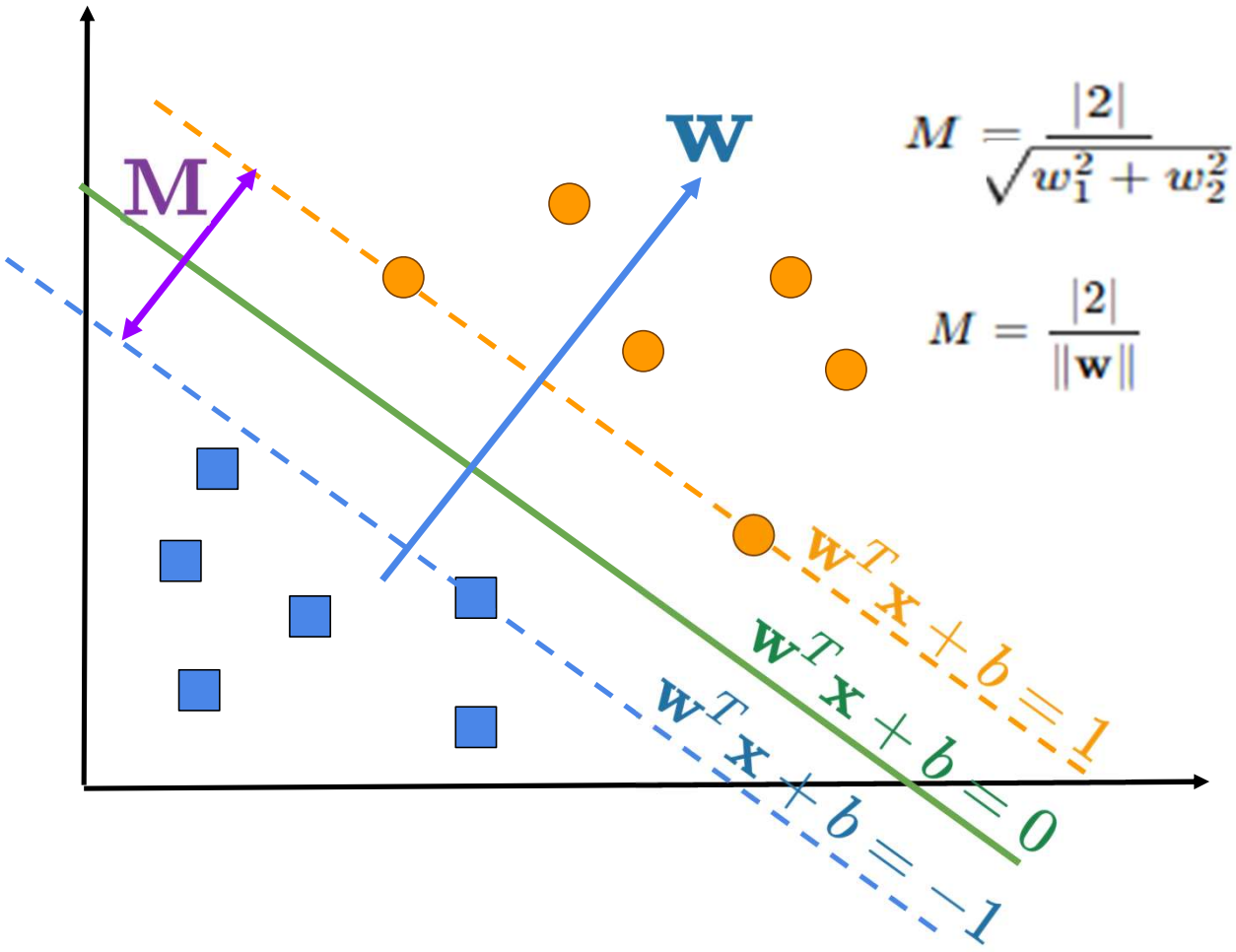
Distance between two parallel lines

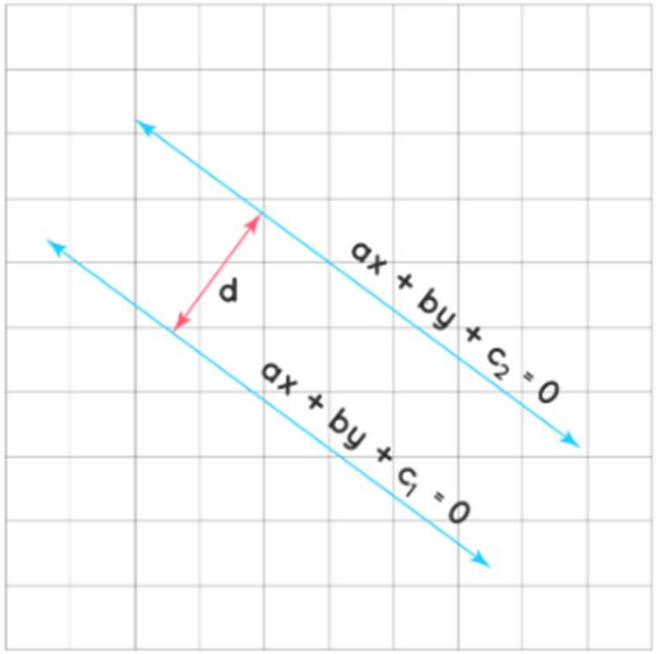




$$d = \frac{|c_2 - c_1|}{\sqrt{a^2 + b^2}}$$

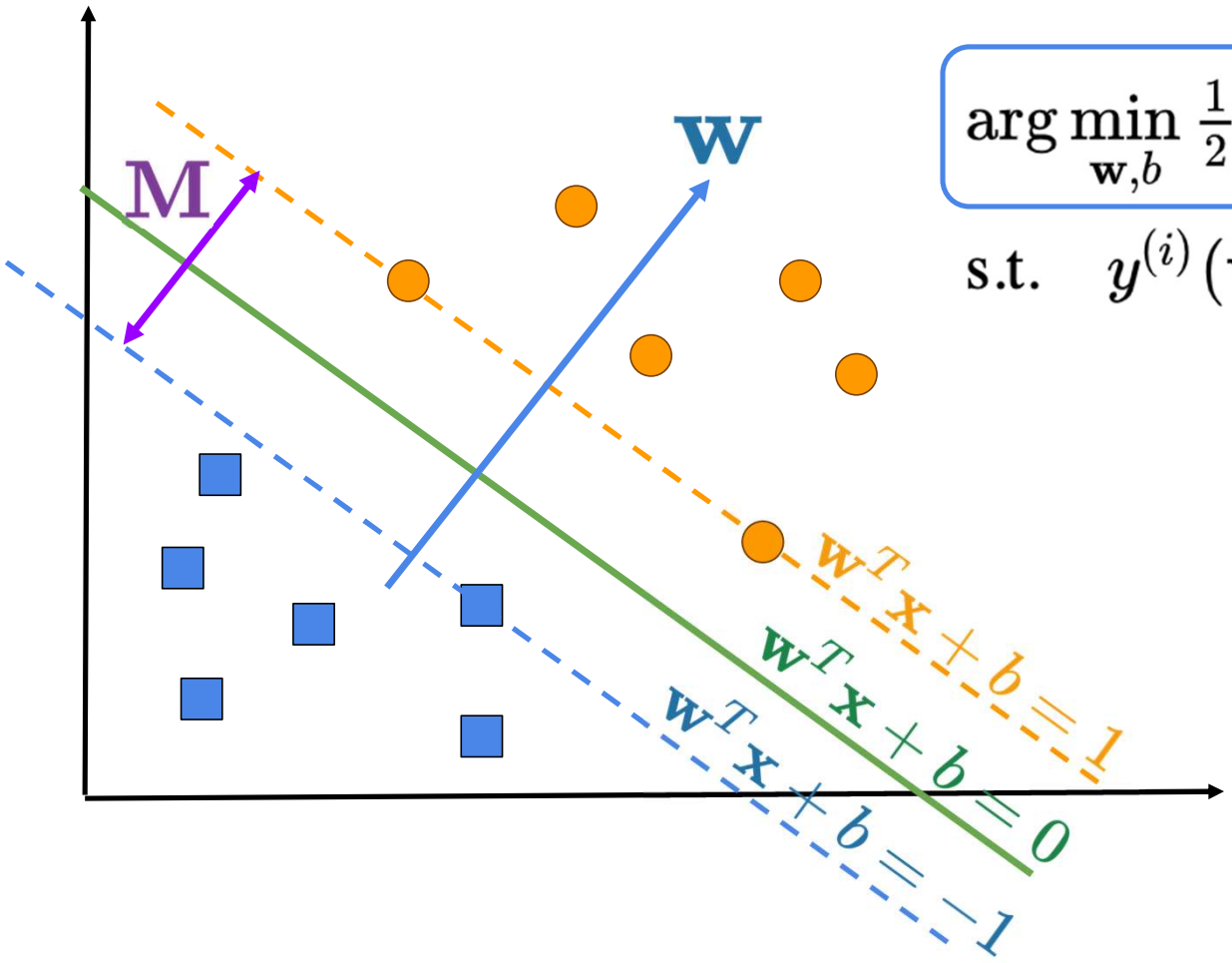
Distance between two parallel lines





$$d = \frac{|c_2 - c_1|}{\sqrt{a^2 + b^2}}$$

Revisit: SVM objective (margin maximization)



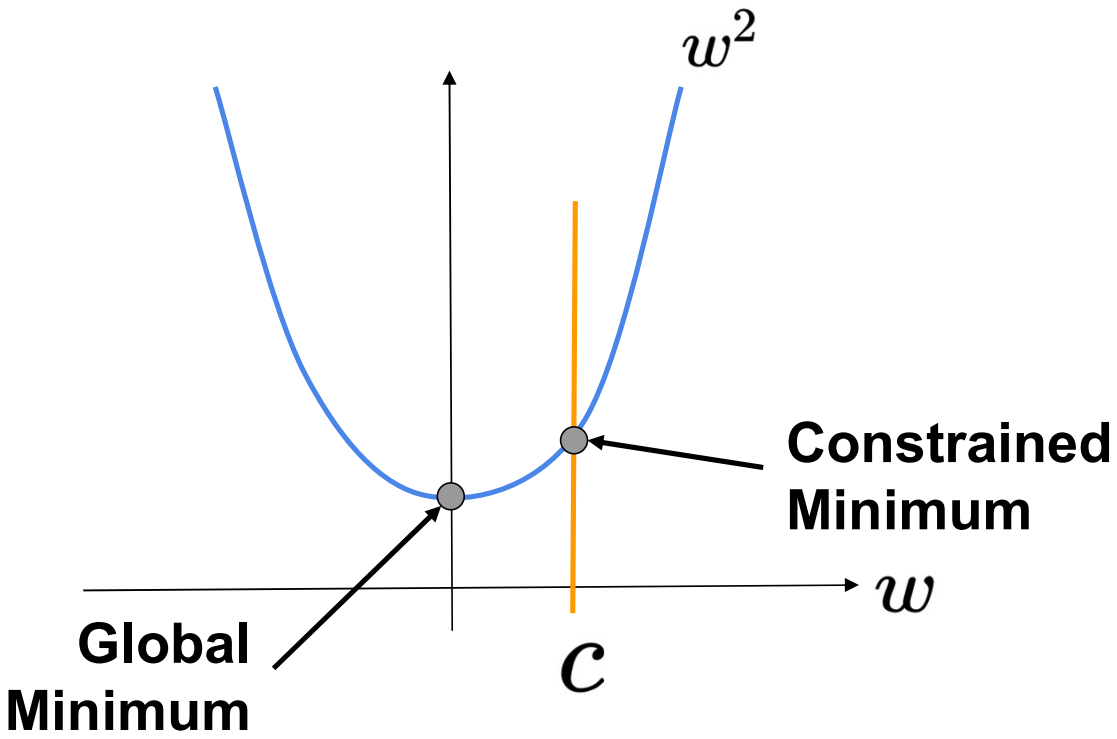
$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$ minimize $||\mathbf{w}||^2$ insteads of $||\mathbf{w}||$

$$\text{s.t.} \quad y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1 \text{ for } i = 1, \ldots, m$$

This can be solved using quadratic programming:

- Quadratic objective
- Linear constraints

Quadratic Optimization



$$\min_{w} w^2$$

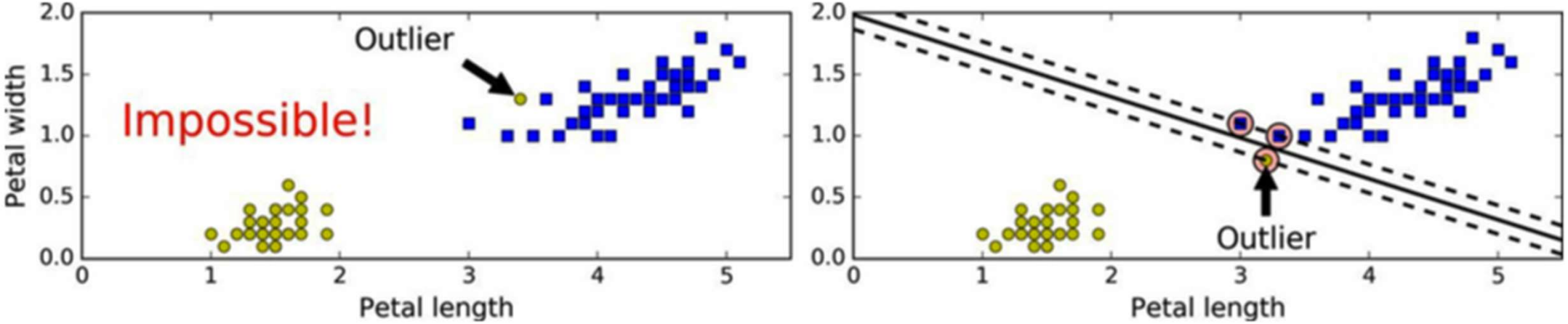
$$\text{s.t.} \quad w \geq c$$

This is a convex quadratic optimization problem which can be solved by **quadratic programming**.
We can use **off-the-self solvers (or good-old GD)** to solve it.

