CS 412

FEB 6TH – LOGISTIC REGRESSION / SVM

HTF – CHAPTER 12

HW1 Review

Important takeaways from HW1

- `cross_val_score()`
- Unusual graphs?
- What values of k seemed best?
- Where there any mistakes in the reasoning of the process?

Feature Extraction

Rather than selecting a subset of features, we can procedurally generate new features

- What dimensions in the data are best at explaining?
- They'll often be combinations of features, and as usual, we want to offload the selection process as much as possible.

$$X^d \longrightarrow \boxed{\text{Process}} \longrightarrow X^k$$

PCA

LDA

Principal Components Analysis

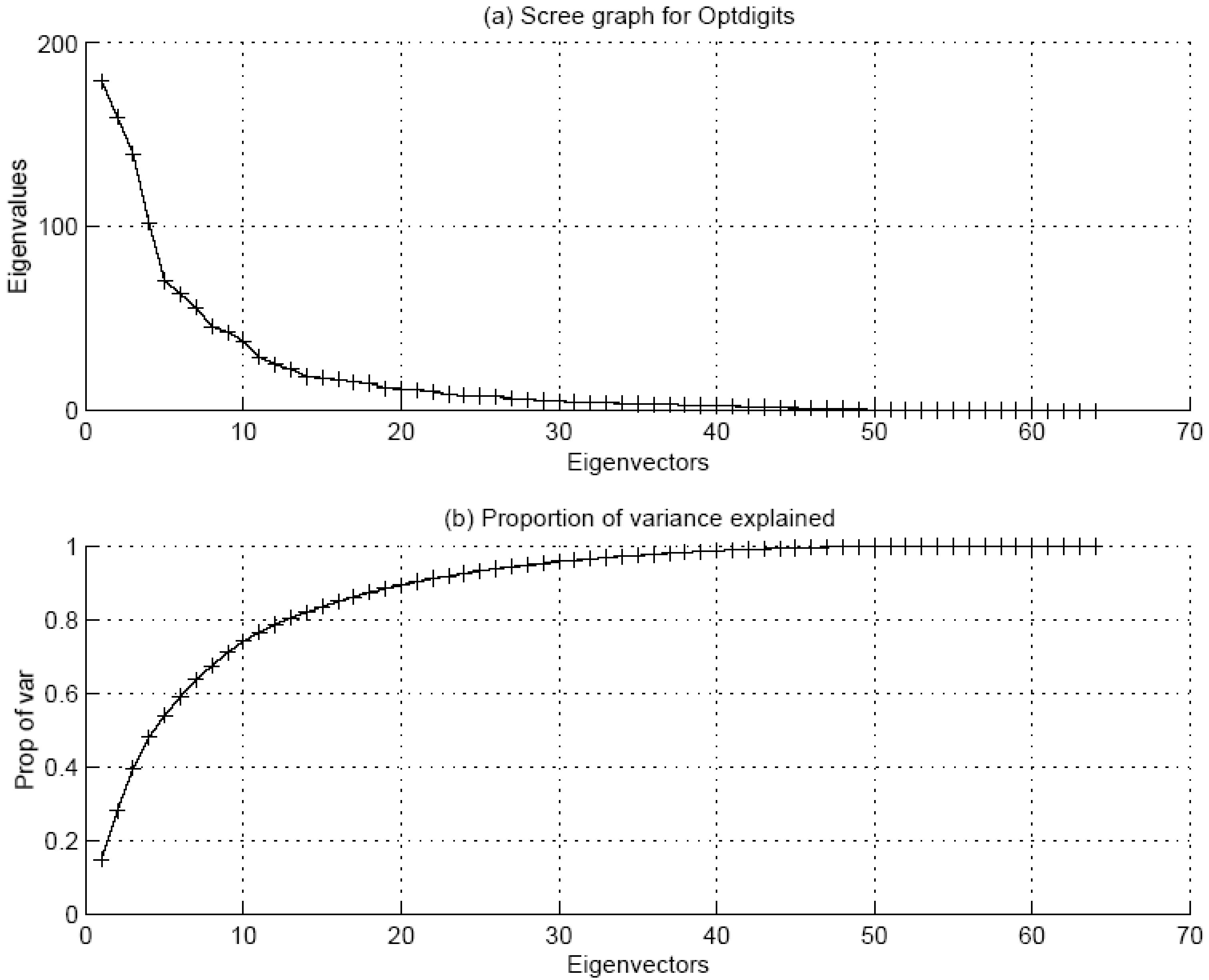
Find a low-dimensional space such that when $\boldsymbol{x}$ is projected there, information loss is minimized.

