Ch. 3
A Tour of Machine Learning Classifiers Using Scikit-learn

Ray 3.6.2017

Choosing a classification algorithm

1. Selection of features.
2. Choosing a performance metric.
3. Choosing a classifier and optimization algorithm.
4. Evaluating the performance of the model.
5. Tuning the algorithm.

Training a perceptron via scikit-learn

scikit provides:

- datasets: iris
- cross-validatoion
- data preprocessing
- Why data standardization?

```
>>> from sklearn import datasets
>>> import numpy as np
>>> iris = datasets.loadiris()
>>> X = iris.data[:, [2, 3]]
>>> y = iris.target

>>> from sklearn.cross_validation import train_test_split
 >>> X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=0)

>>> from sklearn.preprocessing import StandardScaler
>>> sc = StandardScaler()
>>> sc.fit(X_train)
>>> X_train_std = sc.transform(X_train)
>>> X_test_std = sc.transform(X_test)
```

Modeling class probabilities via logistic regression

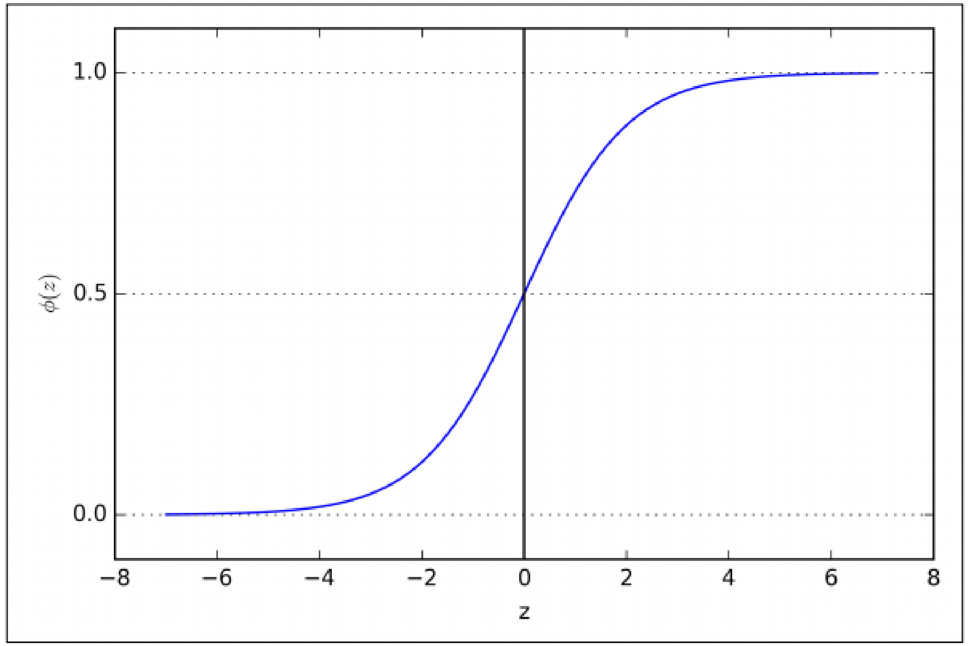
- Perceptron in practice, the weights are continuously being updated since there is always at least one misclassified sample present in each epoch
- the cost function is hard to converge.
- Perceptron is only linear seperatable

Logistic regression intuition and conditional probabilities

- p is the probability in favor of the event (y=1), $0 \leq p \leq 1$
- $z = \boldsymbol{w}^T \boldsymbol{x} = w_0 + w_1 x_1 + \cdots + w_m x_m$ imply probability but the range is entire real number

=> $z = \boldsymbol{w}^T \boldsymbol{x} = w_0 + w_1 x_1 + \cdots + w_m x_m = log \frac{p}{(1-p)}$

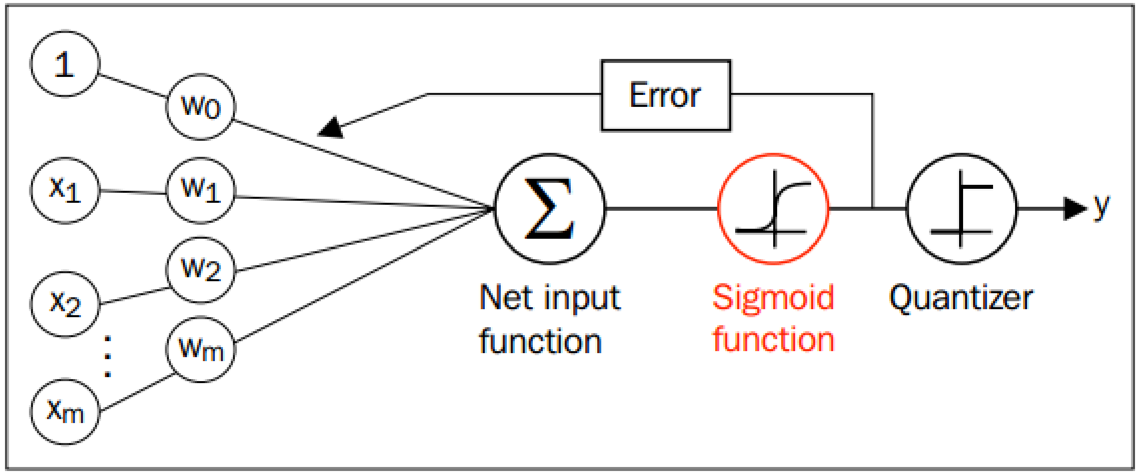
- Sigmoid function:
  => p= $\phi(z) = \frac{1}{1+e^{-z}}$



Logistic regression intuition and conditional probabilities

- Use Quantizer

$$\hat{y} = \begin{cases} 1 & if\, \phi(z) \geq 0.5 \\ 0 & otherwise \end{cases}$$

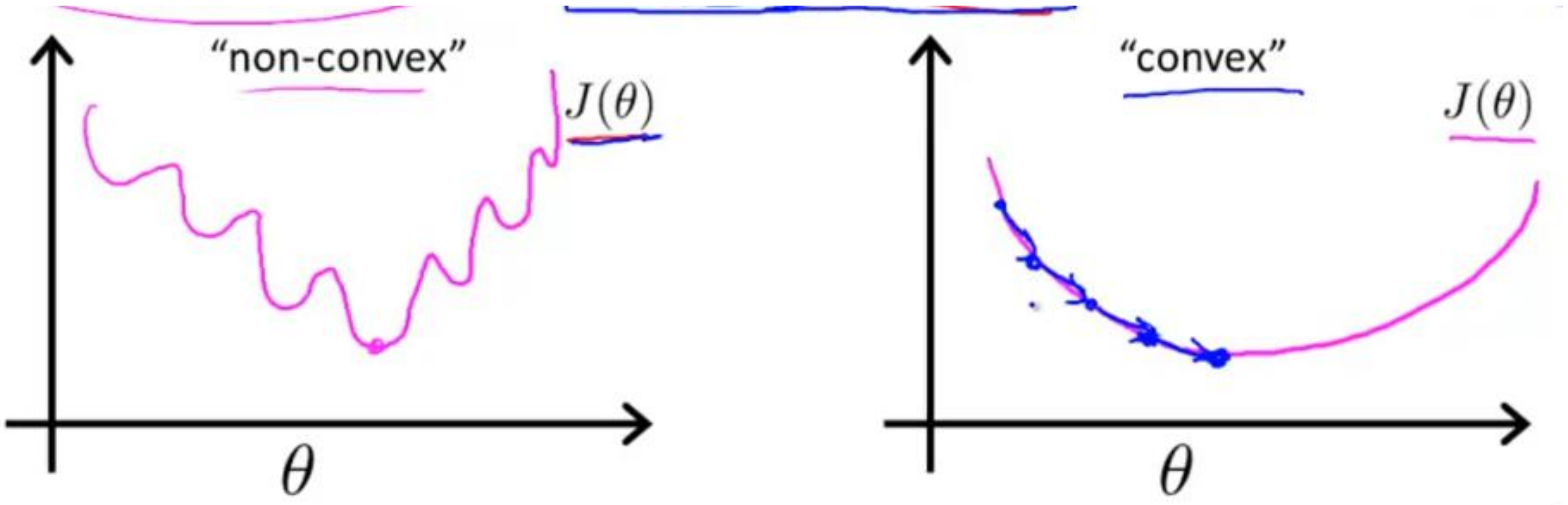


Learning the weights of the logistic cost function

- In previous chapter, the cost function is originally sum-squared-error cost function

$$J(w) = \sum_i \frac{1}{2}\left(\phi\left(z^{(i)}\right) - y^{(i)}\right)^2$$

- But this kind of cost function leads to non-convex, bad to optimize



Learning the weights of the logistic cost function

- Use likelihood function

$$L(\boldsymbol{w}) = P(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{w}) = \prod_{i=1}^{n} P\left(y^{(i)} \mid x^{(i)}; \boldsymbol{w}\right) = \prod_{i=1}^{n} \left(\phi\left(z^{(i)}\right)\right)^{y^{(i)}} \left(1 - \phi\left(z^{(i)}\right)\right)^{1-y^{(i)}}$$

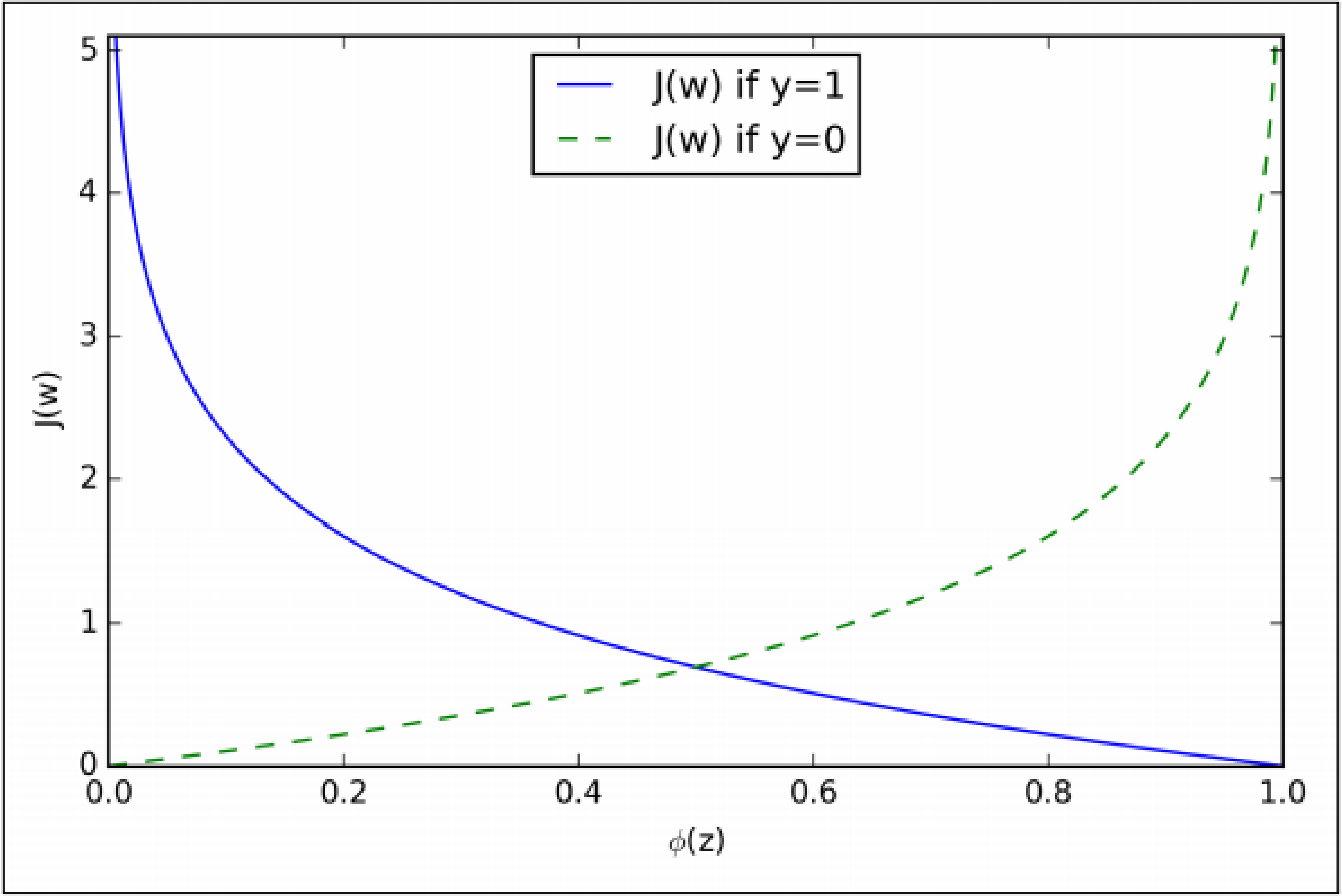
- Log for easy to optimize

$$\log L(\boldsymbol{w}) = \sum_{i=1}^{n} \left[y^{(i)} \log\left(\phi\left(z^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - \phi\left(z^{(i)}\right)\right) \right]$$