Hard Margin Classification

So far, we've used **hard margin** classification: all training samples are on the "correct side of the street":

- Only work if the data is linearly separable (Left Figure)
- Sensitive to **outliers** → not generalize (Right Figure)

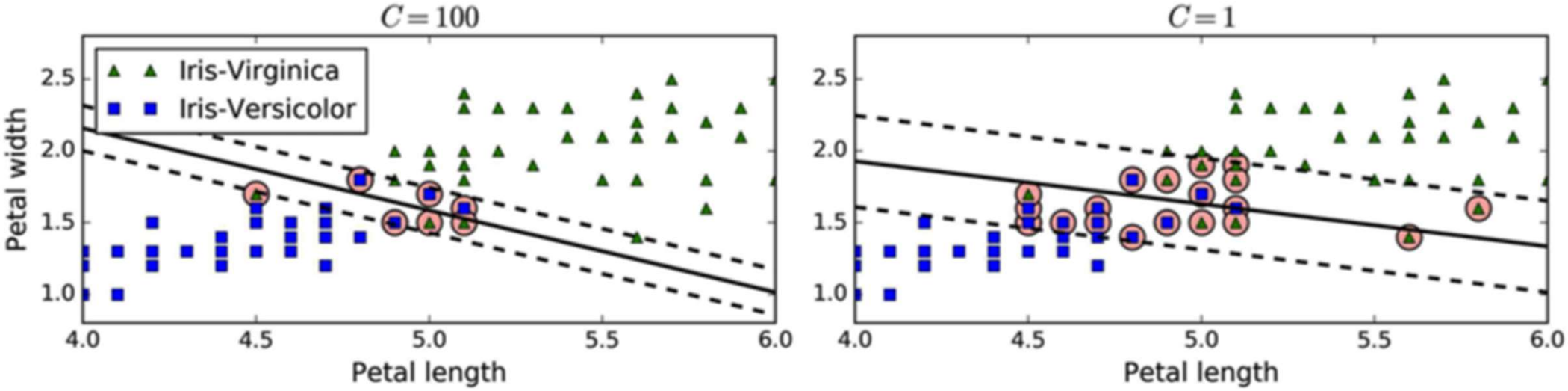


We need to use a more flexible model

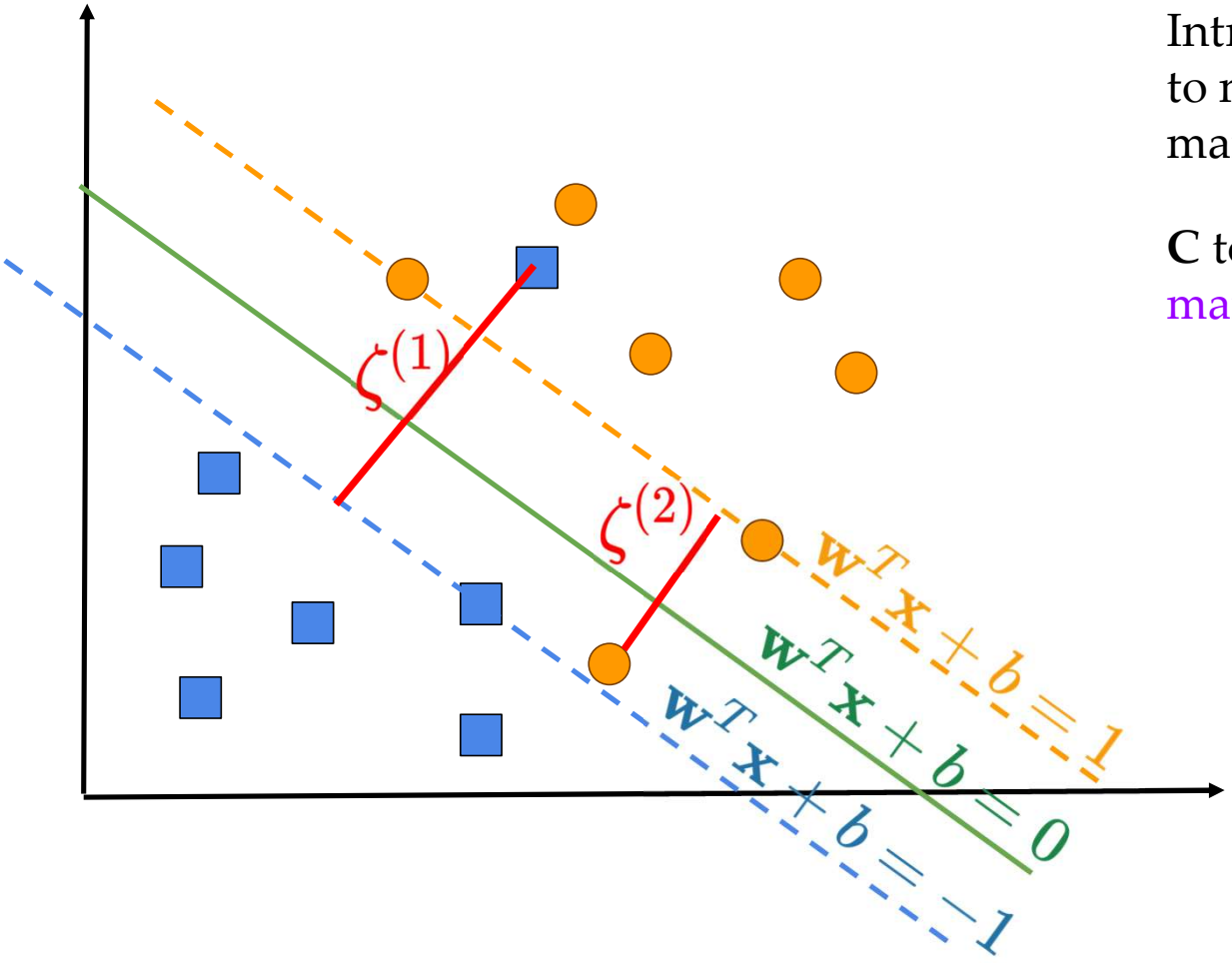
Soft Margin Classification

Objective: find a good balance between keeping the margin as large as possible while limiting the margin **violations**

Controlled by hyperparameter **C**: larger value → smaller margin but less violation



Soft margin linear SVM (soft SVM)



Introduce a slack variable ζ (*zeta*) for **each** sample to measures how much a sample violates the margin.

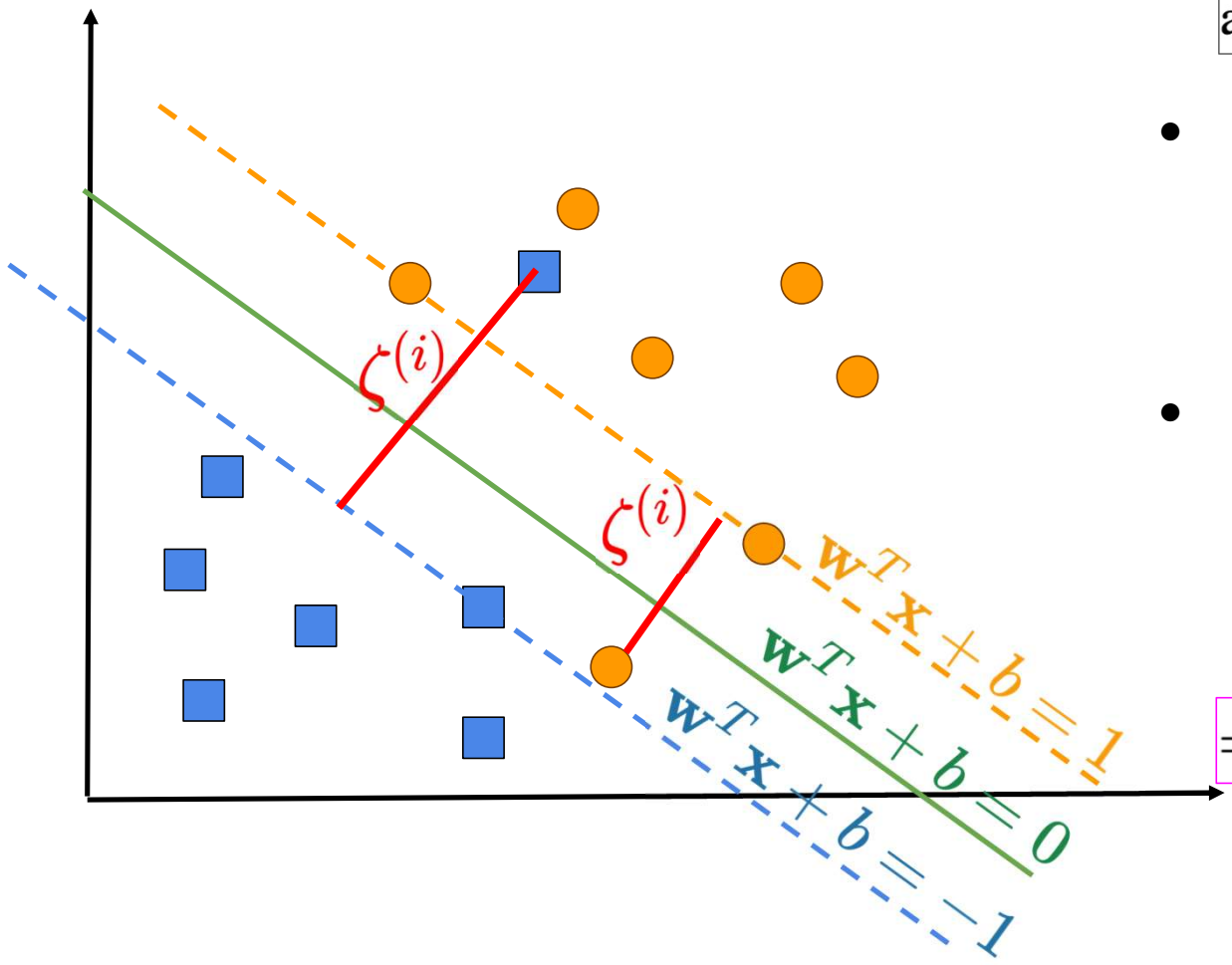
C to control to tradeoff between maximize the margin and minimize the margin violation.

$$\arg\min_{\mathbf{w},b,} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{m}\zeta^{(i)}$$

$$\text{subject to } y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1 - \zeta^{(i)}$$

$$\text{and } \zeta^{(i)} \geq 0 \text{ for } i = 1, \ldots, m$$

Choosing the value of ζ_i



$$\arg\min_{\mathbf{w},b,} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{m} \zeta^{(i)}$$
$$\text{subject to } y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1 - \zeta^{(i)}$$
$$\text{and } \zeta^{(i)} \geq 0 \text{ for } i = 1, \ldots, m$$