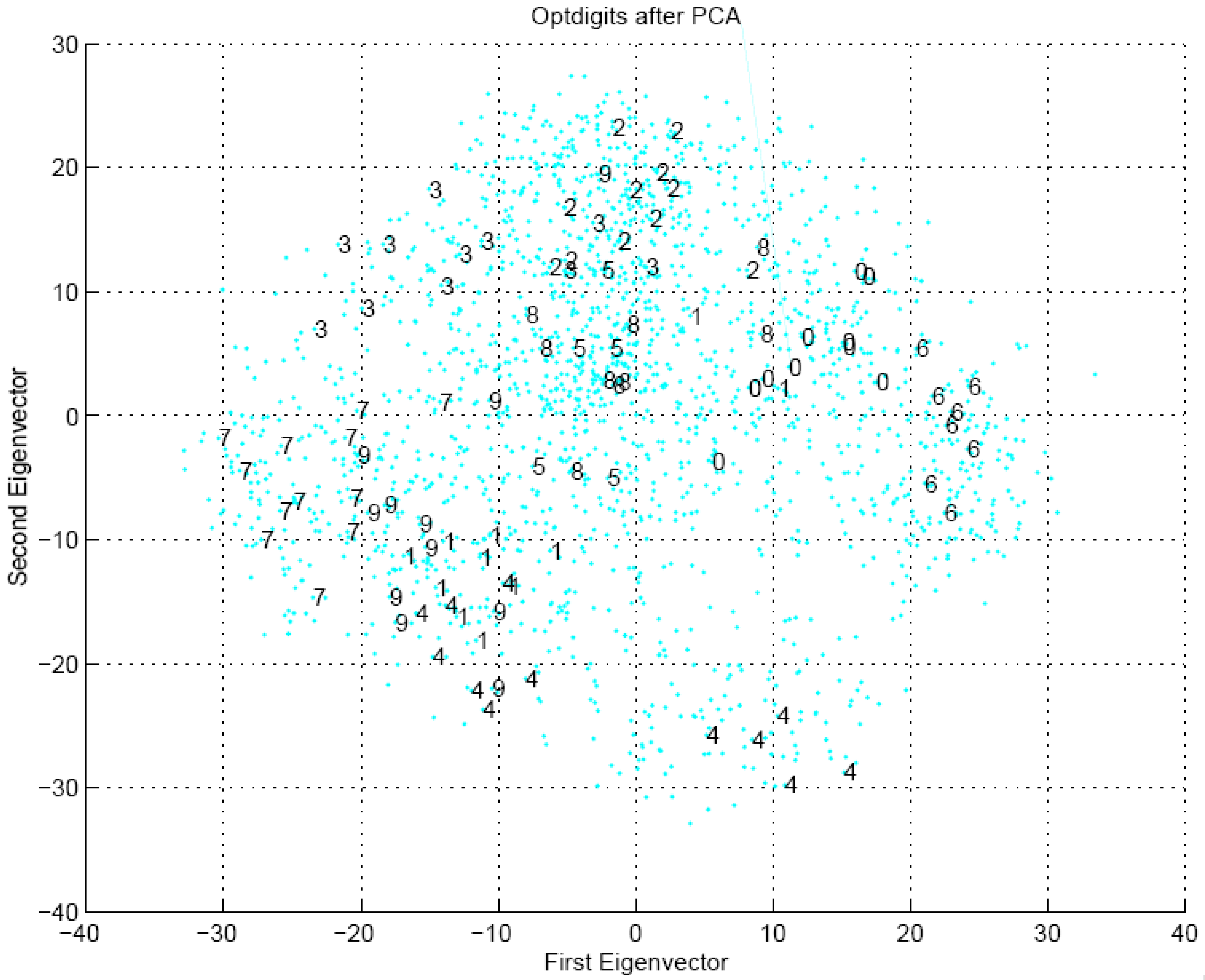
The projection of $\boldsymbol{x}$ on the direction of $\boldsymbol{w}$ is: $z = \boldsymbol{w}^T\boldsymbol{x}$