- what we need to do is maximize $\log L(\boldsymbol{w})$
- for easy to optimize (gradient descend), add negative sign

$$J(\boldsymbol{w}) = \sum_{i=1}^{n} \left[-y^{(i)} \log\left(\phi\left(z^{(i)}\right)\right) - \left(1 - y^{(i)}\right) \log\left(1 - \phi\left(z^{(i)}\right)\right) \right]$$

- minimize J(w)


J(w) if y=1
J(w) if y=0
J(w)
φ(z)
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5

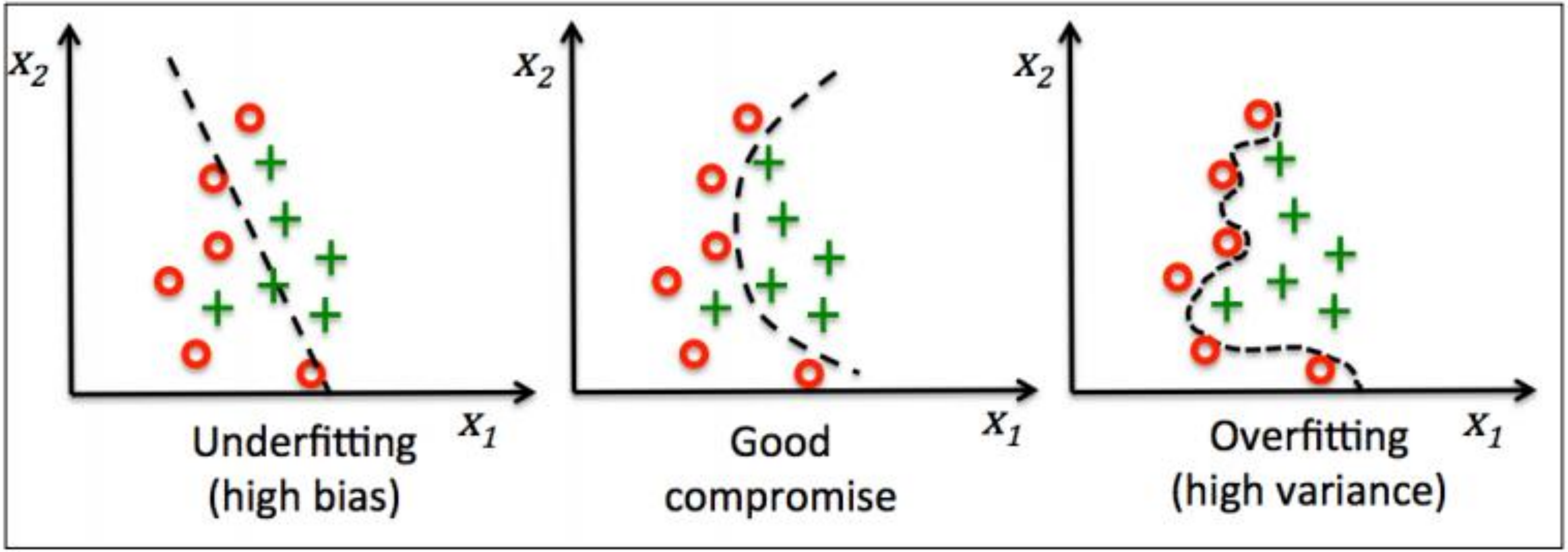
Learning the weights of the logistic cost function

- update weights w

$$\boldsymbol{w} := \boldsymbol{w} + \Delta \boldsymbol{w}$$

$$\Delta w_j = -\eta \frac{\partial J}{\partial w_j} = \eta \sum_{i=1}^{n} \left(y^{(i)} - \phi\left(z^{(i)}\right) \right) x_j^{(i)}$$

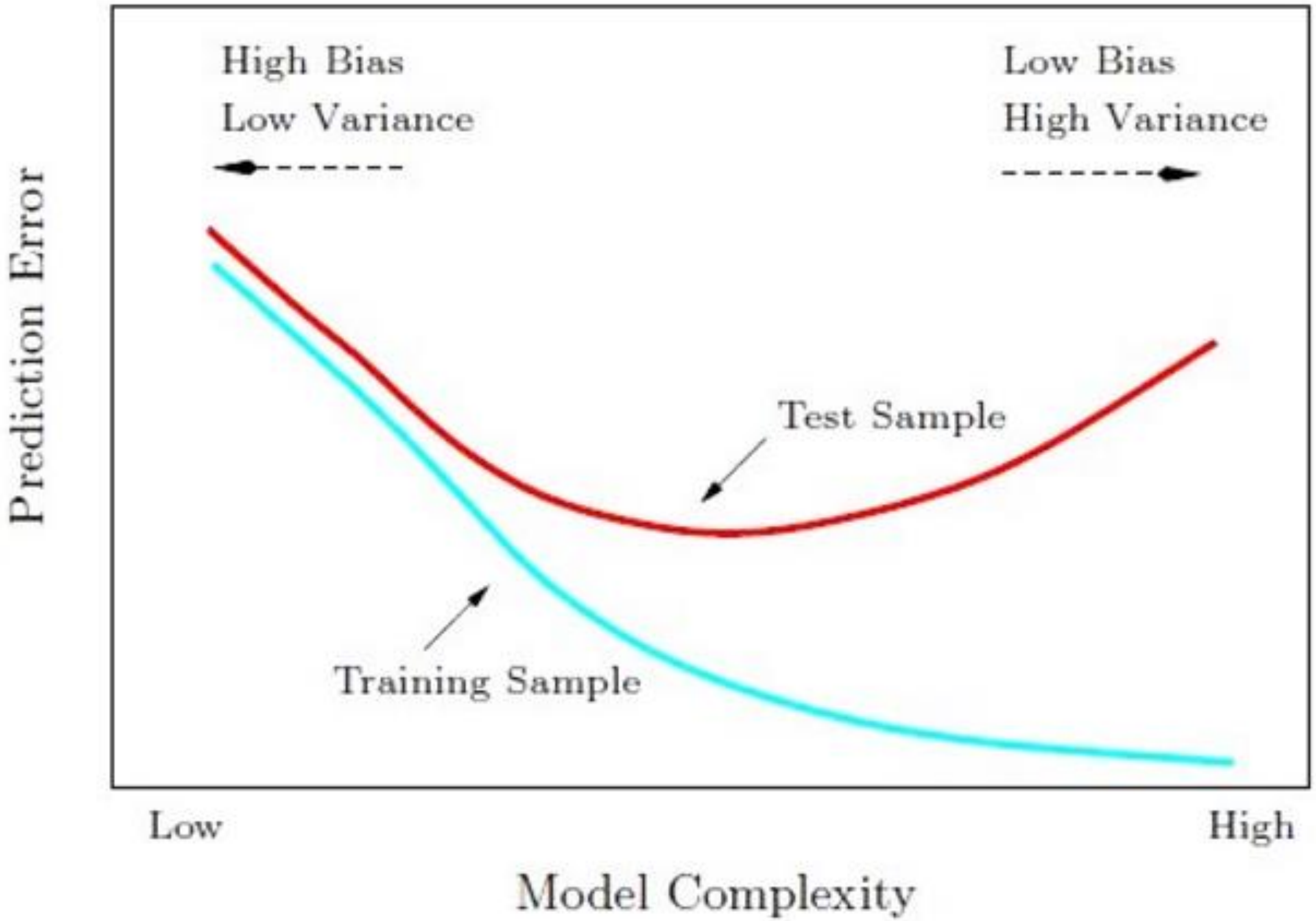
Tackling overfitting via regularization



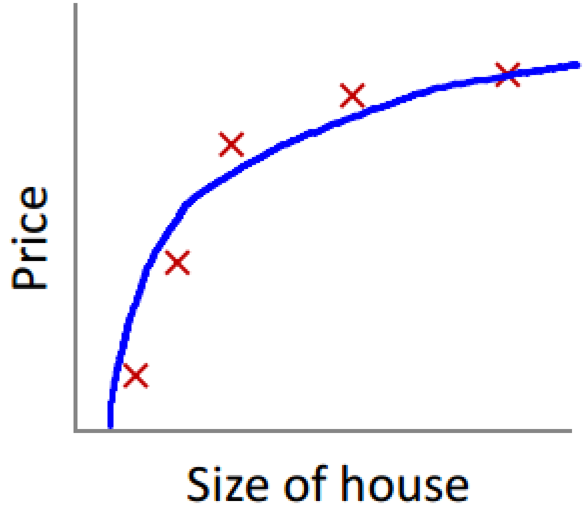
Variance versus Bias

- Variance: 當用不同training datasets得出來的models來預測, 所得預測結果的一致性,
- Bias:當用不同training datasets得出來的models來預測,所得預測結果與實際Label的差值, 差值越高, bias越高
- 兩個極端的例子：
  - 記住訓練集合上所有data的label，這樣的系統是**低bias**、高variance。
  - 無論輸入什麼data，總是預測一個相同的label，這樣的系統是高bias、**低variance**。

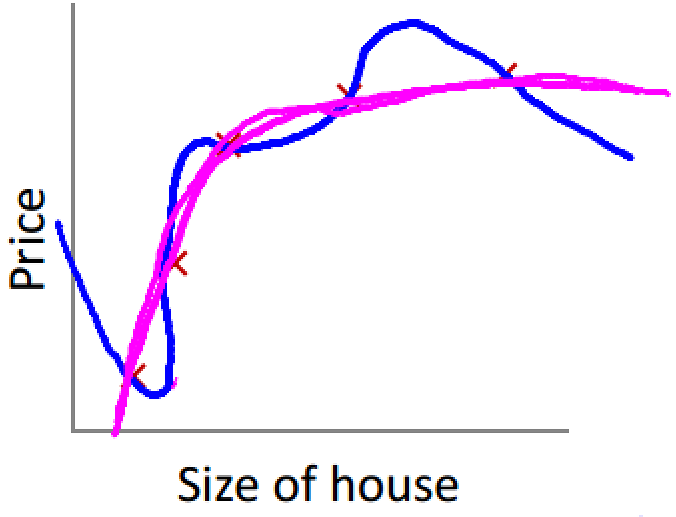
Variance versus Bias



Intuition



$$\theta_0 + \theta_1 x + \theta_2 x^2$$



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$$

Suppose we penalize and make θ_3, θ_4 really small.

$$\rightarrow \min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$$

$\theta_3 \approx 0 \qquad \theta_4 \approx 0$

Regularization.