- If the example is on the safe side:

$$y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1, \text{ so}$$
$$\zeta^{(i)} = 0$$

- If the example is violating the margin:

$$y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) < 1, \text{ so}$$
$$\zeta^{(i)} = 1 - y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)$$

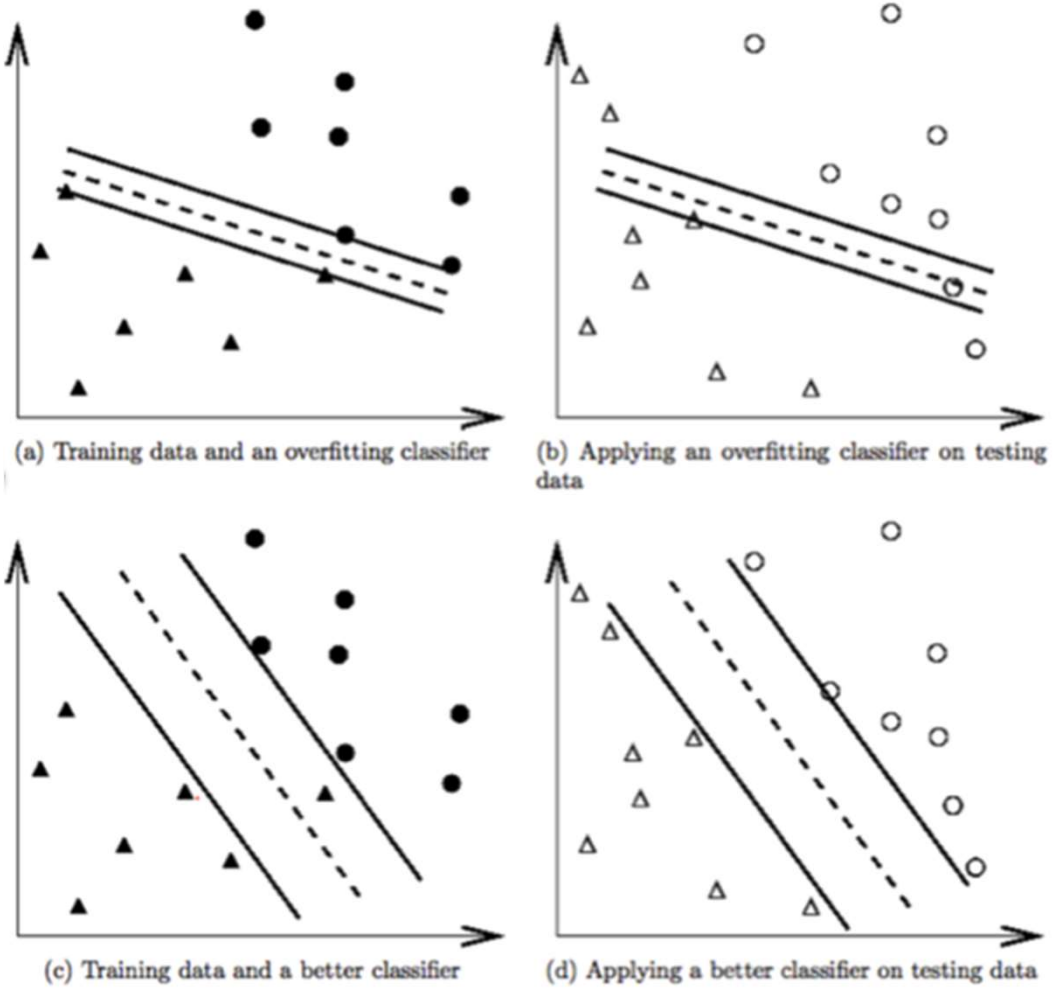
$$\Rightarrow \zeta^{(i)} = \max\left(0, 1 - y^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)\right)$$

Finding the right C

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{m} L_{\text{hinge}}(y_i, \mathbf{w}^\top \mathbf{x}^{(i)} + b)$$

Large C: means that misclassification are bad -- resulting in smaller margin and less training error

Small C: Resulting in larger margin and more training error, but hopefully better testing error



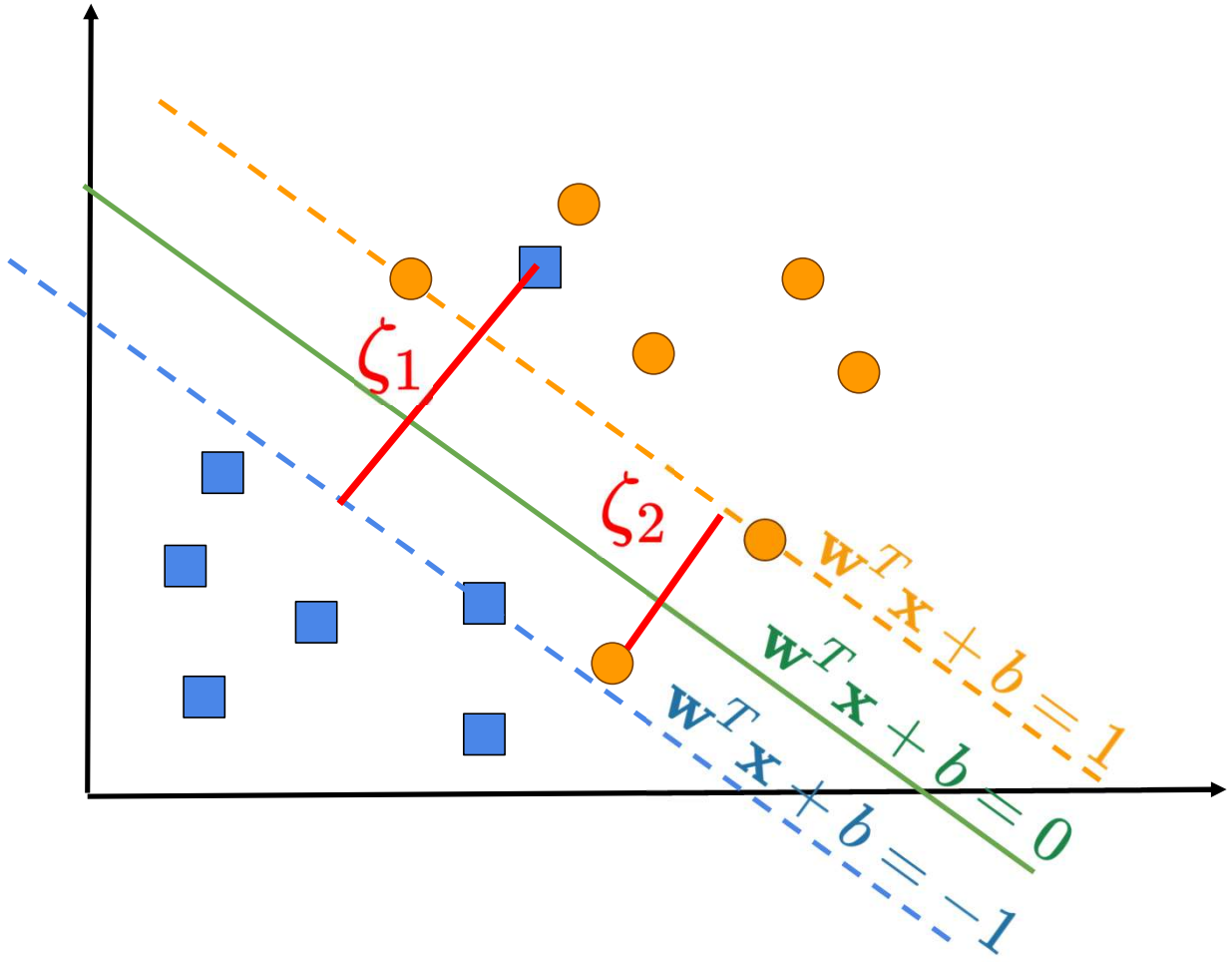
(a) Training data and an overfitting classifier

(b) Applying an overfitting classifier on testing data

(c) Training data and a better classifier

(d) Applying a better classifier on testing data

Rewrite the objective function



$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{m} \zeta^{(i)}$$

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{m} L_{\text{hinge}}(y_i, \mathbf{w}^\top \mathbf{x}^{(i)} + b)$$

Regularization **Hinge Loss on Data**

Again, **C** is used to control the tradeoff between **maximize the margin** and **minimize hinge loss**.

Hinge Loss for data

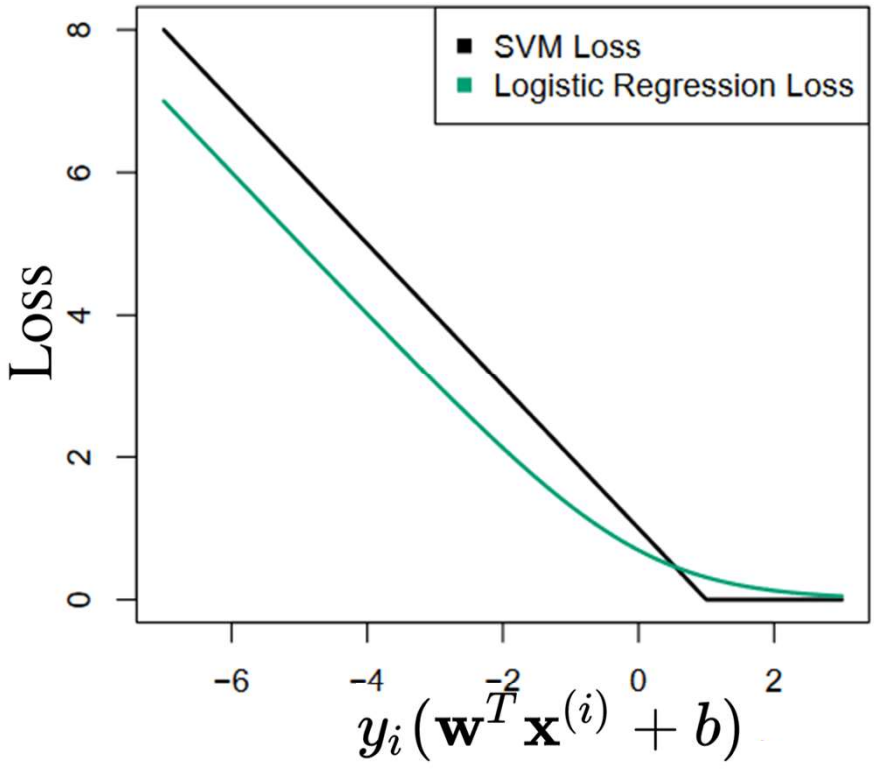
$$\zeta^{(i)} = \max(0, 1 - y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)) = L_{\text{hinge}}(y^{(i)}, \mathbf{w}^\top \mathbf{x}^{(i)} + b)$$

"**hinge loss**" analogy

26

Hinge Loss for data

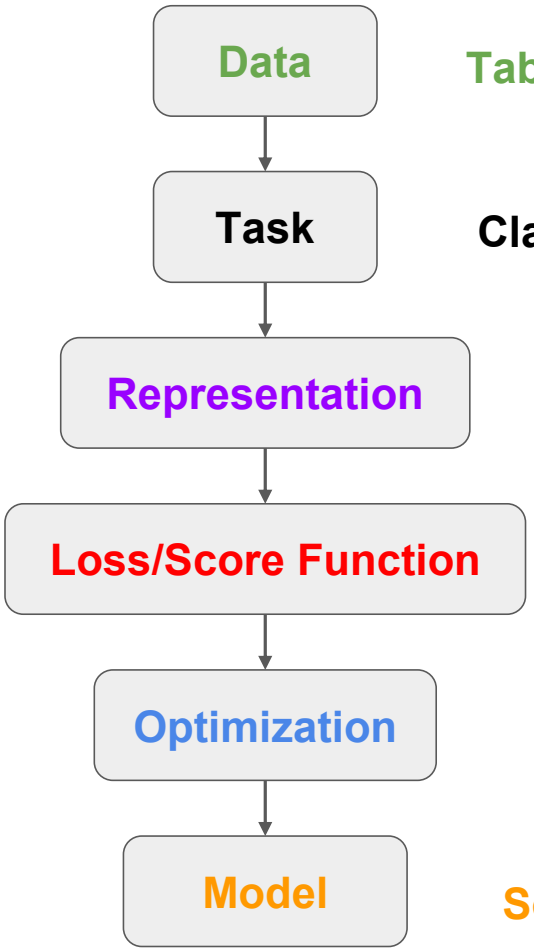
$$\zeta^{(i)} = \max(0, 1 - y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)) = L_{\text{hinge}}(y^{(i)}, \mathbf{w}^\top \mathbf{x}^{(i)} + b)$$



•Logistic regression focuses on maximizing the probability of the data. The farther the data lies from the separating hyperplane (on the correct side), the happier LR is.