Find $\boldsymbol{w}$ such that **Var(z) is maximized**

$$\begin{aligned}\mathrm{Var}(z) = \mathrm{Var}(\boldsymbol{w}^T\boldsymbol{x}) &= \mathrm{E}[(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})^2] \\ &= \mathrm{E}[(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{\mu})] \\ &= \mathrm{E}[\boldsymbol{w}^T(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{w}] \\ &= \boldsymbol{w}^{\mathrm{T}}\,\mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T]\boldsymbol{w} = \boldsymbol{w}^T\textstyle\sum\boldsymbol{w}\end{aligned}$$

where $\mathrm{Var}(\boldsymbol{x}) = \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T] = \sum$


(a) Scree graph for Optdigits
Eigenvalues
200
100
0
0
10
20
30
40
50
60
70
Eigenvectors
(b) Proportion of variance explained
Prop of var
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60
70
Eigenvectors


Optdigits after PCA
Second Eigenvector
First Eigenvector

Linear Discriminant Analysis

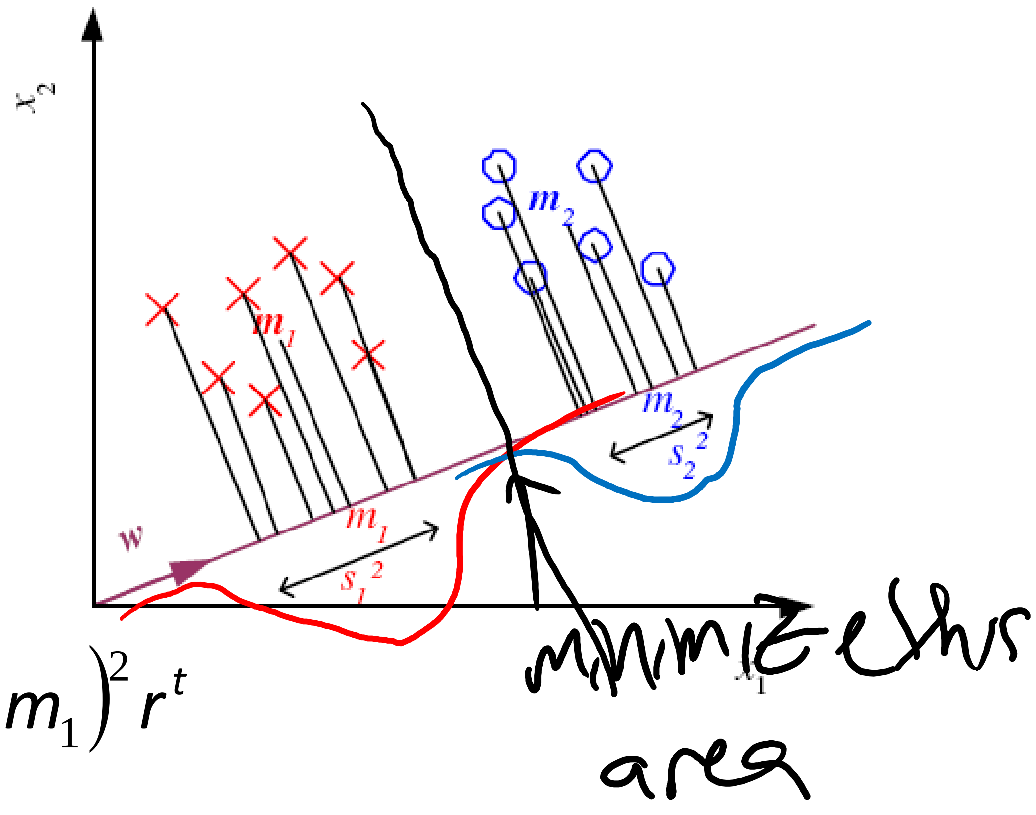
Find a low-dimensional space such that when $\boldsymbol{x}$ is projected, classes are well-separated.