Small values for parameters $\boxed{\theta_0, \theta_1, \dots, \theta_n}$

- "Simpler" hypothesis
- Less prone to overfitting

$\boxed{\theta_3, \theta_4}$ ≈ 0

Housing:

- Features: $x_1, x_2, \dots, x_{100}$
- Parameters: $\theta_0, \theta_1, \theta_2, \dots, \theta_{100}$

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda \sum_{i=1}^{n} \theta_j^2\right]$$

$\cancel{\theta_0}$ $\theta_1, \theta_2, \theta_3, \dots, \theta_{100}$

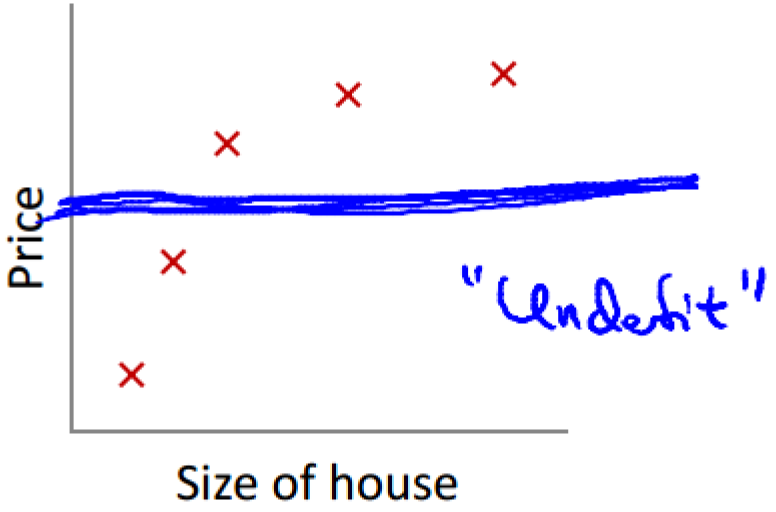
$\cancel{\theta_0}$

Andrew Ng

In regularized linear regression, we choose θ to minimize

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \underline{\lambda}\sum_{j=1}^{n}\theta_j^2\right]$$

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?



$h_\theta(x)$

$$\theta_0 + \theta_1 \cancel{x} + \theta_2 \cancel{x^2} + \theta_3 \cancel{x^3} + \theta_4 \cancel{x^4}$$

$\theta_1, \theta_2, \theta_3, \theta_4$

$\theta_1 \approx 0, \theta_2 \approx 0$

$\theta_3 \approx 0, \theta_4 \approx 0$

$$\boxed{h_\theta(x) = \theta_0}$$

Andrew Ng

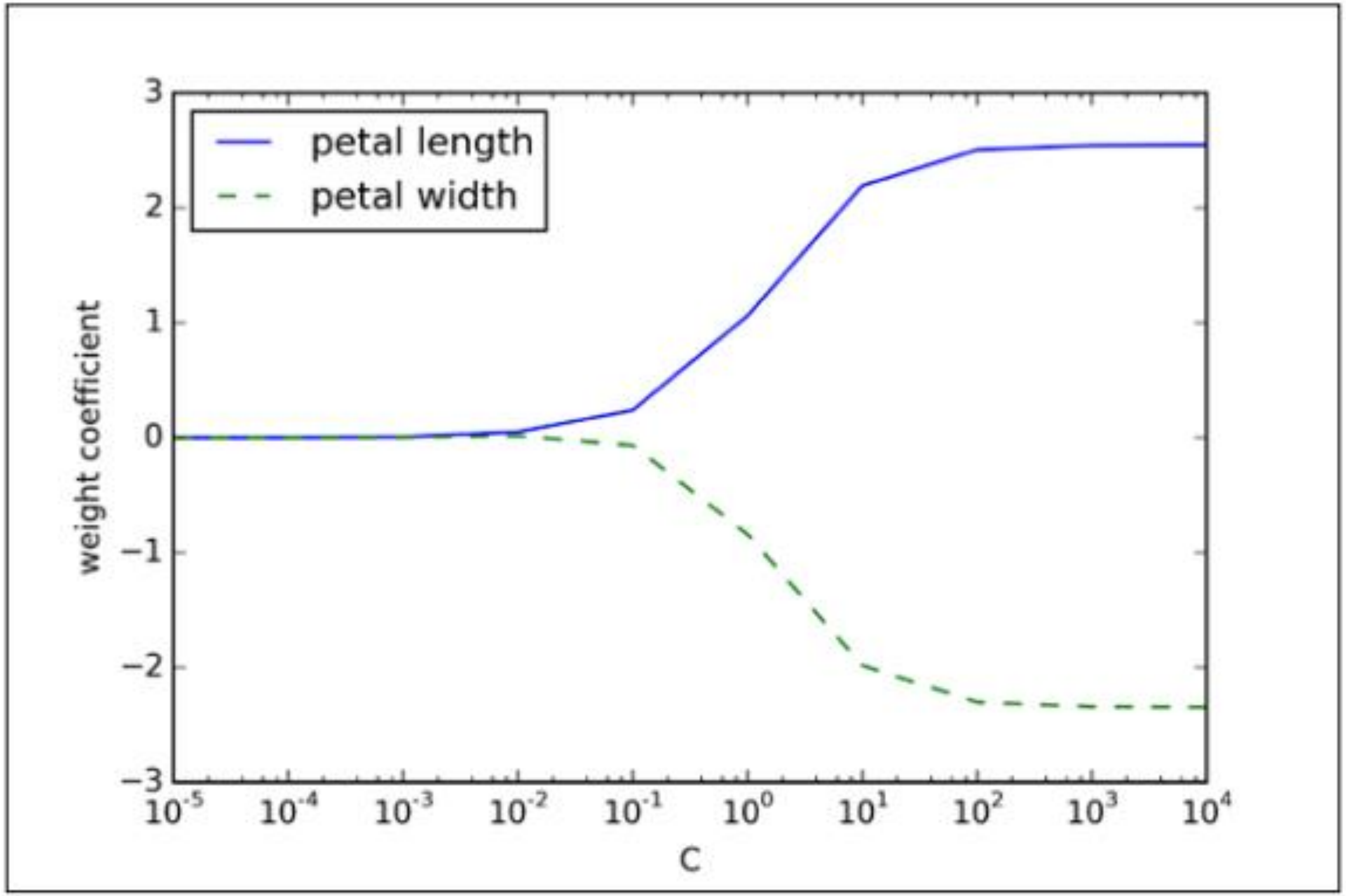
L2-regularization

$$\frac{\lambda}{2}\|\boldsymbol{w}\|^2 = \frac{\lambda}{2}\sum_{j=1}^{m} w_j^2$$

$$C = \frac{1}{\lambda}$$

$$J(\boldsymbol{w}) = \sum_{i=1}^{n}\left[-y^{(i)}\log\left(\phi\left(z^{(i)}\right)\right) - \left(1-y^{(i)}\right)\log\left(1-\phi\left(z^{(i)}\right)\right)\right] + \frac{\lambda}{2}\|w\|^2$$

$$J(\boldsymbol{w}) = C\left[\sum_{i=1}^{n}\left(-y^{(i)}\log\left(\phi\left(z^{(i)}\right)\right) - \left(1-y^{(i)}\right)\right)\log\left(1-\phi\left(z^{(i)}\right)\right)\right] + \frac{1}{2}\|\boldsymbol{w}\|^2$$