•An SVM tries to find the separating hyperplane that maximizes the distance of the closest points to the margin (the support vectors). If a point is not a support vector, it doesn't really matter

Recap: Support Vector Machines



Tabular (table) form

Classification / Regression / Outlier Detection

Maximized Margin

Hinge-Loss

Quadratic Programming

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{i=1}^{m} L_{\text{hinge}}(y_i, \mathbf{w}^\top \mathbf{x}^{(i)} + b)$$

Soft Margin Model

Summary: Learning Objectives

- ✓ Know **maximized margin** classification
- ✓ Derive **objective function** for Linear SVM
- ✓ Handle **soft-margin** classification with Hinge Loss

Next: Kernel Methods

Computing the decision boundary

- The primal formulation of the optimization problem is:

  Minimize $\frac{1}{2}||w||^2$

  subject to $1 - y_i(w^T x_i + b) \leq 0, \quad \forall i$

Computing the decision boundary

- The primal formulation of the optimization problem is:

  Minimize $\frac{1}{2}||w||^2$

  subject to $1 - y_i(w^T x_i + b) \leq 0, \quad \forall i$

How many constraints?

Computing the decision boundary

- The primal formulation of the optimization problem is:

  Minimize $\frac{1}{2}||w||^2$

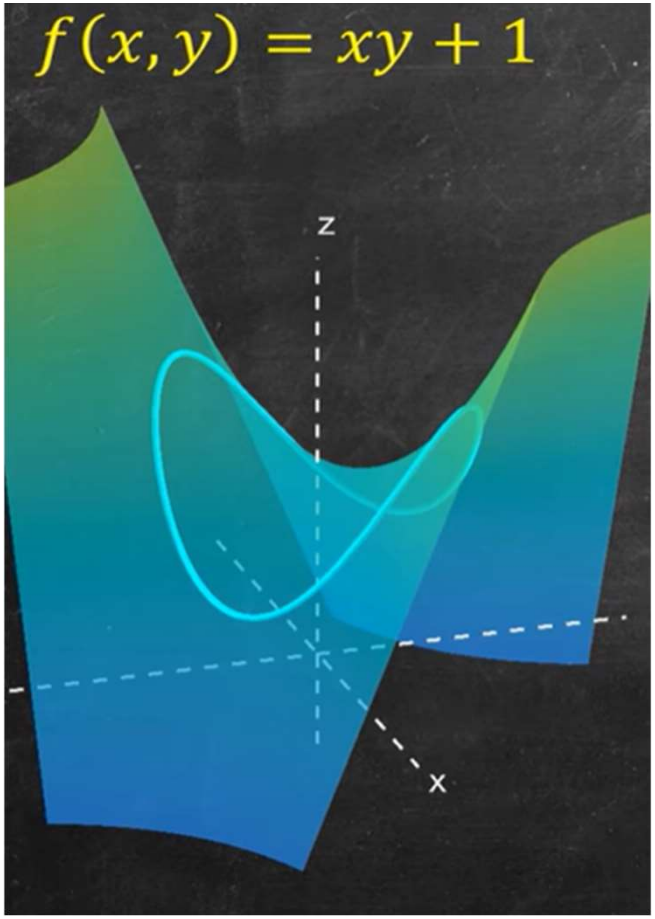
  subject to $1 - y_i(w^T x_i + b) \leq 0, \quad \forall i$

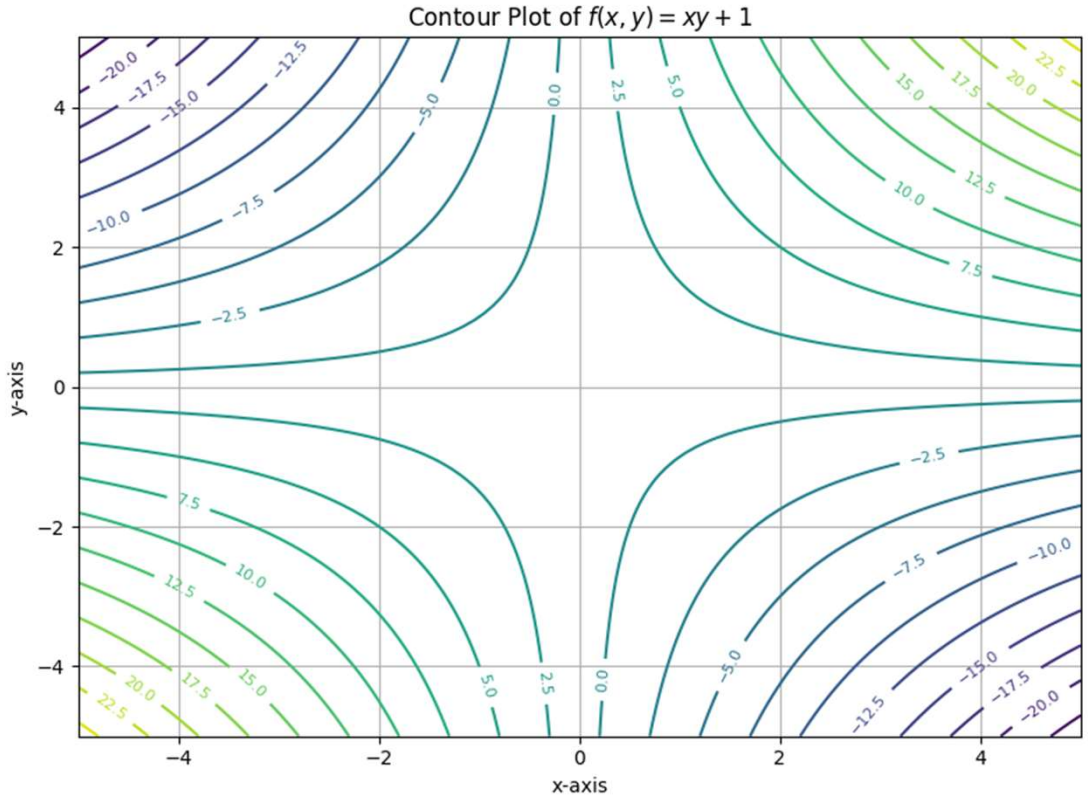
- The Lagrangian is[1]:

$$\mathcal{L} = \frac{1}{2} w^T w + \sum_{i=1}^{n} \alpha_i (1 - y_i (w^T x_i + b))$$