Find $\boldsymbol{w}$ that maximizes the separation

Linear Discriminant Analysis

Find a low-dimensional space such that when is projected, classes are well-separated.

Find w that maximizes the separation



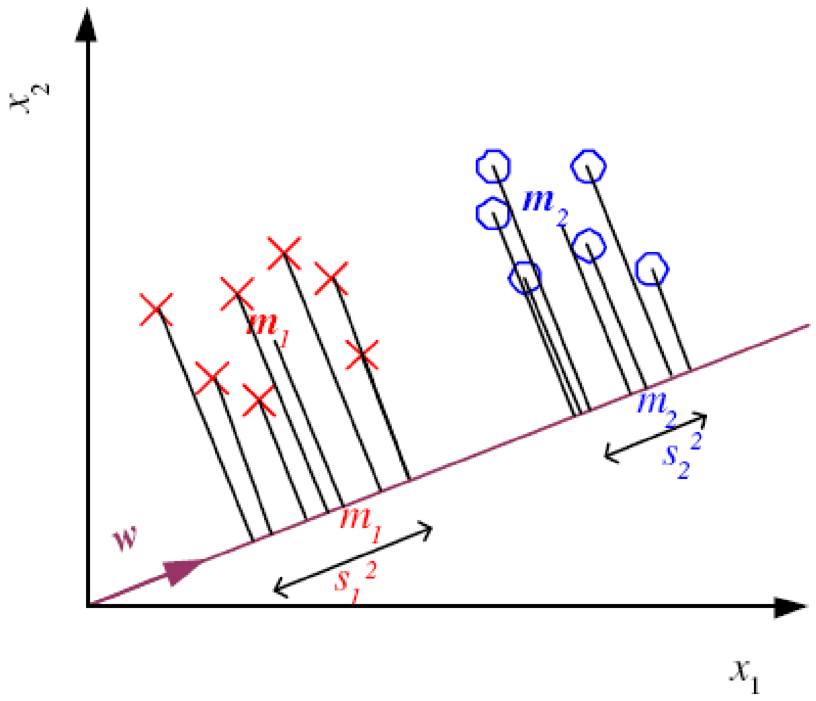
$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

$$m_1 = \frac{\sum_t \mathbf{w}^T \mathbf{x}^t r^t}{\sum_t r^t} \qquad s_1^2 = \sum_t \left(\mathbf{w}^T \mathbf{x}^t - m_1\right)^2 r^t$$