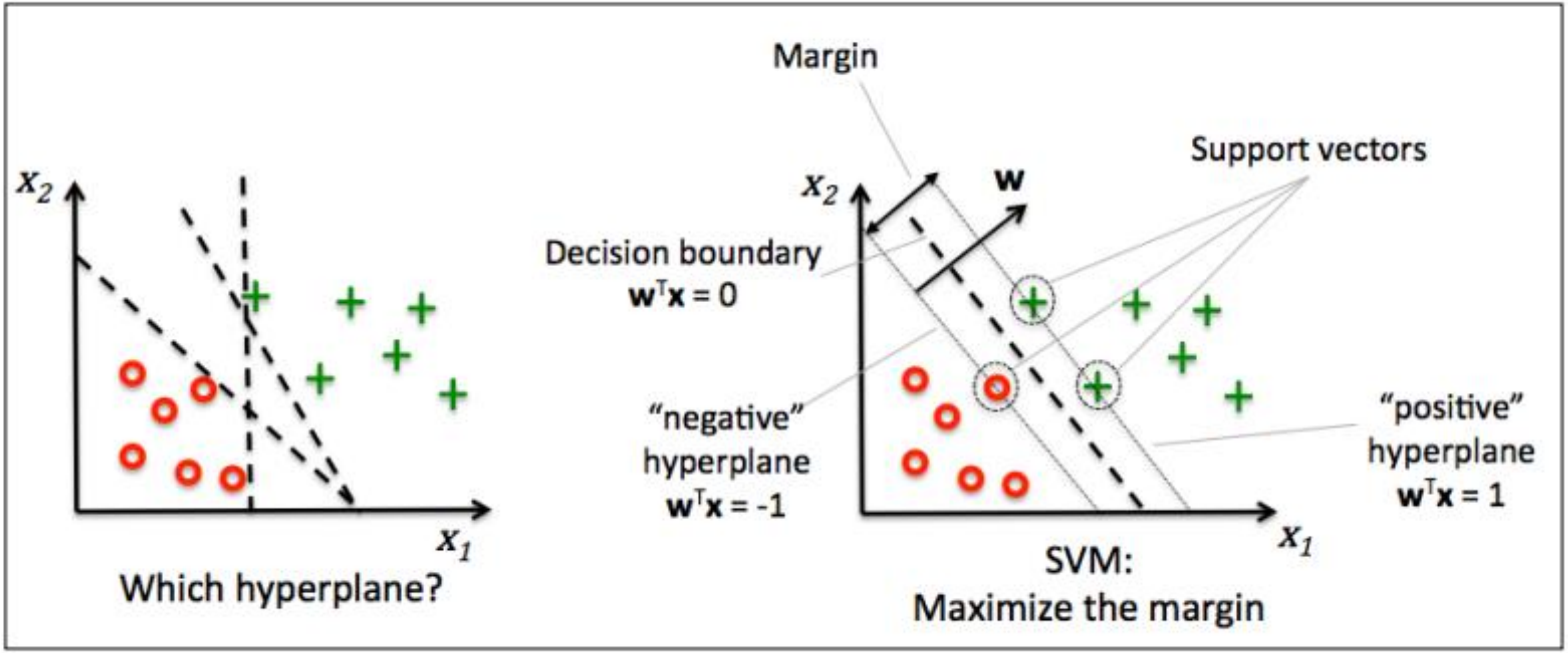


regularization strength ←

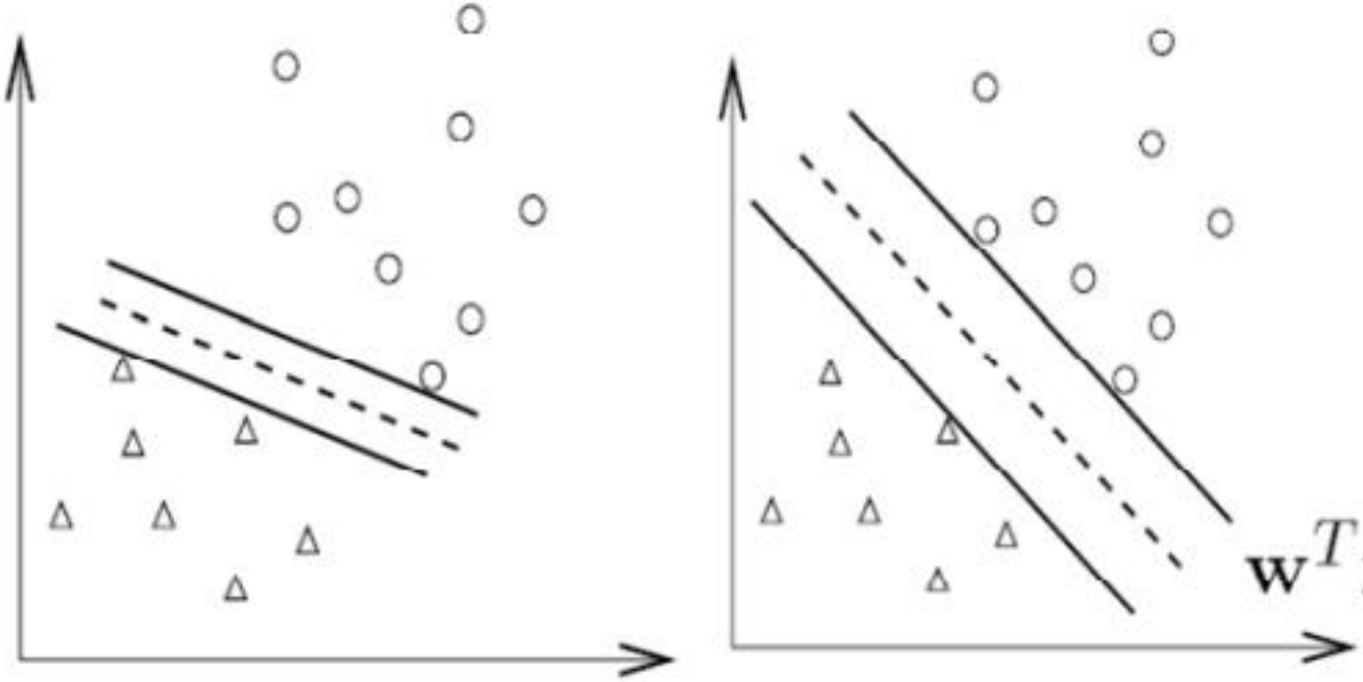
SVM: Maximum margin classification with support vector machines

- The margin is defined as the distance between the separating hyperplane (decision boundary) and the training samples that are closest to this hyperplane, which are the so-called support vectors
- optimization objective is to maximize the margin

- Support Vector是指在Training data set中，用於分類上給予最多資訊的點



x_2
x_1
Which hyperplane?
Margin
Support vectors
x_2
w
Decision boundary
$\mathbf{w}^T\mathbf{x} = 0$
"negative"
hyperplane
$\mathbf{w}^T\mathbf{x} = -1$
"positive"
hyperplane
$\mathbf{w}^T\mathbf{x} = 1$
x_1
SVM:
Maximize the margin



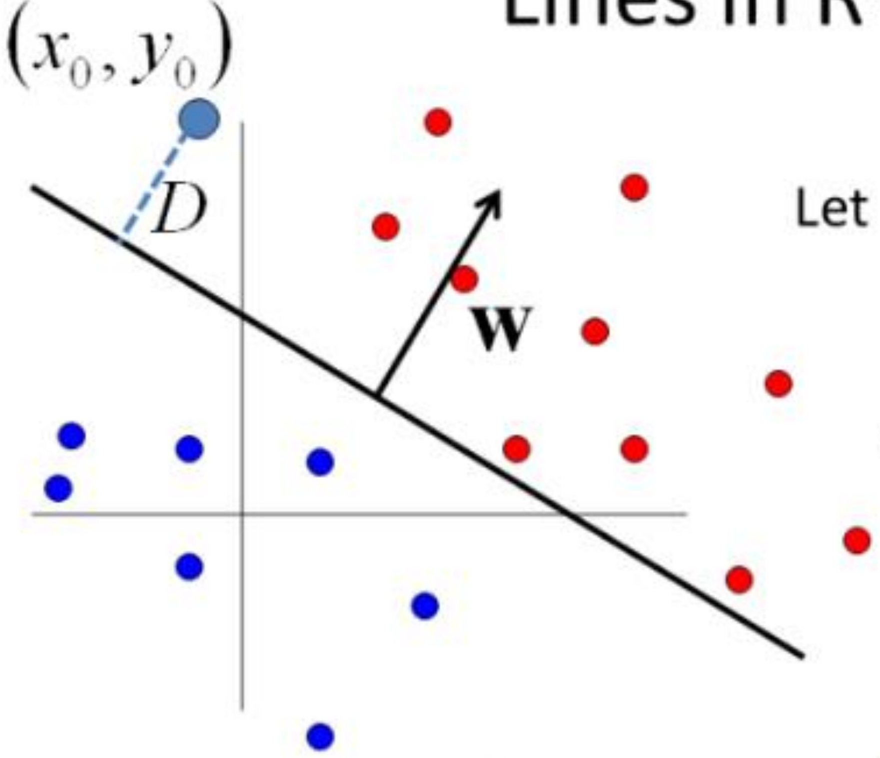
$$\mathbf{w}^T\mathbf{x} + b = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix}$$

Maximum margin intuition

- decision boundaries with large margins is that they tend to have a lower generalization error
- whereas models with small margins are more prone to overfitting

Maximum margin intuition

Lines in R^2



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

$$ax + cy + w_0 = 0$$

$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + w_0 = 0$$

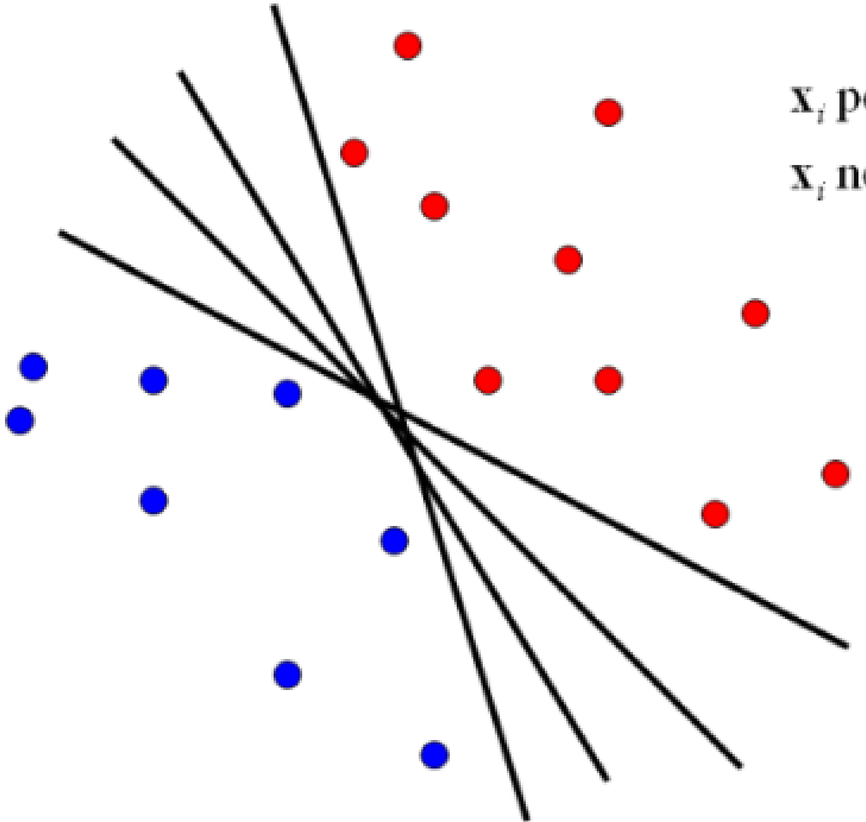
$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + b}{|\mathbf{w}|}$$

distance from point to line

Maximum margin intuition

Linear classifiers

- Find linear function to separate positive and negative examples



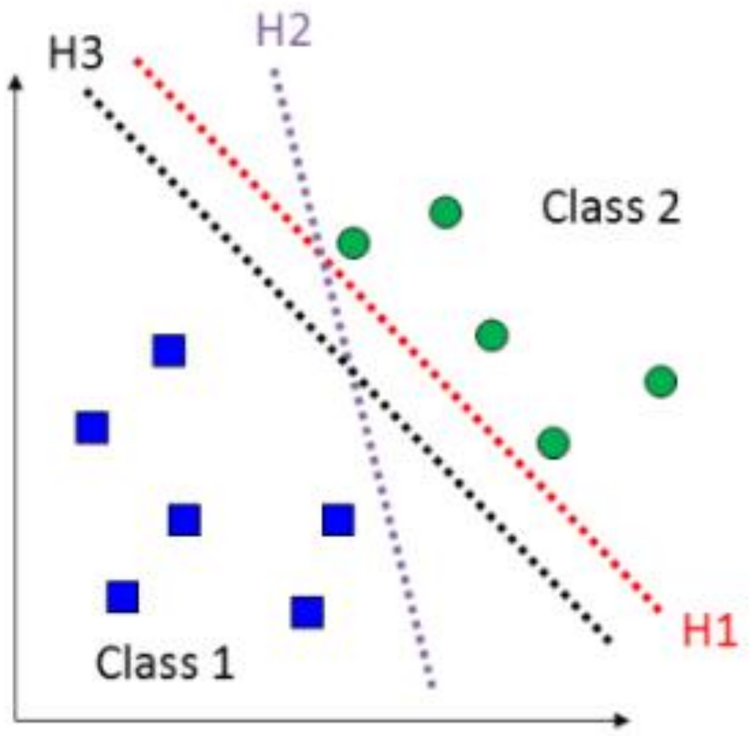
$\mathbf{x}_i$ positive : $\quad \mathbf{x}_i \cdot \mathbf{w} + w_0 \geq 0$

$\mathbf{x}_i$ negative : $\quad \mathbf{x}_i \cdot \mathbf{w} + w_0 < 0$

How many lines we may have to separate the data?

Which line is best?

Maximum margin intuition



Hyperplanes H1, H2, and H3 are candidate classifiers.
Which one is preferred? Why?