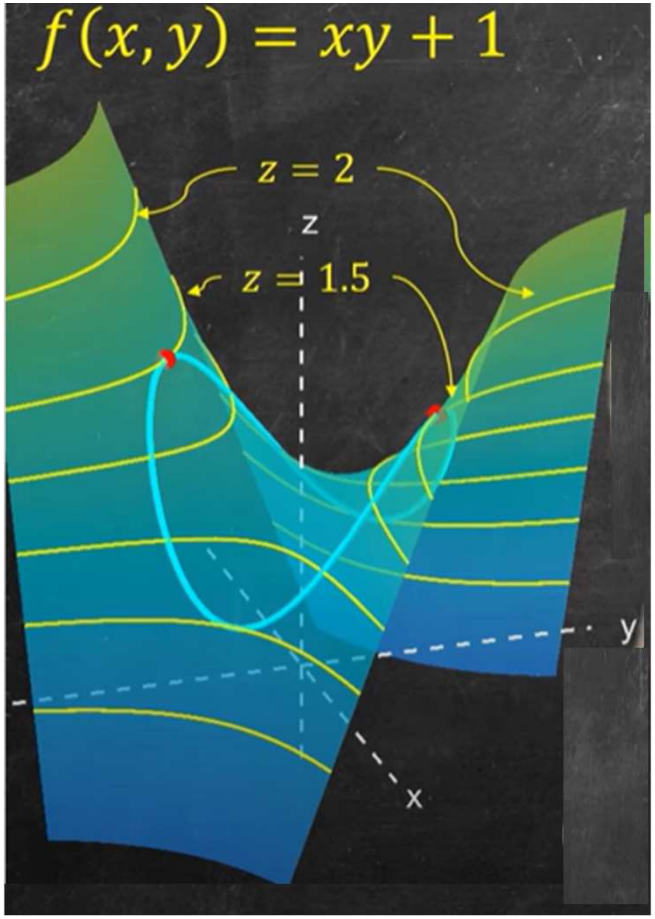
Sharpen Your Axe

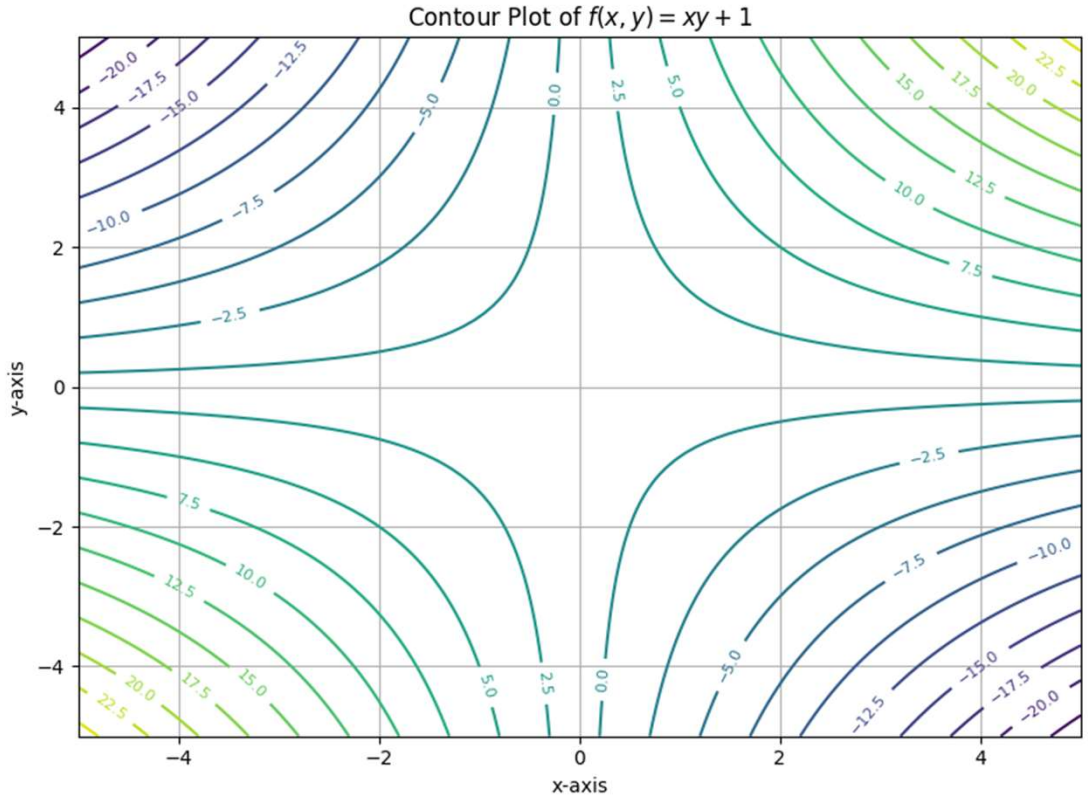
Contour lines



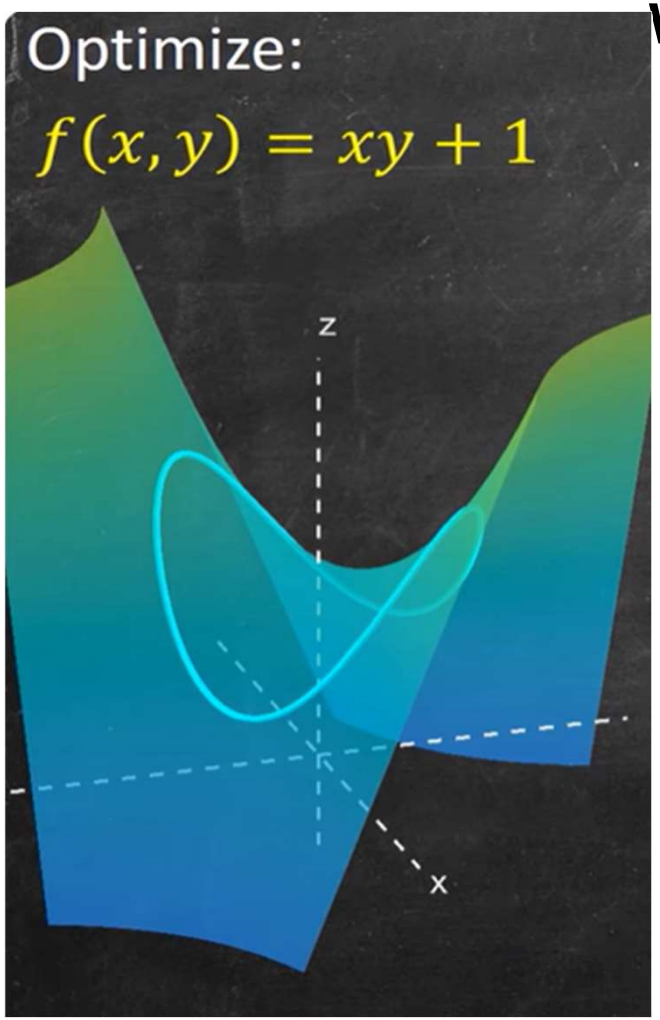


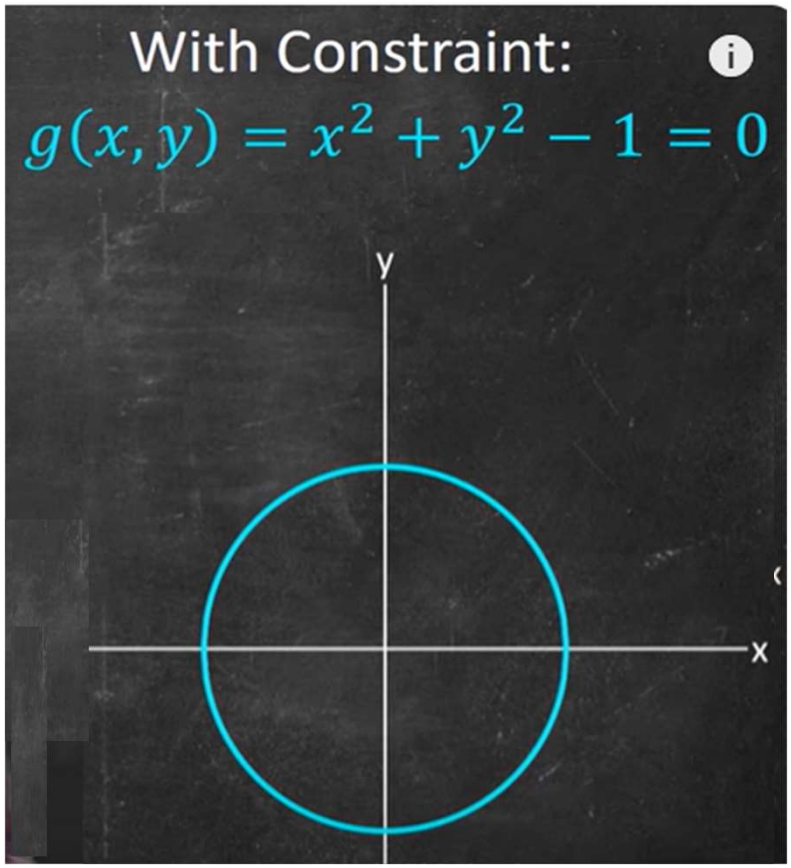
Contour lines





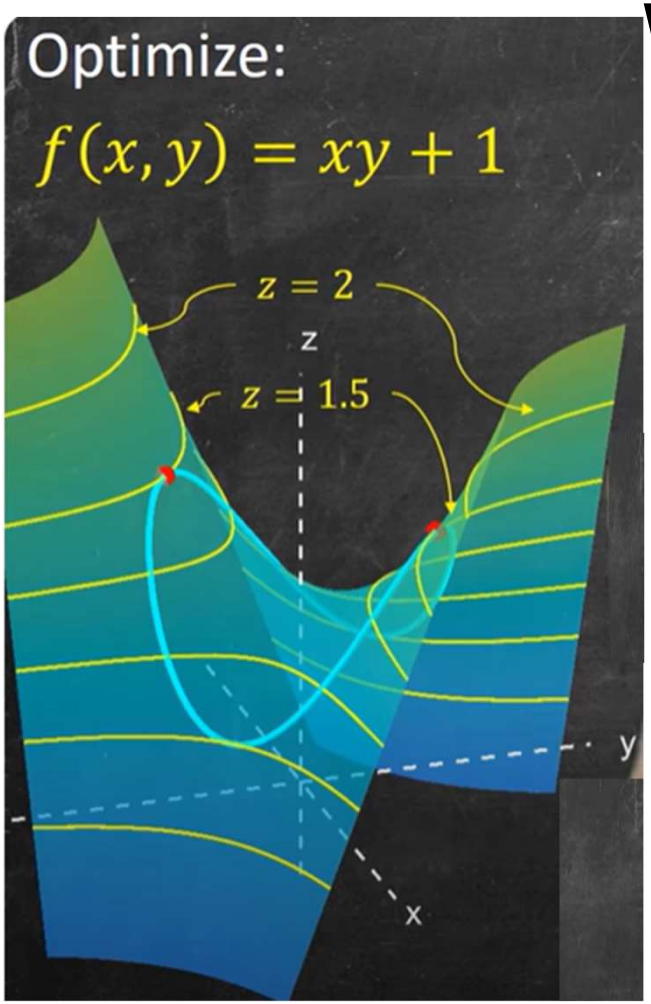
What is Lagrange Multiplier?

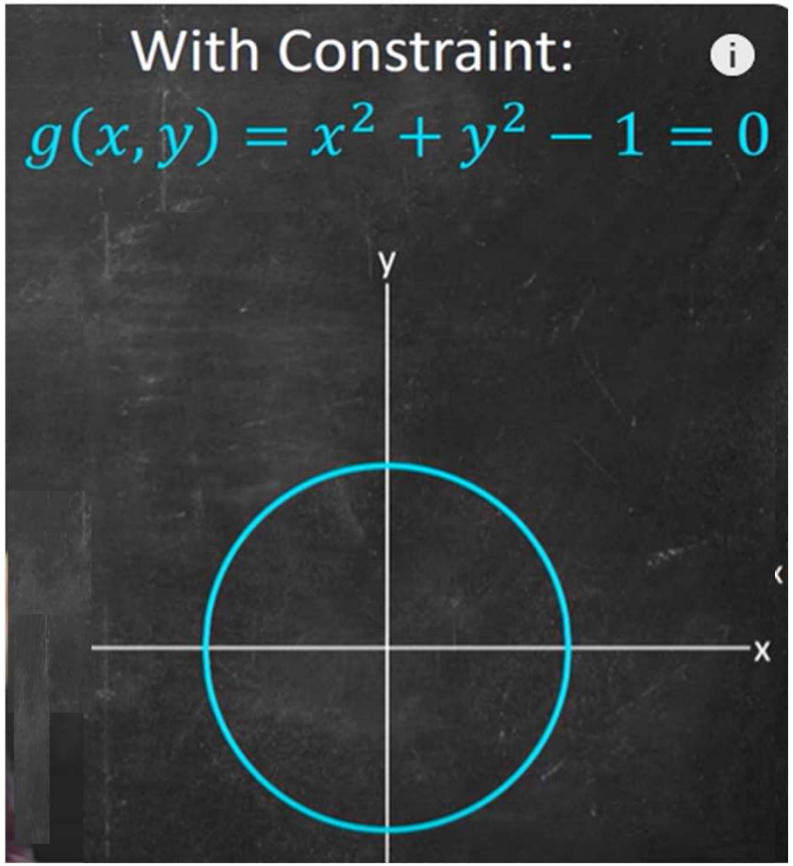




https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

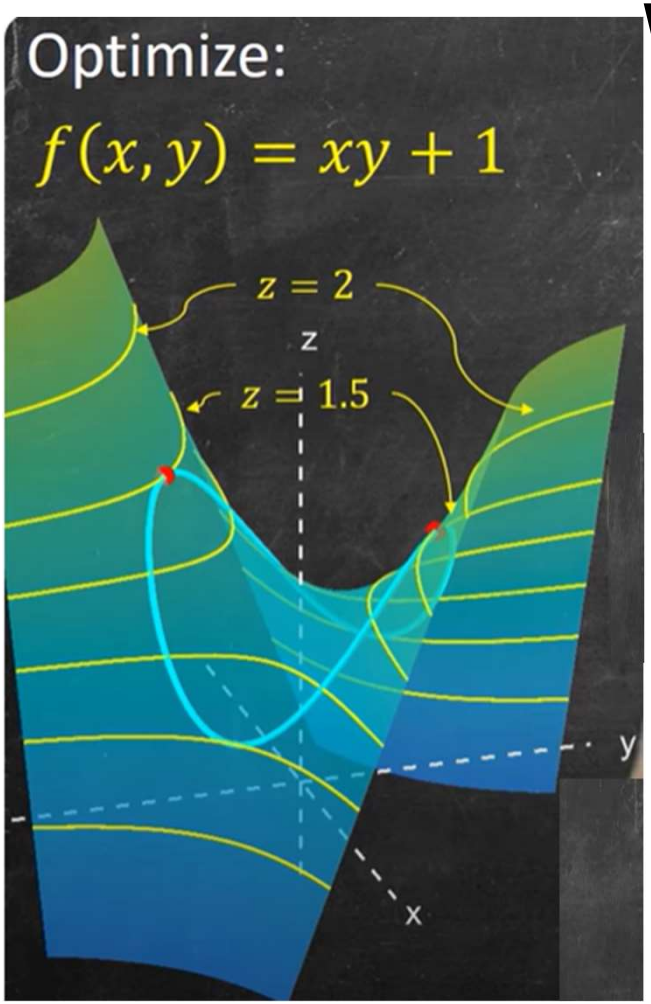
What is Lagrange Multiplier?