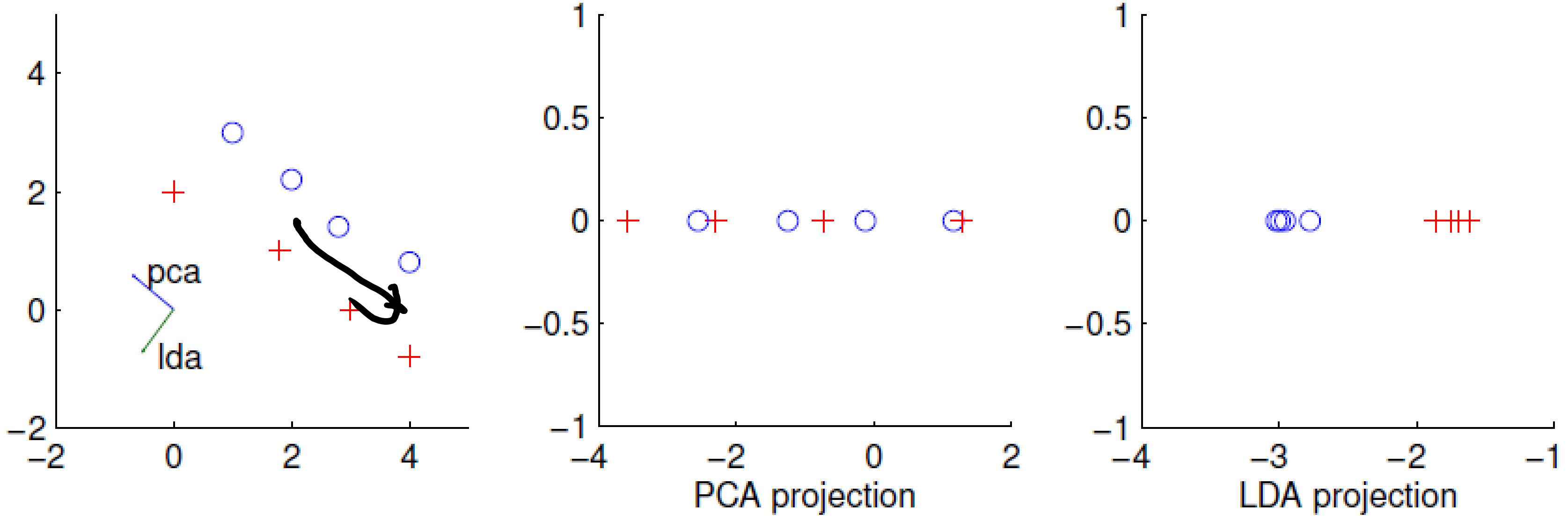
Linear Discriminant Analysis

Find a low-dimensional space such that when is projected, classes are well-separated.

Find $\boldsymbol{w}$ that maximizes the separation



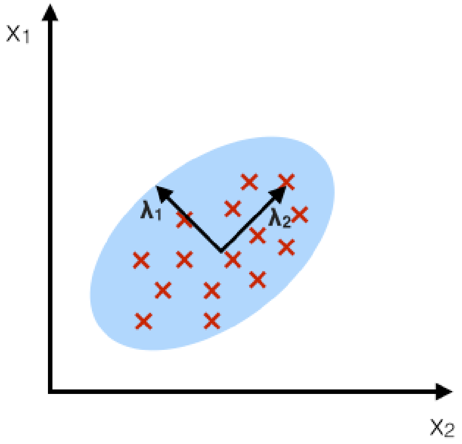
PCA vs LDA



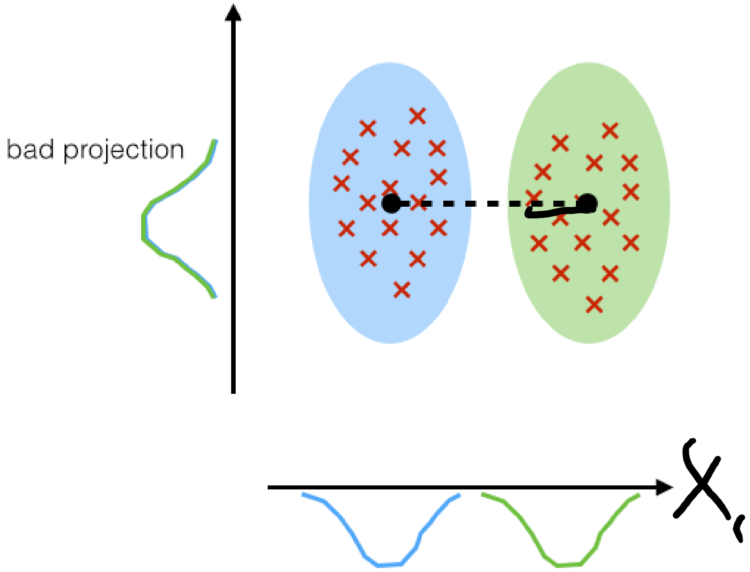
4
2
0
−2
−2
0
2
4
pca
lda
1
0.5
0
−0.5
−1
−4
−2
0
2
PCA projection
1
0.5
0
−0.5
−1
−4
−3
−2
−1
LDA projection