Maximum margin intuition

- objective function:

$$w_0 + \boldsymbol{w}^T \boldsymbol{x}^{(i)} \geq 1 \; if \; y^{(i)} = 1$$

$$w_0 + \boldsymbol{w}^T \boldsymbol{x}^{(i)} < -1 \; if \; y^{(i)} = -1$$

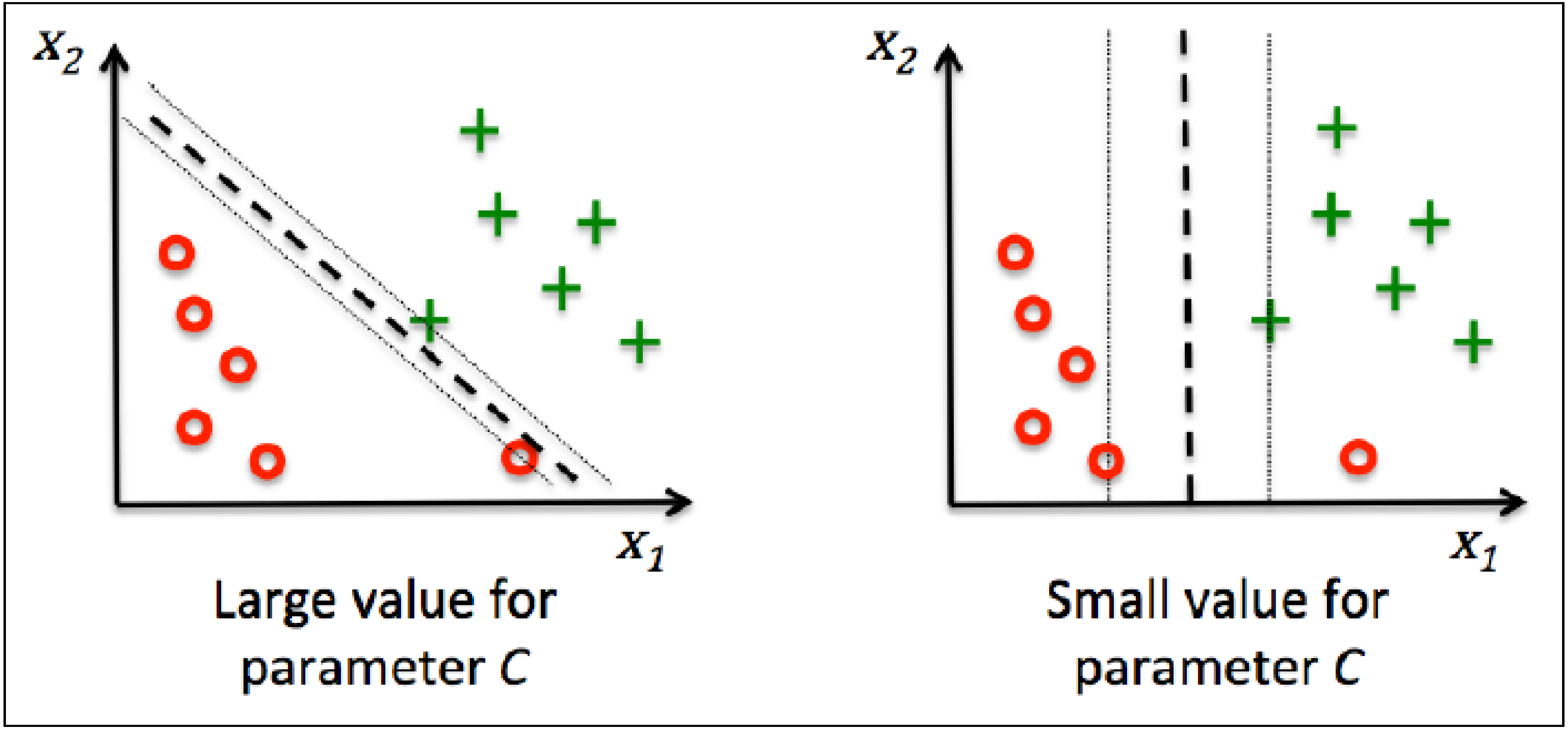
- cost function & optimization:
  - quadratic programming

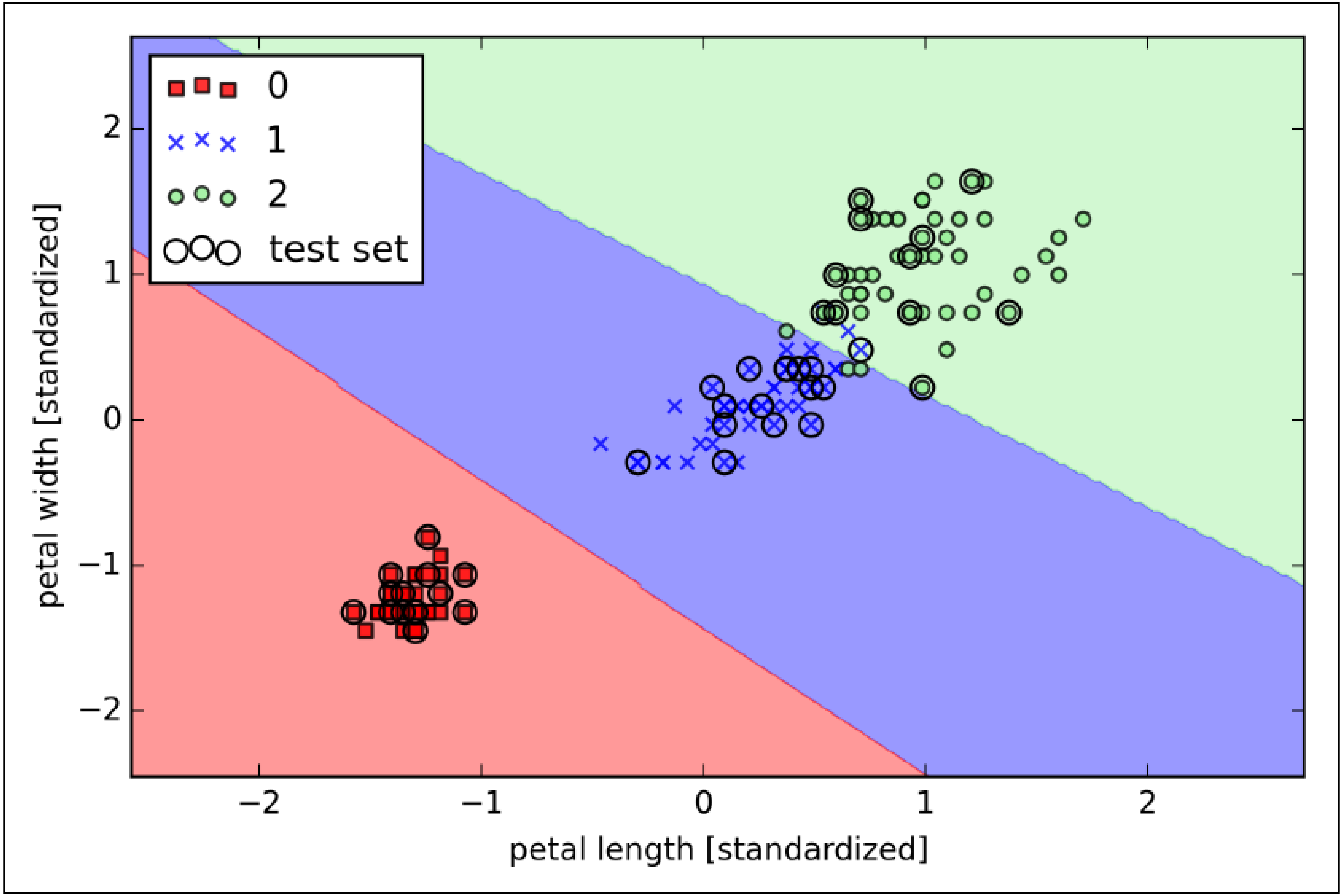
Dealing with the nonlinearly separable case using slack variables

$$\boldsymbol{w}^T \boldsymbol{x}^{(i)} \geq 1 - \xi^{(i)} \; if \;\; y^{(i)} = 1$$

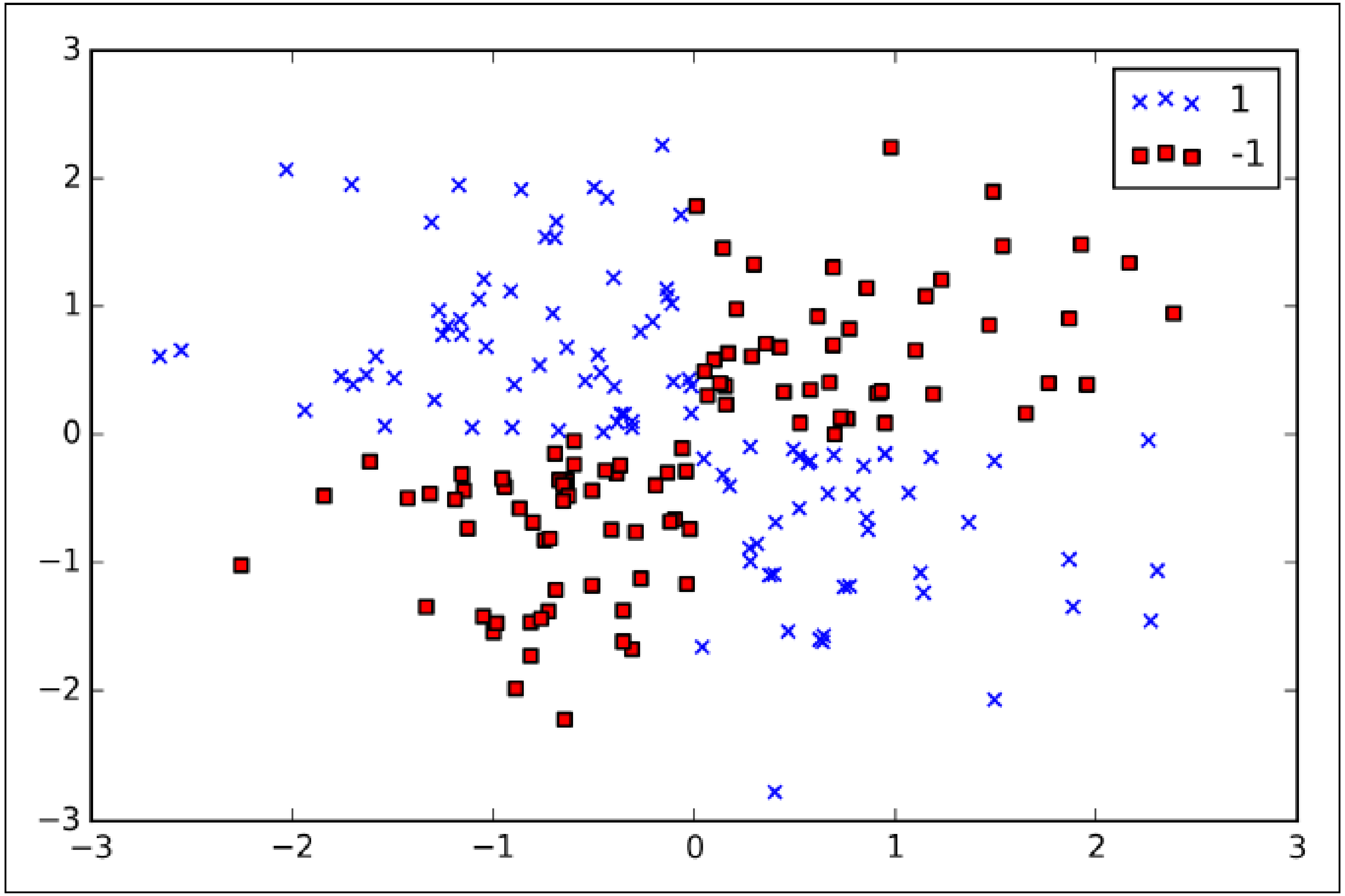
$$\boldsymbol{w}^T \boldsymbol{x}^{(i)} \leq -1 + \xi^{(i)} \; if \;\; y^{(i)} = -1$$

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\left(\sum_i \xi^{(i)}\right)$$


x_2
x_1
Large value for parameter C
x_2
x_1
Small value for parameter C


0
1
2
test set
petal width [standardized]
petal length [standardized]
−2
−1
0
1
2