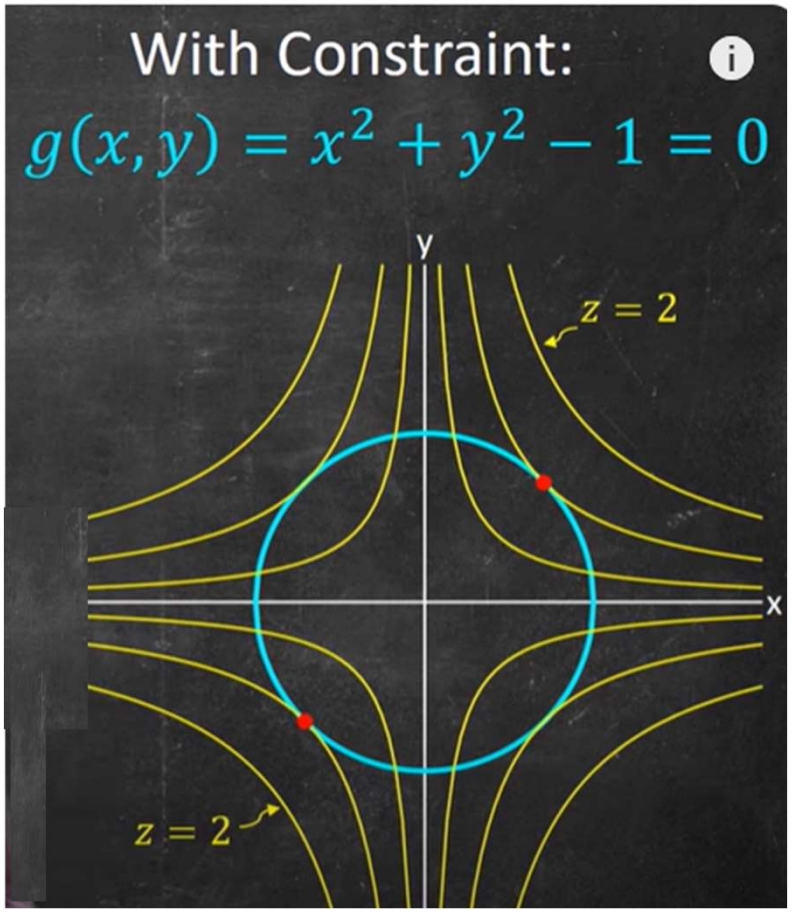




https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

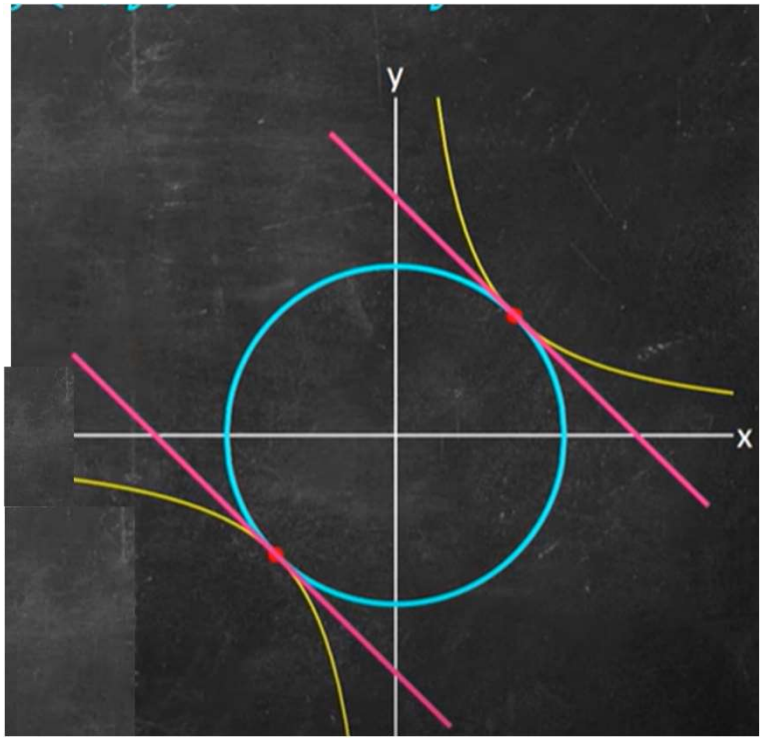
What is Lagrange Multiplier?





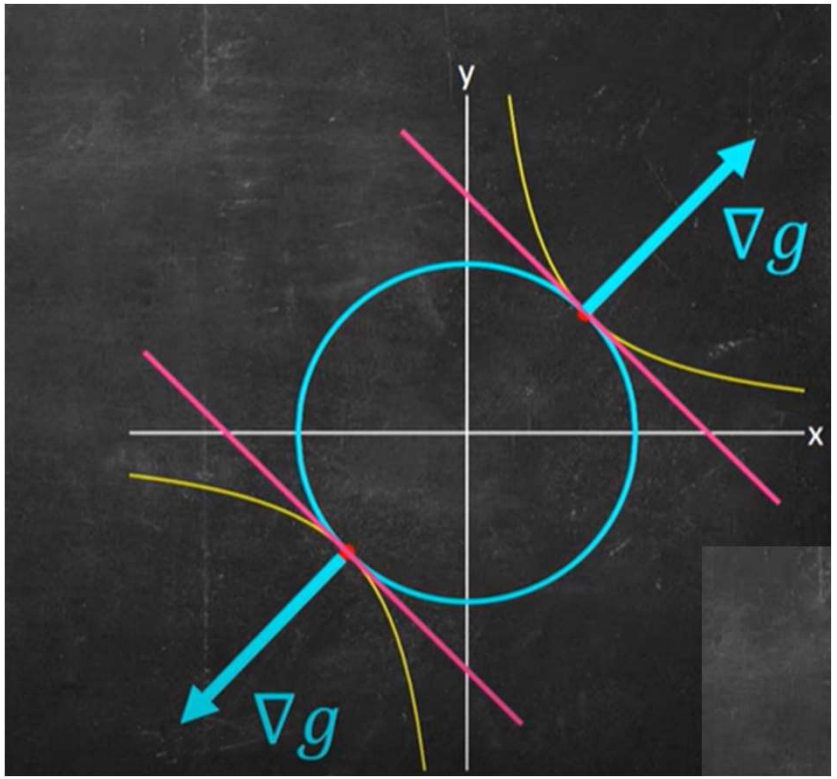
https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

What is Lagrange Multiplier?



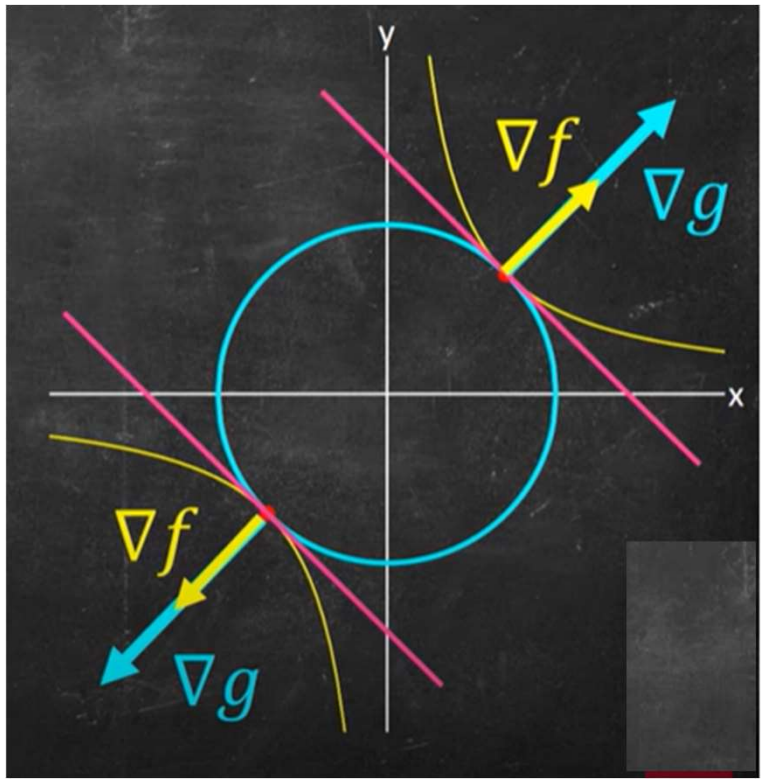
https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

What is Lagrange Multiplier?



https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

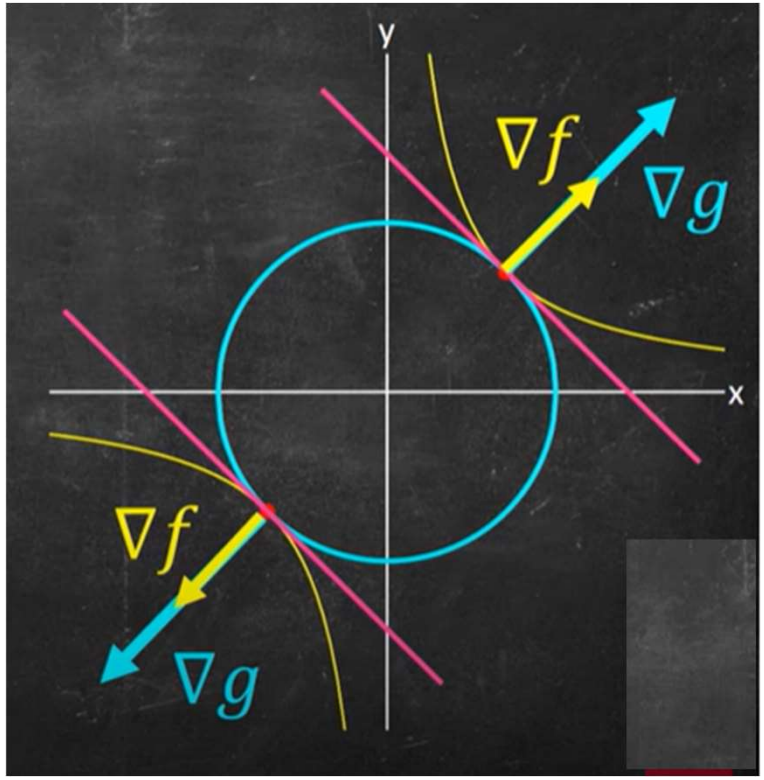
What is Lagrange Multiplier?



https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

What is Lagrange Multiplier?

Lagrange Multipliers:
Simultaneously solve
$\nabla f = \lambda \nabla g$
$g = 0$



In English: we are looking for points where the gradient of the objective function and the gradient of constraints are scalar multiple of each other

https://www.youtube.com/watch?v=8mjcnxGMwFo&ab_channel=Dr.TreforBazett

Back to lecture

Caution: That was only a hint to understand the intuition behind Lagrange multipliers and dual formulation
Many details are hided for simplicity

Computing the decision boundary

- The primal formulation of the optimization problem is:

  Minimize $\frac{1}{2}||w||^2$

  subject to $1 - y_i(w^T x_i + b) \leq 0, \quad \forall i$

- The Lagrangian is[1]:

$$\mathcal{L} = \frac{1}{2} w^T w + \sum_{i=1}^{n} \alpha_i (1 - y_i(w^T x_i + b))$$

Computing the decision boundary

- The primal formulation of the optimization problem is:

  Minimize $\frac{1}{2}||w||^2$

  subject to $1 - y_i(w^T x_i + b) \leq 0, \quad \forall i$

- The Lagrangian is[1]:

$$\mathcal{L} = \frac{1}{2} w^T w + \sum_{i=1}^{n} \alpha_i (1 - y_i(w^T x_i + b))$$

Lagrange multipliers

objective

constraints

Computing the decision boundary

- If we compute the derivative of $\mathcal{L}$ with respect to w and b and we set them to zero (we are computing the gradient):

$$w + \sum_{i=1}^{n} \alpha_i(-y_i)x_i = 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i$$ With respect to w

and

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$ With respect to b

The Dual Formulation

- If we substitute w in the Lagrangian $\mathcal{L}$

You do not have to memorize this proof

$$
\begin{aligned}
\mathcal{L} &= \frac{1}{2}\sum_{i=1}^{n}\alpha_i y_i x_i^T \sum_{j=1}^{n}\alpha_j y_j x_j + \sum_{i=1}^{n}\alpha_i \left(1 - y_i(\sum_{j=1}^{n}\alpha_j y_j x_j^T x_i + b)\right) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i y_i \sum_{j=1}^{n}\alpha_j y_j x_j^T x_i \\
&\quad -b\sum_{i=1}^{n}\alpha_i y_i \quad (\textit{and given that } \sum_{i=1}^{n}\alpha_i y_i = 0) \\
&= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{n}\alpha_i \quad (\textit{rearranging terms})
\end{aligned}
$$

The Dual Formulation

- If we substitute w in the Lagrangian $\mathcal{L}$

You do not have to memorize this proof

And I do not have to explain it

$$\begin{aligned}
\mathcal{L} &= \frac{1}{2}\sum_{i=1}^{n}\alpha_i y_i x_i^T \sum_{j=1}^{n}\alpha_j y_j x_j + \sum_{i=1}^{n}\alpha_i \left(1 - y_i(\sum_{j=1}^{n}\alpha_j y_j x_j^T x_i + b)\right) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i y_i \sum_{j=1}^{n}\alpha_j y_j x_j^T x_i \\
&\quad -b\sum_{i=1}^{n}\alpha_i y_i \quad (\textit{and given that} \quad \sum_{i=1}^{n}\alpha_i y_i = 0) \\
&= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{n}\alpha_i \quad (\textit{rearranging terms})
\end{aligned}$$

The Dual Formulation

- This problem is known as the *dual problem* (the original is the primal)
- This formulation only depends on the α_i
- Both problems are linked, given the w we can compute the α_i and viceversa
- This means that we can solve one instead of the other
- Now this problem is a maximization problem ($\alpha_i \geq 0$)