PCA vs LDA

PCA:
component axes that maximize the variance



LDA:
maximizing the component axes for class-separation



PCA and LDA are not limited to the axes

PCA vs LDA

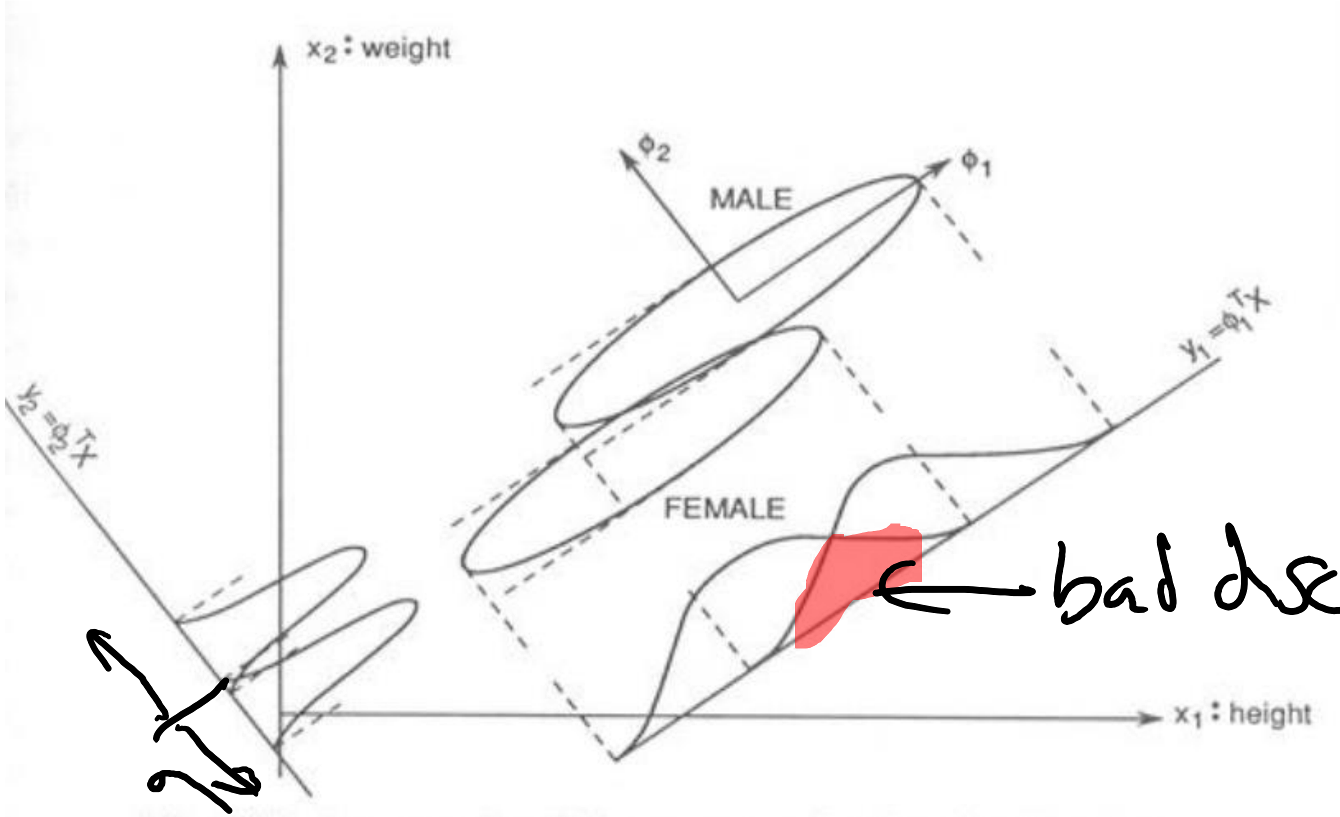


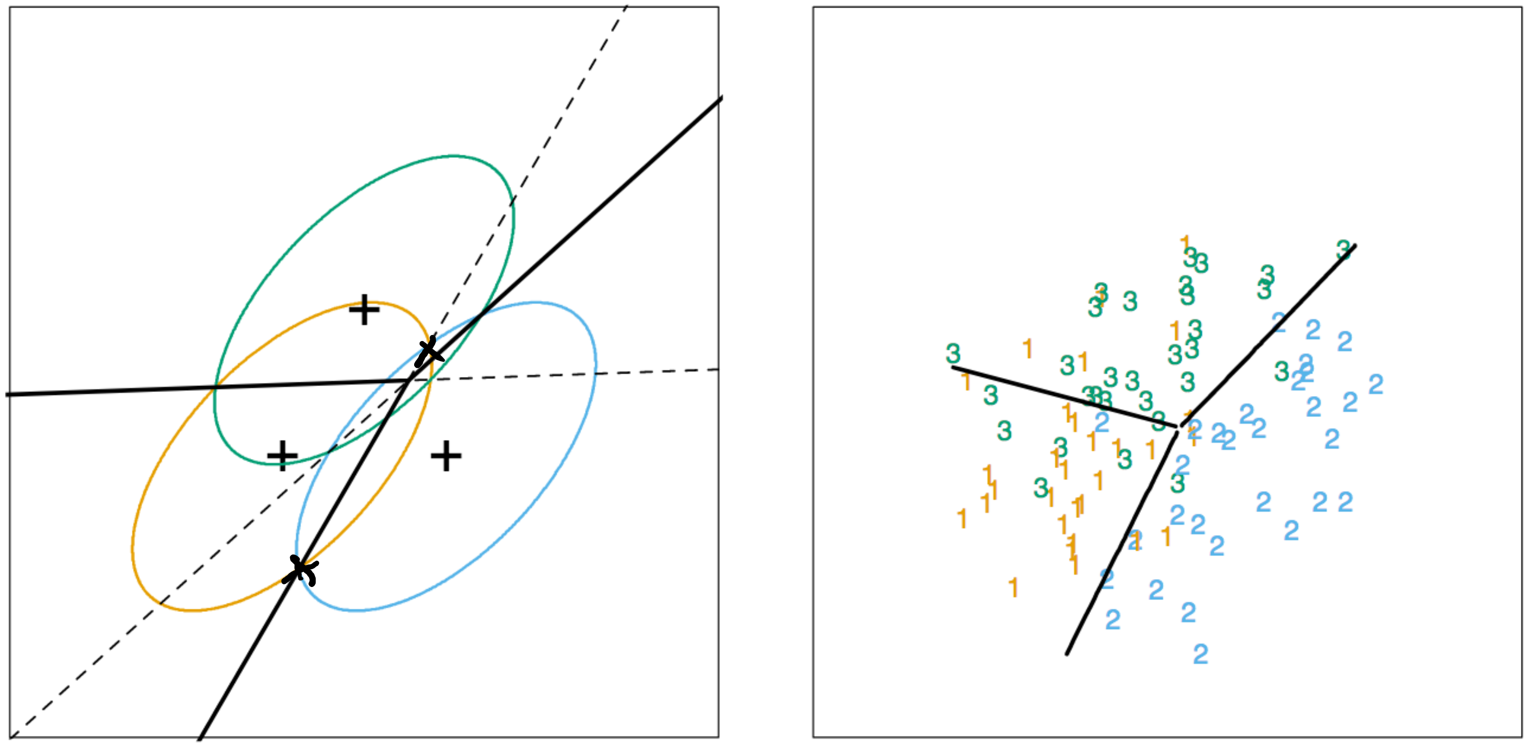
Fig. 10-1 An example of feature extraction for classification.