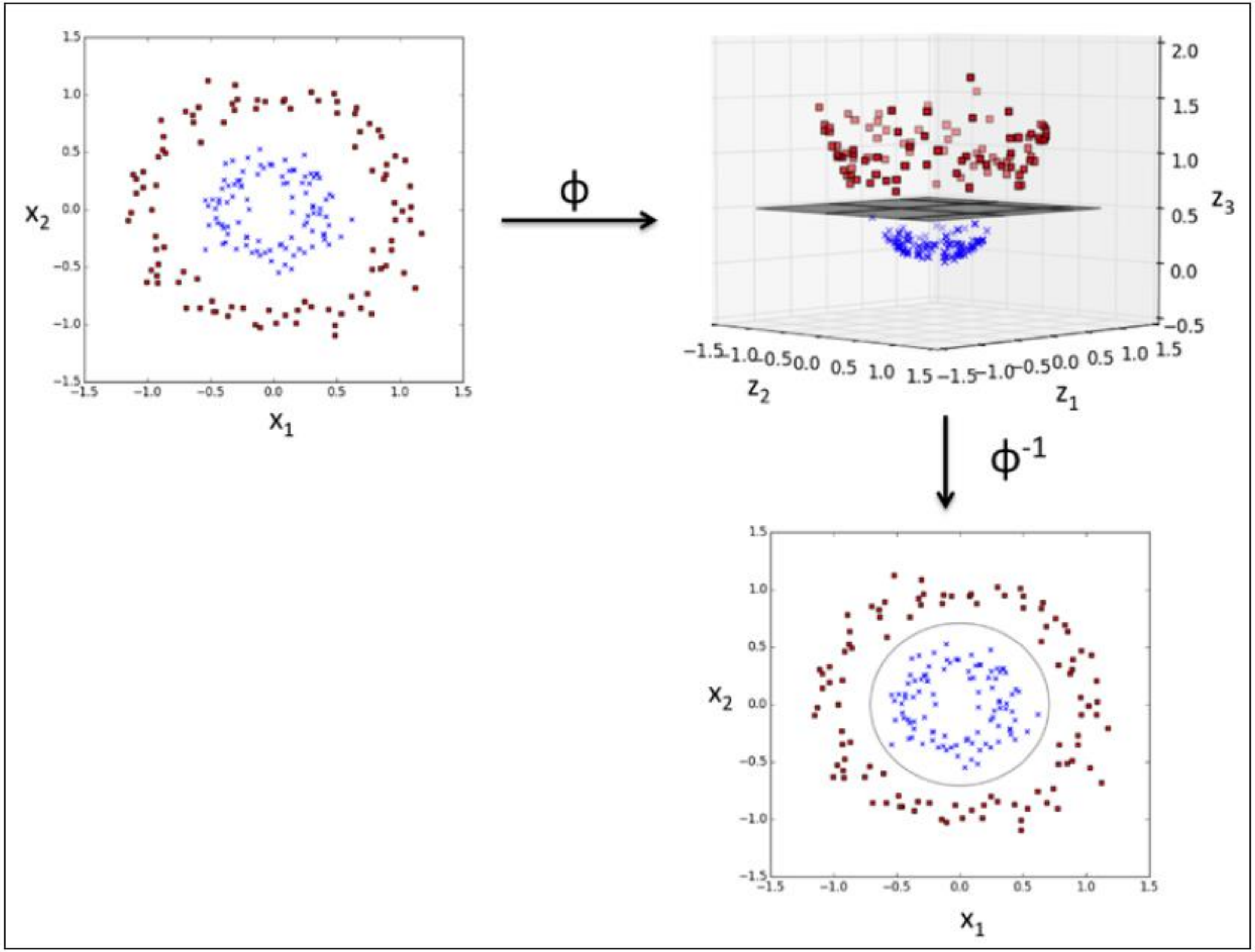
Solving nonlinear problems using a kernel SVM



Solving nonlinear problems using a kernel SVM

- original features to project them onto a higher dimensional space via a mapping function $\phi(\cdot)$

$$\phi(x_1, x_2) = (z_1, z_2, z_3) = \left(x_1, x_2, x_1^2 + x_2^2\right)$$



x1
x2
ϕ
z1
z2
z3
ϕ-1

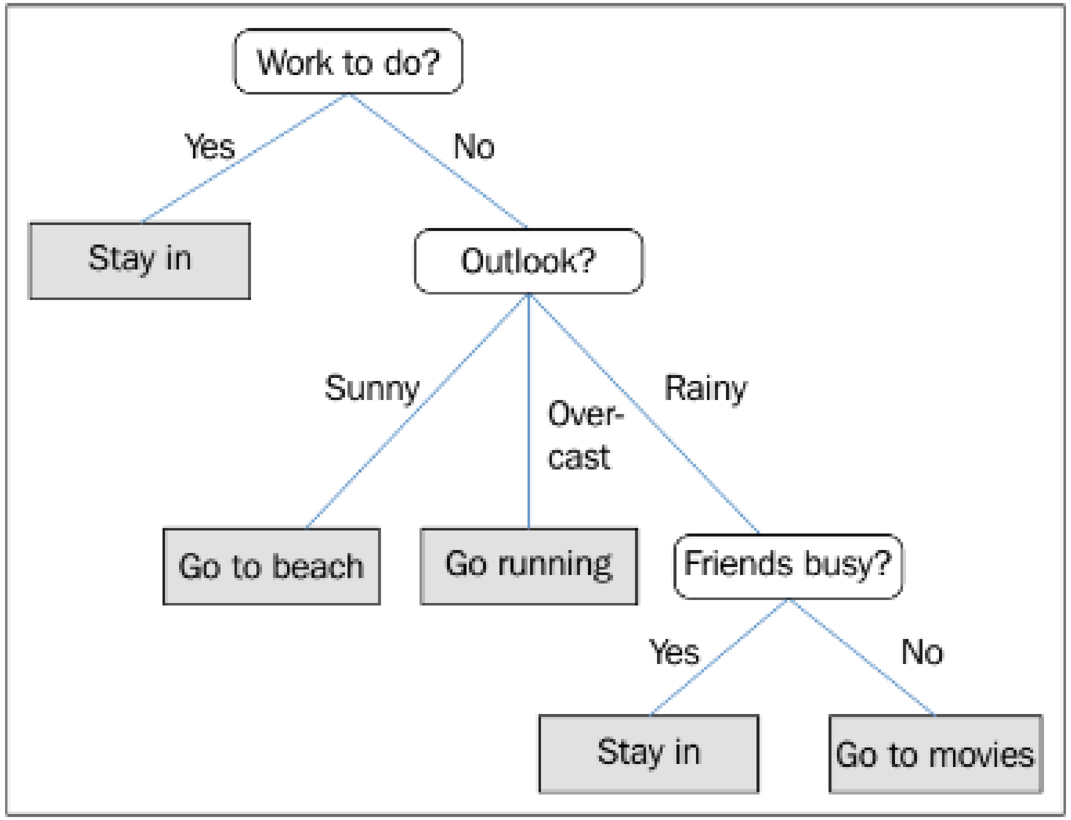
Using the kernel trick to find separating hyperplanes in higher dimensional space

Decision tree learning

- Information gain(IG): 代表從這個if-else question中獲得的有效資訊

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j)$$

- I(D), impurity measures:
  代表在datasets D 中, 資料的亂度

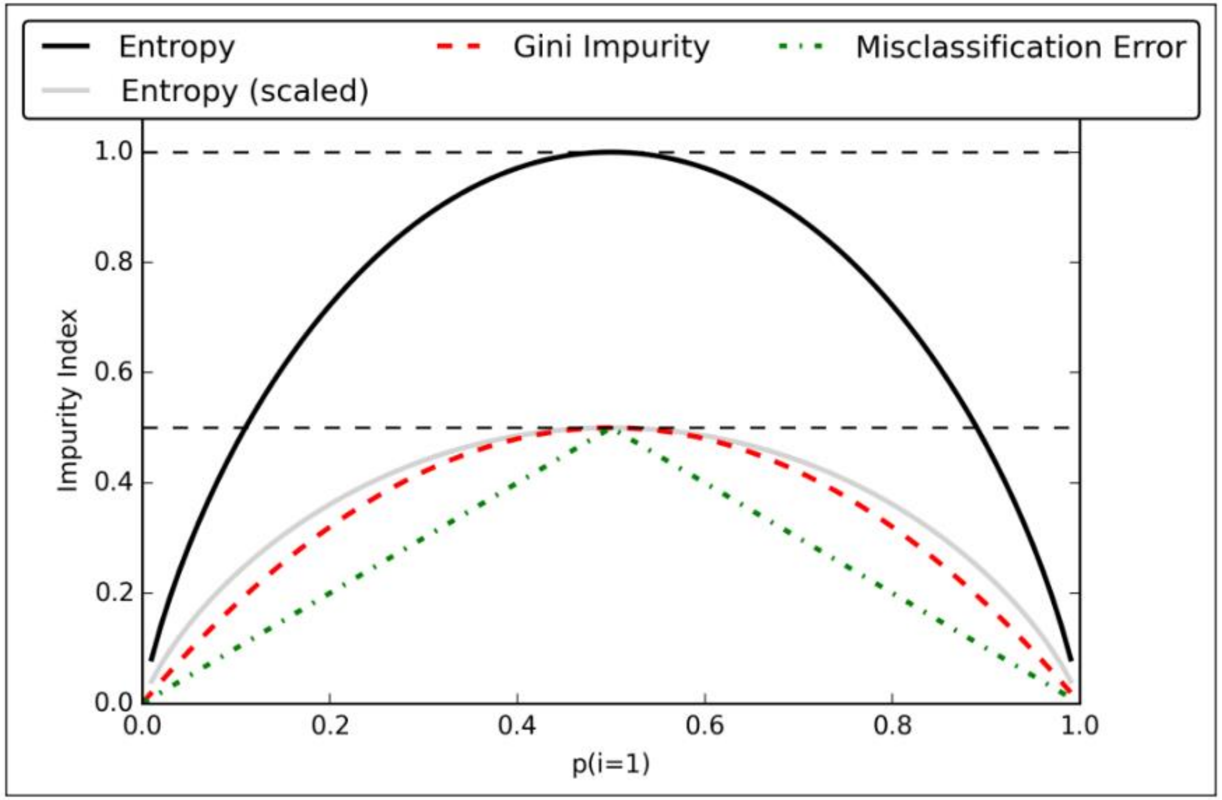


Decision tree learning: Impurity Measures

- I(D), impurity measures 有三種:
  - Entropy: $I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t)$
  - Gini impurity: $I_G(t) = \sum_{i=1}^{c} p(i|t)\left(1 - p(i|t)\right) = 1 - \sum_{i=1}^{c} p(i|t)^2$
  - Classification error: $I_E = 1 - \max\left\{p(i|t)\right\}$
  - $p(i|t)$: class i 在 data subset t分布的程度

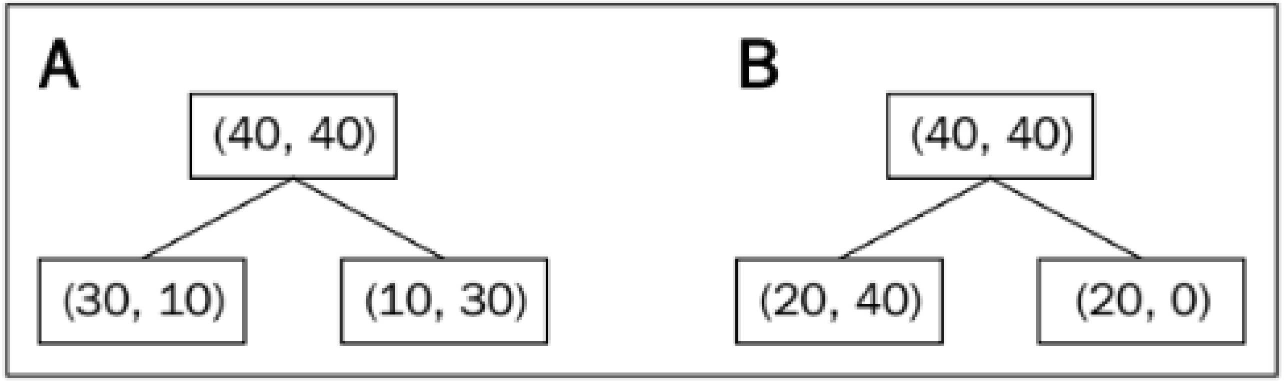
Decision tree learning: Impurity Measures

- In a binary class setting,
  - the entropy is 0 if $p(i=1|t)=1$ or $p(i=0|t)=0$,
  - If the classes are di $p(i=1|t)=0.5$ and $p(i=0|t)=0.5$, the entropy is 1.