The Dual Problem

- The formulation of the dual problem is

  Maximize $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$

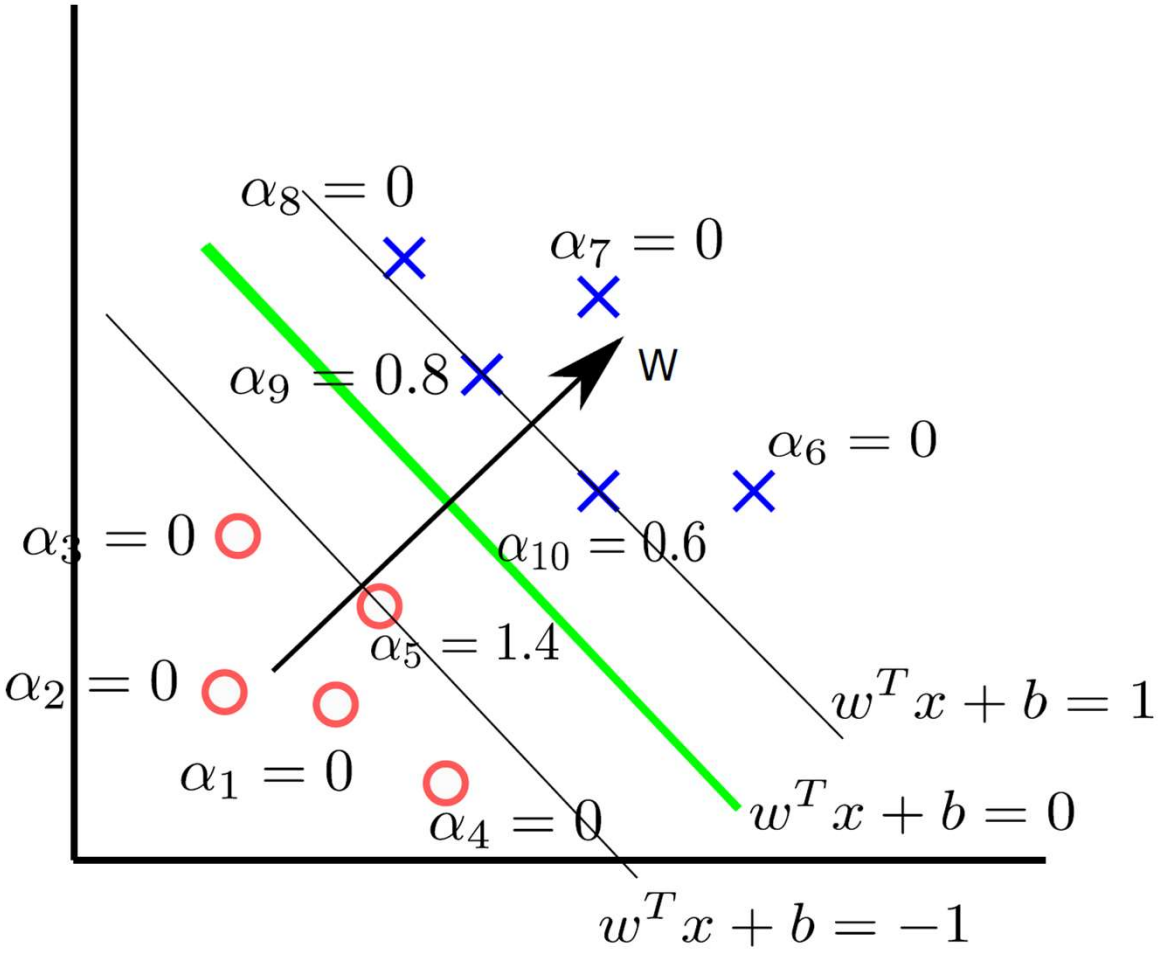
  subject to $\sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i \geq 0 \ \ \forall i$

- This a Quadratic Programming problem (QP)
- This means that the parameters form a parabolloidal surface, and an optimal can be found
- We can obtain w by

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

- b can be obtained from a positive support vector (x_{psv}) knowing that $w^T x_{psv} + b = 1$

Geometric Interpretation



Characteristics of the solution

- Many of the α_i are zero
  - w is a linear combination of a small number of examples
  - We obtain a *sparse* representation of the data (data compression)
- The examples x_i with non zero α_i are the support vectors (SV)
- The vector of parameters can be expressed as:

$$w = \sum_{\forall i \in SV} \alpha_i y_i x_i$$

Classifying new instances

- In order to classify a new instance z we just have to compute

$$sign(w^T z + b) = sign(\sum_{\forall i \in SV} \alpha_i y_i x_i^T z + b)$$

- This means that w does not need to be explicitly computed

Numerical Example on classifying new instance

Given Decision Function

$$\text{sign}(w^T z + b) = \text{sign}\left(\sum_{i \in \text{SV}} \alpha_i y_i x_i^T z + b\right)$$

Where:

- α_i: Lagrange multiplier for support vector x_i.
- y_i: Class label ($+1$ or -1).
- x_i: Support vector in the training set.
- z: New instance to classify.
- b: Bias term.

Numerical Example on classifying new instance

SVM Parameters

1. Support Vectors ($\mathbf{x}_i$):
   - $\mathbf{x}_1 = [6.5, 1]$, $y_1 = +1$, $\alpha_1 = 1.33$
   - $\mathbf{x}_2 = [3, 0.5]$, $y_2 = -1$, $\alpha_2 = 5.00$
   - $\mathbf{x}_3 = [2, -0.5]$, $y_3 = +1$, $\alpha_3 = 3.67$
2. Lagrange Multipliers (α_i):
   - $\alpha_1 = 1.33$
   - $\alpha_2 = 5.00$
   - $\alpha_3 = 3.67$
3. Class Labels (y_i):
   - $y_1 = +1$
   - $y_2 = -1$
   - $y_3 = +1$
4. Bias (b):
   - $b = -2.5$

Numerical Example on classifying new instance

Let the new instance z have two features:

$$\mathbf{z} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

$$\mathbf{sign}(w^T z + b) = \mathbf{sign}\left(\sum_{i \in \mathrm{SV}} \alpha_i y_i x_i^T z + b\right)$$

Numerical Example on classifying new instance

Let the new instance z have two features:

$$z = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$\text{sign}(w^T z + b) = \text{sign}\left(\sum_{i \in \text{SV}} \alpha_i y_i x_i^T z + b\right)$$

Step 1: Compute the Dot Products $\mathbf{x}_i^\top \mathbf{z}$ for Each Support Vector:

1. For $\mathbf{x}_1 = [6.5, 1]$:

$$\mathbf{x}_1^\top \mathbf{z} = 6.5 \cdot 4 + 1 \cdot 2 = 26 + 2 = 28$$

2. For $\mathbf{x}_2 = [3, 0.5]$:

$$\mathbf{x}_2^\top \mathbf{z} = 3 \cdot 4 + 0.5 \cdot 2 = 12 + 1 = 13$$

3. For $\mathbf{x}_3 = [2, -0.5]$:

$$\mathbf{x}_3^\top \mathbf{z} = 2 \cdot 4 + (-0.5) \cdot 2 = 8 - 1 = 7$$

Step 2: Substitute the Values:

1. Contribution from $\mathbf{x}_1$:

$$\alpha_1 y_1 \mathbf{x}_1^\top \mathbf{z} = 1.33 \cdot (+1) \cdot 28 = 37.24$$

2. Contribution from $\mathbf{x}_2$:

$$\alpha_2 y_2 \mathbf{x}_2^\top \mathbf{z} = 5.00 \cdot (-1) \cdot 13 = -65.00$$

3. Contribution from $\mathbf{x}_3$:

$$\alpha_3 y_3 \mathbf{x}_3^\top \mathbf{z} = 3.67 \cdot (+1) \cdot 7 = 25.69$$

Step 3: Add Them Together with the Bias:

$$\text{Decision Value} = 37.24 + (-65.00) + 25.69 + (-2.5) = -4.57$$

Step 4: Apply the Sign Function:

$$\text{sign}(-4.57) = -1$$

Numerical Example on classifying new instance

Can you find the decision boundary equation?

Let the new instance z have two features:

$$z = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$\text{sign}(w^T z + b) = \text{sign}\left(\sum_{i \in \text{SV}} \alpha_i y_i x_i^T z + b\right)$$

Step 1: Compute the Dot Products $\mathbf{x}_i^\top \mathbf{z}$ for Each Support Vector:

1. For $\mathbf{x}_1 = [6.5, 1]$:

$$\mathbf{x}_1^\top \mathbf{z} = 6.5 \cdot 4 + 1 \cdot 2 = 26 + 2 = 28$$

2. For $\mathbf{x}_2 = [3, 0.5]$:

$$\mathbf{x}_2^\top \mathbf{z} = 3 \cdot 4 + 0.5 \cdot 2 = 12 + 1 = 13$$

3. For $\mathbf{x}_3 = [2, -0.5]$:

$$\mathbf{x}_3^\top \mathbf{z} = 2 \cdot 4 + (-0.5) \cdot 2 = 8 - 1 = 7$$

Step 2: Substitute the Values:

1. Contribution from $\mathbf{x}_1$:

$$\alpha_1 y_1 \mathbf{x}_1^\top \mathbf{z} = 1.33 \cdot (+1) \cdot 28 = 37.24$$

2. Contribution from $\mathbf{x}_2$:

$$\alpha_2 y_2 \mathbf{x}_2^\top \mathbf{z} = 5.00 \cdot (-1) \cdot 13 = -65.00$$

3. Contribution from $\mathbf{x}_3$:

$$\alpha_3 y_3 \mathbf{x}_3^\top \mathbf{z} = 3.67 \cdot (+1) \cdot 7 = 25.69$$

Step 3: Add Them Together with the Bias:

$$\text{Decision Value} = 37.24 + (-65.00) + 25.69 + (-2.5) = -4.57$$

Step 4: Apply the Sign Function:

$$\text{sign}(-4.57) = -1$$

Take home message

- Dual formulation turns the problem to a problem in α_i (Lagrange multipliers)
- We do not need to compute the weights of the decision boundary
- For all non-support vectors points $\alpha_i = 0$
- We end up having α_i values corresponding to the support vector
- The weights of the model can be calculated from the support vectors α_is but we do not have to
- So, we describe the model using its support vectors

Take home message

- Dual formulation turns the problem to a problem in α_i (Lagrange mulitpliers)
- We do not need to compute the weights of the decision boundry
- For all non-support vectors points $\alpha_i = 0$
- We end up having α_i values corresponding to the support vector
- The weights of the model can be calculated from the support vectors α_is but we do not have to
- **So, we describe the model using its support vectors**

The dual optimization problem (soft margin)

- Performing the transformation to the dual problem we obtain the following:

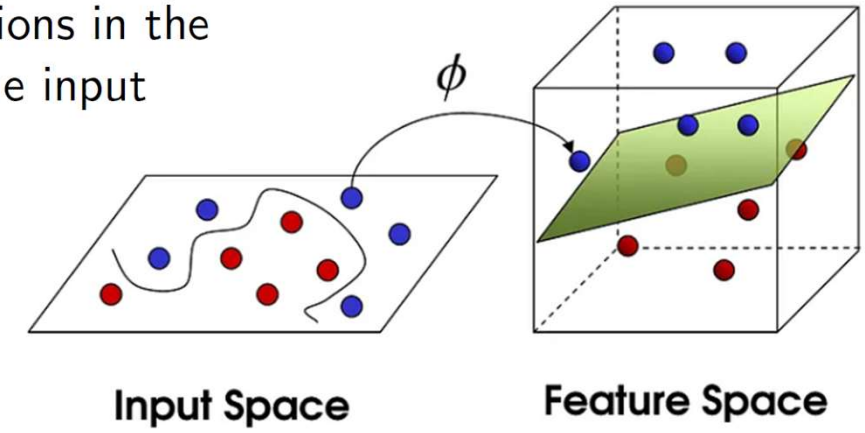
  Maximize $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$

  subject to $\sum_{i=1}^{n} \alpha_i y_i = 0, \quad C \geq \alpha_i \geq 0 \ \ \forall i$

- We can recover the solution as $w = \sum_{\forall i \in SV} \alpha_i, y_i x_i$
- This problem is very similar to the linearly separable case, except that there is a upper bound C on the values of α_i
- This is also a QP problem