$$I_E = 1 - \max\{p(i|t)\}$$

Decision tree learning example (Classification error)



$$I_E(D_p) = 1 - 0.5 = 0.5$$

$$A: I_E(D_{left}) = 1 - \frac{3}{4} = 0.25 \qquad B: I_E(D_{left}) = 1 - \frac{4}{6} = \frac{1}{3}$$

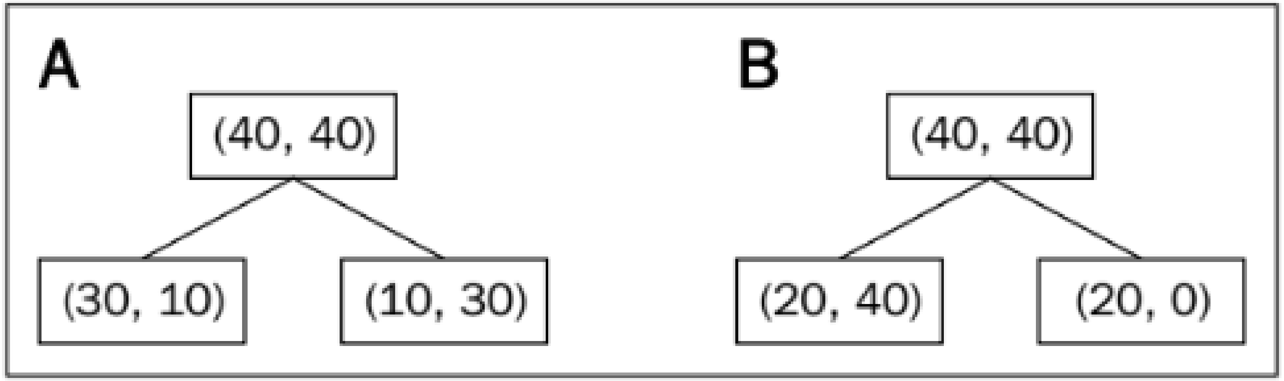
$$A: I_E(D_{right}) = 1 - \frac{3}{4} = 0.25 \qquad B: I_E(D_{right}) = 1 - 1 = 0$$

A is as good as B

$$\boxed{A: IG_E = 0.5 - \frac{4}{8}0.25 - \frac{4}{8}0.25 = 0.25 \qquad B: IG_E = 0.5 - \frac{6}{8} \times \frac{1}{3} - 0 = 0.25}$$

$$I_G(t) = \sum_{i=1}^{c} p(i|t)\left(1 - p(i|t)\right) = 1 - \sum_{i=1}^{c} p(i|t)^2$$

Decision tree learning example (Gini impurity)



$$I_G(D_p) = 1 - \left(0.5^2 + 0.5^2\right) = 0.5$$

$$A: I_G(D_{left}) = 1 - \left(\left(\frac{3}{4}\right)^2 + \left(\frac{1}{4}\right)^2\right) = \frac{3}{8} = 0.375$$

$$B: I_G(D_{left}) = 1 - \left(\left(\frac{2}{6}\right)^2 + \left(\frac{4}{6}\right)^2\right) = \frac{4}{9} = 0.\overline{4}$$

$$A: I_G(D_{right}) = 1 - \left(\left(\frac{1}{4}\right)^2 + \left(\frac{3}{4}\right)^2\right) = \frac{3}{8} = 0.375$$

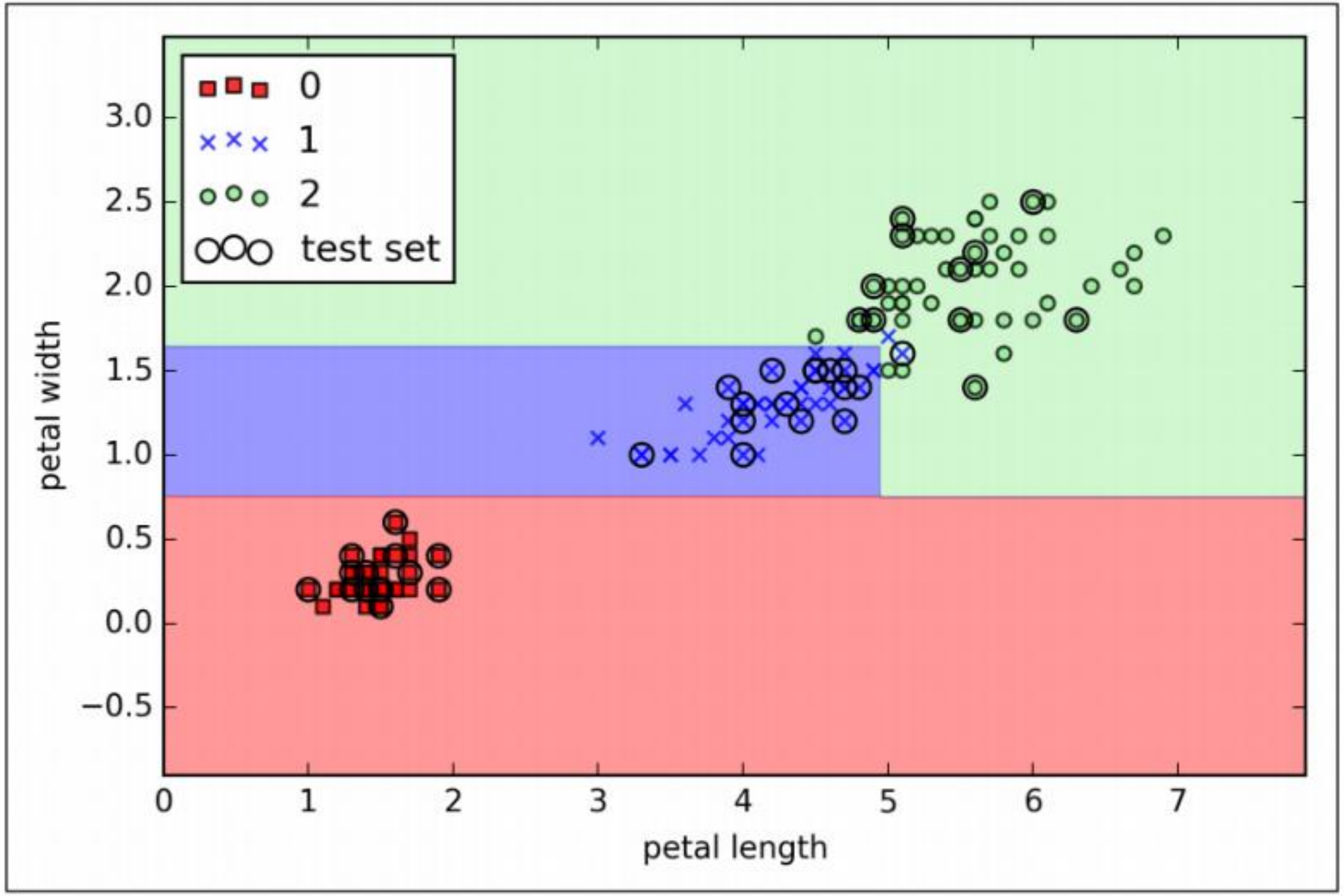
$$B: I_G(D_{right}) = 1 - \left(1^2 + 0^2\right) = 0$$

B is better

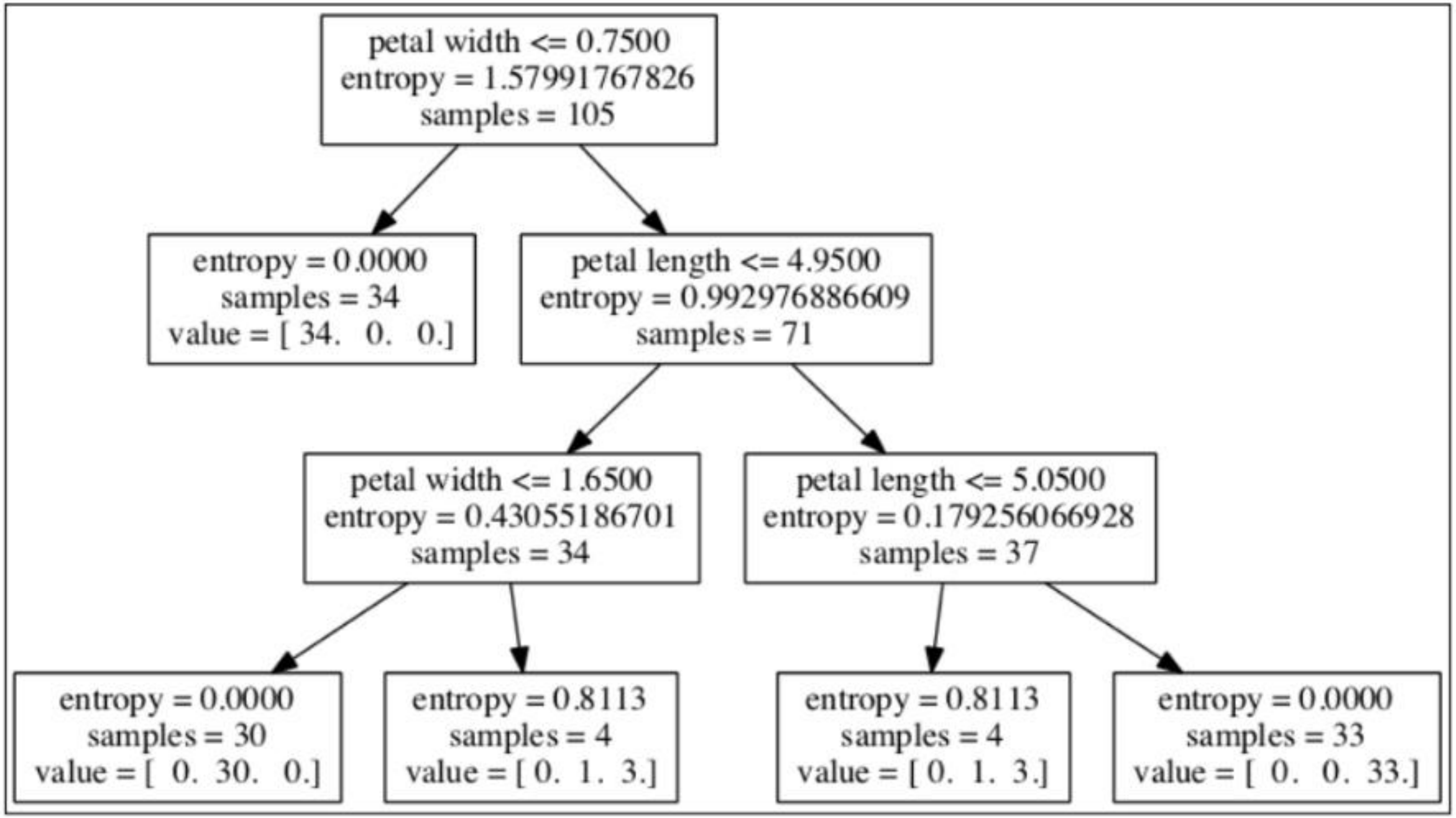
$$A: IG_G = 0.5 - \frac{4}{8}0.375 - \frac{4}{8}0.375 = 0.125$$

$$B: IG_G = 0.5 - \frac{6}{8}0.\overline{4} - 0 = 0.1\overline{6}$$

Building a decision tree via scikit



Building a decision tree via scikit



Random Forest

Create random subsets

$$S_1 = \begin{bmatrix} f_{A12} & f_{B12} & f_{C12} & C_{12} \\ f_{A15} & f_{B15} & f_{C15} & C_{15} \\ \vdots & & \vdots & \\ f_{A35} & f_{B35} & f_{C35} & C_{35} \end{bmatrix} \quad S_2 = \begin{bmatrix} f_{A2} & f_{B2} & f_{C2} & C_2 \\ f_{A6} & f_{B6} & f_{C6} & C_6 \\ \vdots & & \vdots & \\ f_{A20} & f_{B20} & f_{C20} & C_{20} \end{bmatrix}$$

$$S_M = \begin{bmatrix} f_{A4} & f_{B4} & f_{C4} & C_4 \\ f_{A9} & f_{B9} & f_{C9} & C_9 \\ \vdots & & \vdots & \\ f_{A12} & f_{B12} & f_{C12} & C_{12} \end{bmatrix}$$