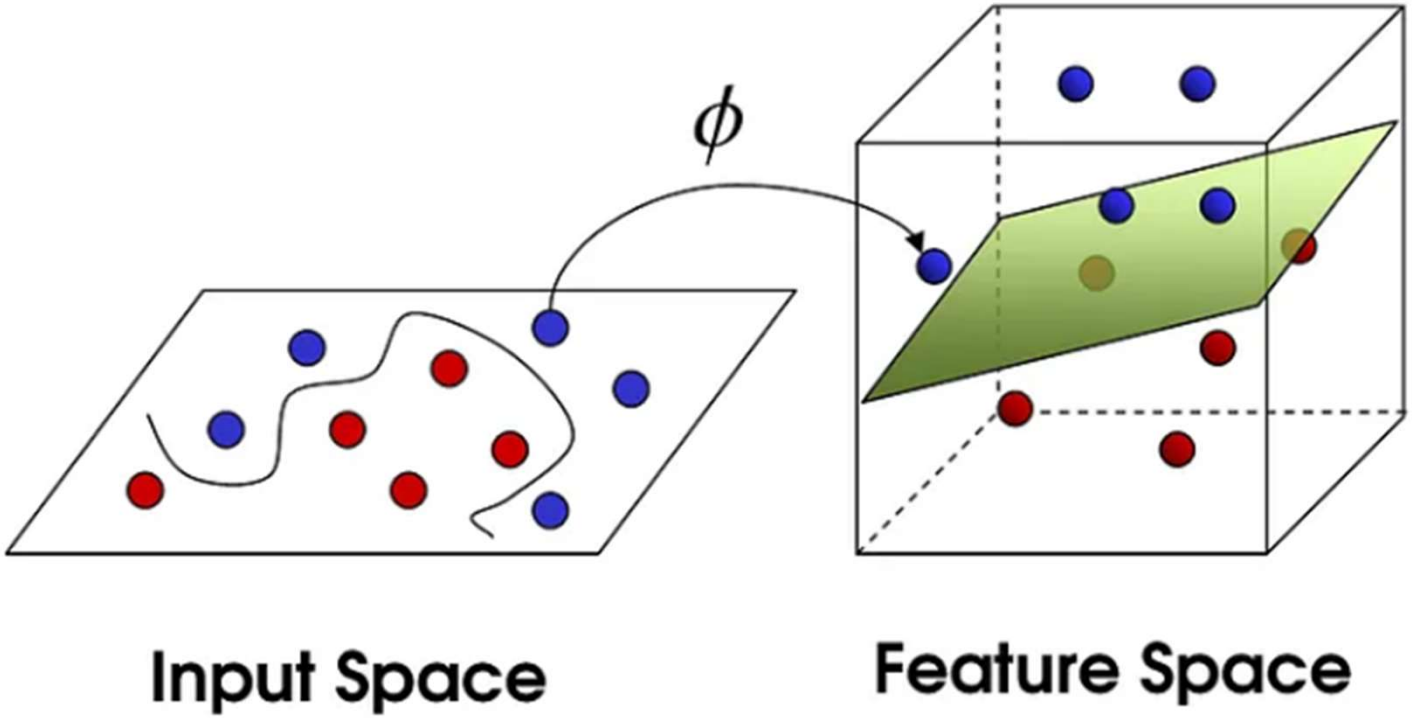
Nonlinear large margin classification

- So far we have only considered large margin classifiers that use a linear boundary
- In order to have better performance we have to be able to obtain non-linear boundaries
- The idea is to transform the data from the input space (the original attributes of the examples) to a higher dimensional space using a function $\phi(x)$
- This new space is called the **feature space**
- The advantage of the transformation is that linear operations in the feature space are equivalent to non-linear operations in the input space



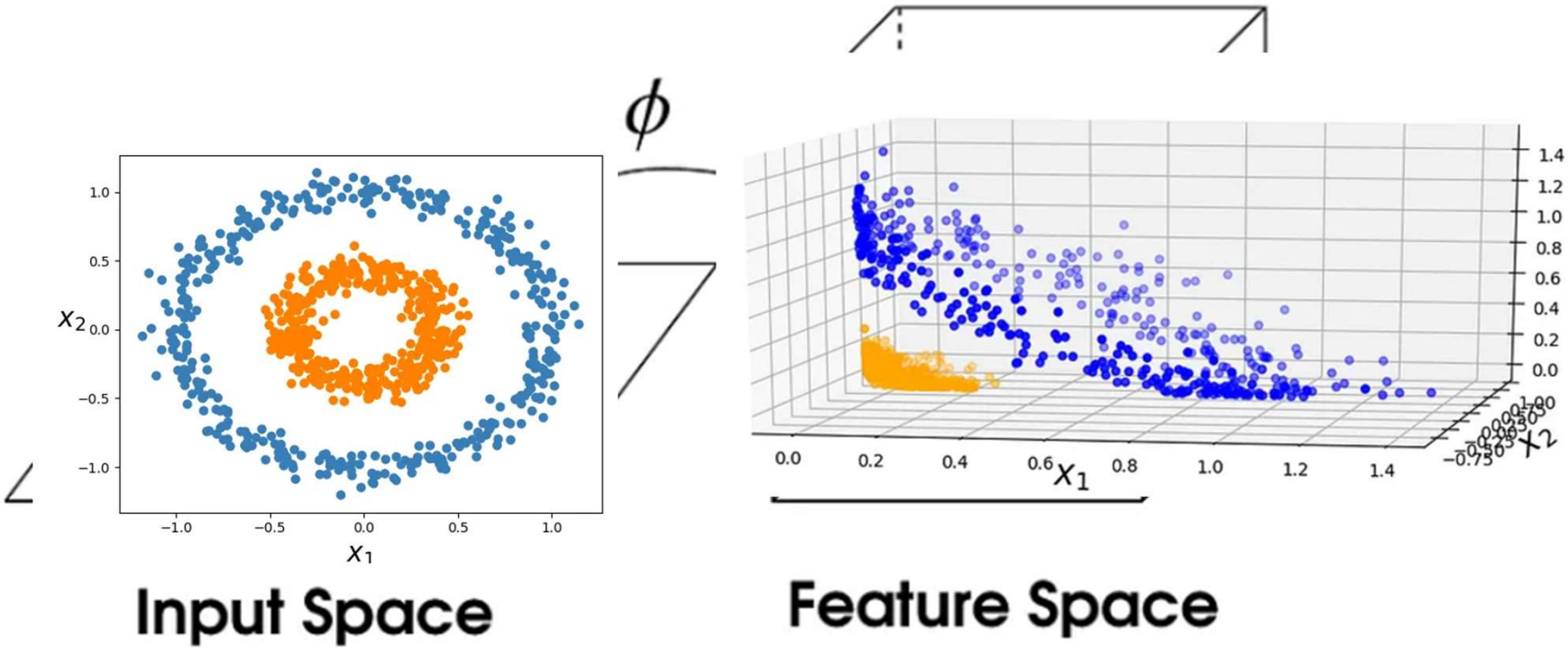
Nonlinear large margin classification

- So far we have only considered large margin classifiers that use a linea
- In or non-
- The attri func
- This
- The featı spac

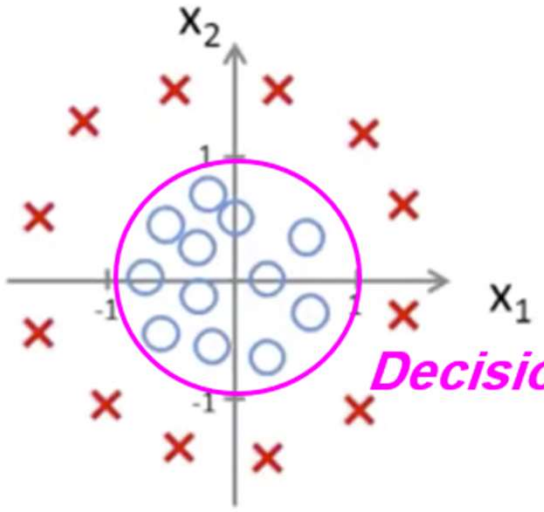


Nonlinear large margin classification

- So fa
  linea
- In or
  non-
- The
  attri
  func
- This
- The
  featı
  spac



Logistic Regression: Explicit Mapping

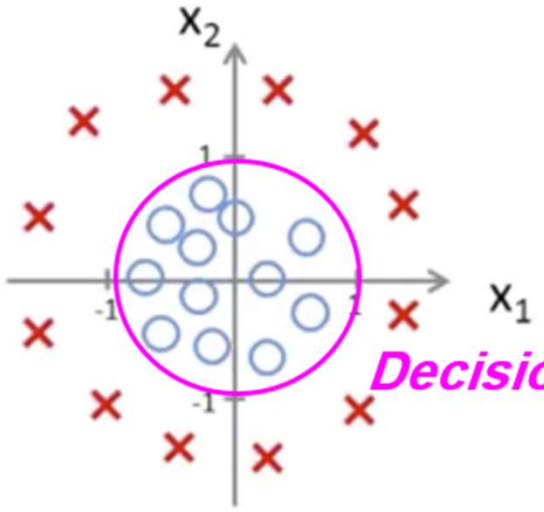


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$\theta = [-1, 0, 0, 1, 1] \qquad Z = \theta^T X = -1 + x_1^2 + x_2^2$$

Compute a linear decision boundary in the new feature space
This is equivalent to non-linear boundary in the input space

Logistic Regression: Explicit Mapping



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$\theta = [-1, 0, 0, 1, 1] \qquad Z = \theta^T X = -1 + x_1^2 + x_2^2$$

We transform dataset samples to new feature space
We do model parameters selection in new feature space

SVM Kernel Trick: Implicit Mapping

- In the problem that dene a SVM only the inner product of the examples is needed

$$\text{Maximize } \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \underbrace{x_i^T x_j}_{\phi(x_i)^T \phi(x_j)}$$

$$\text{subject to } \sum_{i=1}^{n} \alpha_i y_i = 0, \quad C \geq \alpha_i \geq 0 \ \ \forall i$$

- This means that if we can define how to compute this product in the feature space, then it is not necessary to explicitly build it

SVM Kernel Trick: Implicit Mapping

- In the problem that dene a SVM only the inner product of the examples is needed

$$\text{Maximize } \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \underbrace{x_i^T x_j}_{\phi(x_i)^T \phi(x_j)}$$

$$\text{subject to } \sum_{i=1}^{n} \alpha_i y_i = 0, \quad C \geq \alpha_i \geq 0 \ \ \forall i$$

- This means that if we can define how to compute this product in the feature space, then it is not necessary to explicitly build it
- We can dene the kernel function as

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

The Kernel Trick- Example

- We can show how this kernel trick works in an example
- Lets assume a feature space defined as:

$$\phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2 x_2^2, \sqrt{2}x_1 x_2)$$

- A inner product in this feature space is:

$$\left\langle \phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right), \phi\left(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \right\rangle = (1 + x_1 y_1 + x_2 y_2)^2$$

- So, we can define a kernel function to compute inner products in this space as:

$$K(x, y) = (1 + x_1 y_1 + x_2 y_2)^2$$

and we are using only the features from the input space

- Examples of functions that hold this condition are:
  - Polynomial kernel of degree d: $K(x, y) = (x^T y + 1)^d$
  - Gaussian function with width σ: $K(x, y) = \exp(-||x - y||^2/2\sigma^2)$
  - Sigmoid hiperbolical tangent with parameters k and θ: $K(x, y) = \tanh(kx^T y + \theta)$ (only for certain values of the parameters)

Kernel Definition

- A function that takes as its inputs vectors in the original space and returns the dot product of the vectors in the feature space is called a *kernel function*

The Kernel Trick- Example

- We can show how this kernel trick works in an example
- Lets assume a feature space defined as:

$$\phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2 x_2^2, \sqrt{2}x_1 x_2)$$

- A inner product in this feature space is:

$$\left\langle \phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right), \phi\left(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\right) \right\rangle = (1 + x_1 y_1 + x_2 y_2)^2$$

- So, we can define a kernel function to compute inner products in this space as:

$$K(x, y) = (1 + x_1 y_1 + x_2 y_2)^2$$

and we are using only the features from the input space

- Examples of functions that hold this condition are:
  - Polynomial kernel of degree d: $K(x, y) = (x^T y + 1)^d$
  - Gaussian function with width σ: $K(x, y) = \exp(-||x - y||^2/2\sigma^2)$
  - Sigmoid hiperbolical tangent with parameters k and θ: $K(x, y) = \tanh(kx^T y + \theta)$ (only for certain values of the parameters)

Can model decision boundary for infinite dimensional space

The Dual Problem- with Kernels

- Due to the introduction of the kernel function, the optimization problem has to be modified:

  Maximize $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$

  subject to $\sum_{i=1}^{n} \alpha_i y_i = 0, \quad C \geq \alpha_i \geq 0 \ \ \forall i$

- For classifying new examples:

$$h(z) = sign\left(\sum_{\forall i \in SV} \alpha_i, y_i k(x_i, z) + b\right)$$

- Since the training of the SVM only needs the value of $K(x_i, x_j)$ there is no constrains about how the examples are represented
- We only have to define the kernel as a similarity among examples

Take home message

Next Lecture

Introduction to Artificial Neural Networks