Create random subsets

$$S_1 = \begin{bmatrix} f_{A12} & f_{B12} & f_{C12} & C_{12} \\ f_{A15} & f_{B15} & f_{C15} & C_{15} \\ \vdots & & \vdots & \\ f_{A35} & f_{B35} & f_{C35} & C_{35} \end{bmatrix} S_2 = \begin{bmatrix} f_{A2} & f_{B2} & f_{C2} & C_{2} \\ f_{A6} & f_{B6} & f_{C6} & C_{6} \\ \vdots & & \vdots & \\ f_{A20} & f_{B20} & f_{C20} & C_{20} \end{bmatrix}$$

$$S_M = \begin{bmatrix} f_{A4} & f_{B4} & f_{C4} & C_{4} \\ f_{A9} & f_{B9} & f_{C9} & C_{9} \\ \vdots & & \vdots & \\ f_{A12} & f_{B12} & f_{C12} & C_{12} \end{bmatrix}$$

Create random subsets

$$S_1 = \begin{bmatrix} f_{A12} & f_{B12} & f_{C12} & C_{12} \\ f_{A15} & f_{B15} & f_{C15} & C_{15} \\ \vdots & & \vdots & \\ f_{A35} & f_{B35} & f_{C35} & C_{35} \end{bmatrix} \quad S_2 = \begin{bmatrix} f_{A2} & f_{B2} & f_{C2} & C_{2} \\ f_{A6} & f_{B6} & f_{C6} & C_{6} \\ \vdots & & \vdots & \\ f_{A20} & f_{B20} & f_{C20} & C_{20} \end{bmatrix}$$

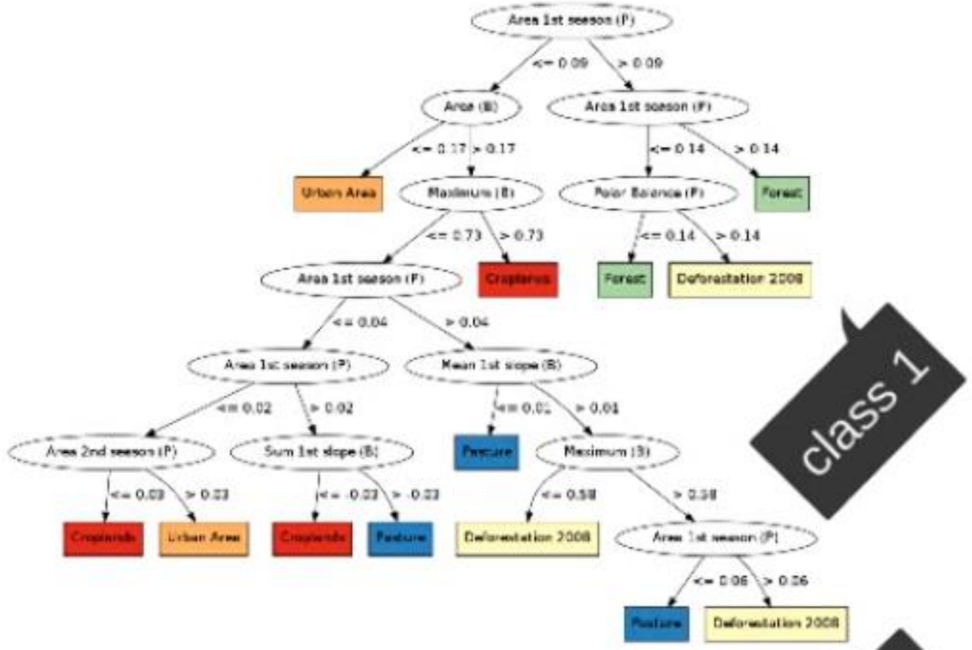
Decision tree 1

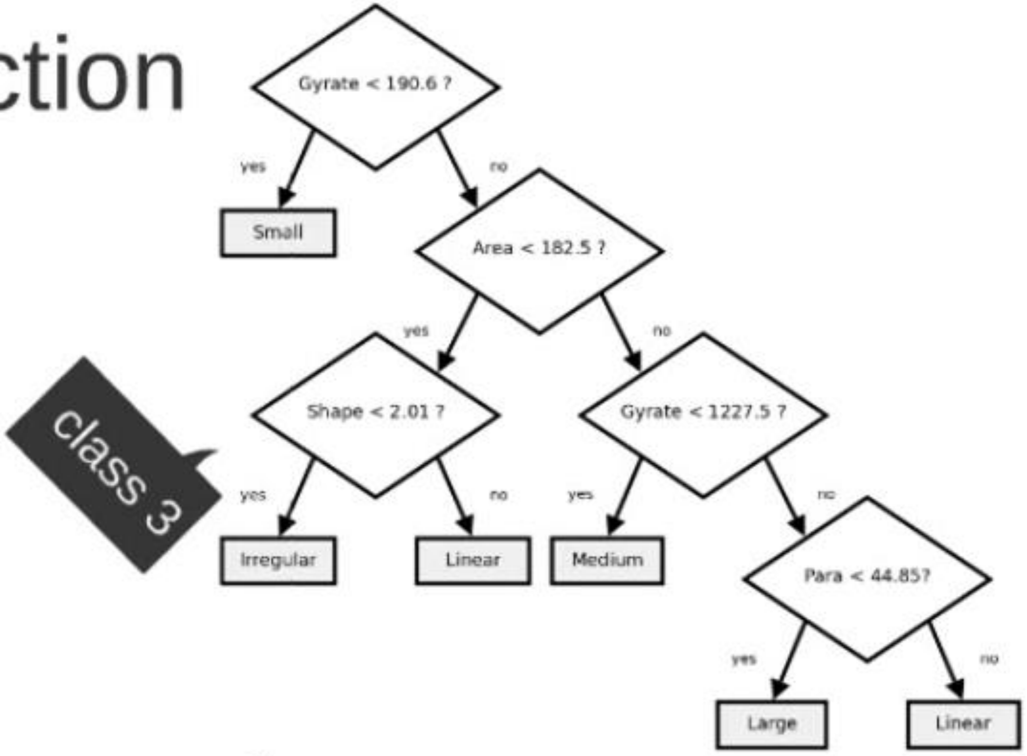
$$S_M = \begin{bmatrix} f_{A4} & f_{B4} & f_{C4} & C_{4} \\ f_{A9} & f_{B9} & f_{C9} & C_{9} \\ \vdots & & \vdots & \\ f_{A12} & f_{B12} & f_{C12} & C_{12} \end{bmatrix}$$

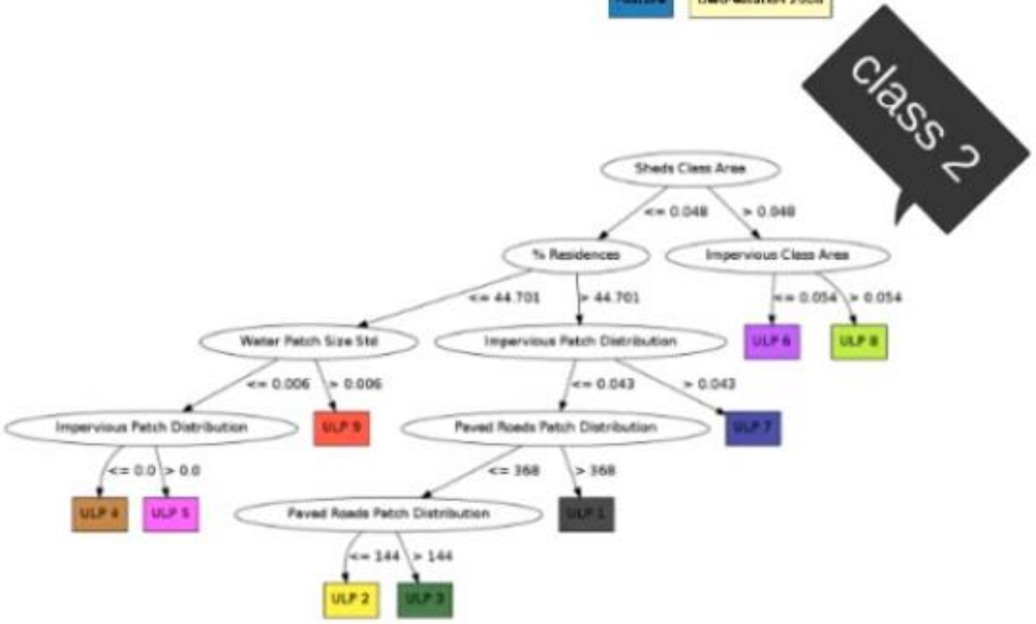
Decision tree 2

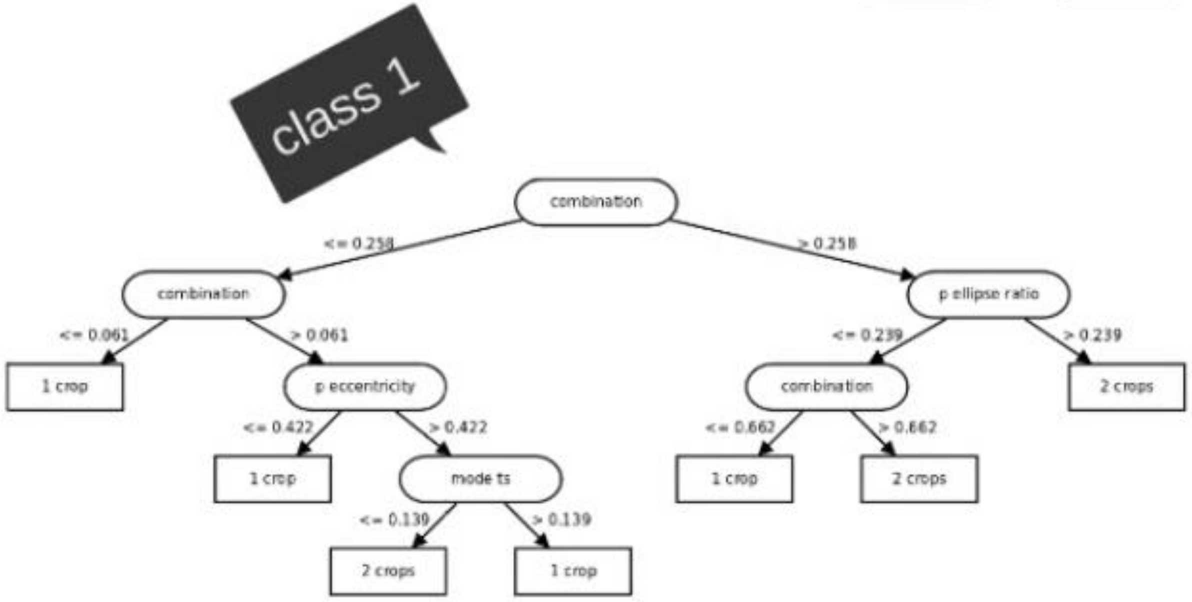
Decision tree M

Class prediction









- A chapter covers too many things