PCA

bad discriminator

low variance

identically distributed



Discriminate analysis

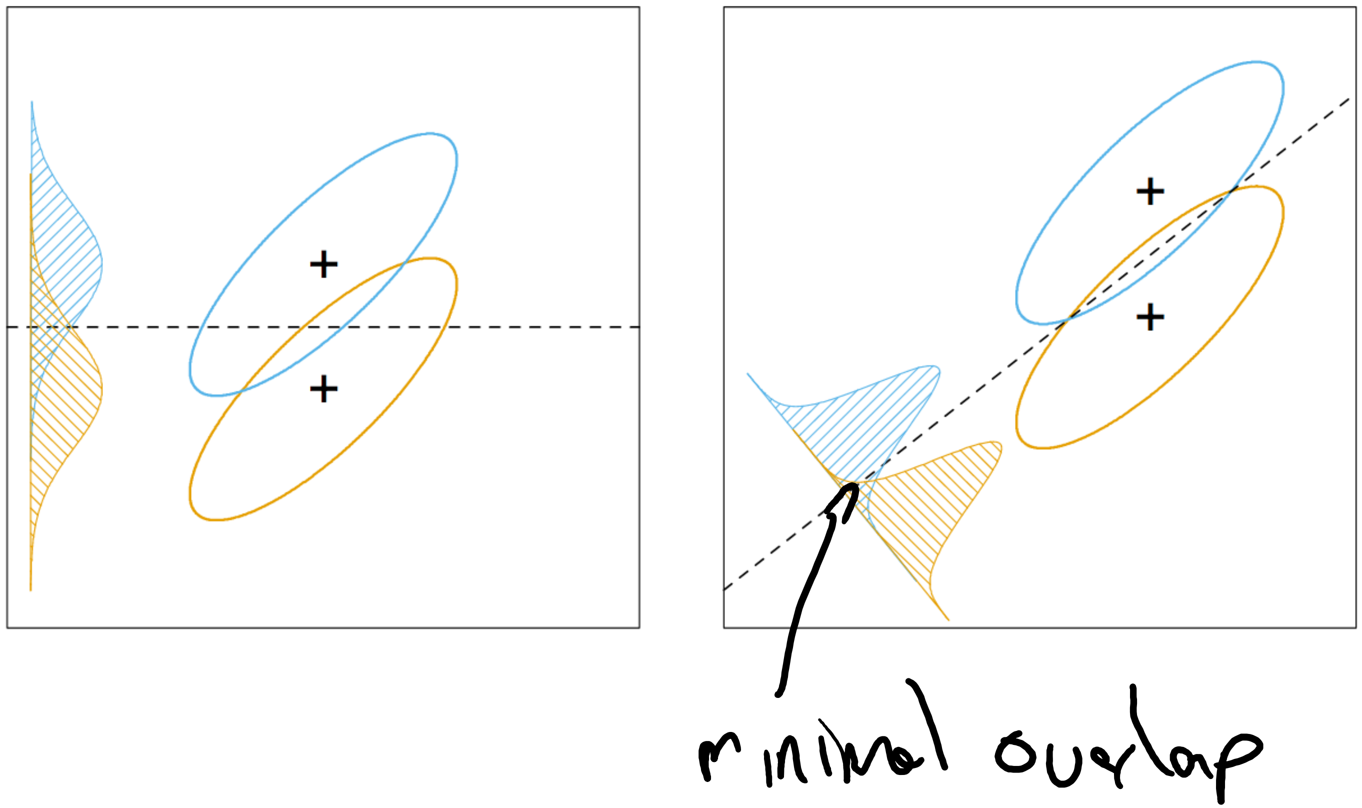
Suppose that the data is randomly distributed in $\mathbb{R}^p$ relative to some multivariate Gaussian distribution.

- Find the center of each distribution region and then separate the regions based on what minimizes the probability of overlap
- Assume that each of the regions shares the same distribution

Discriminate analysis

To minimize overlap, we want to discriminate between the classes along the axis that best discriminates the data

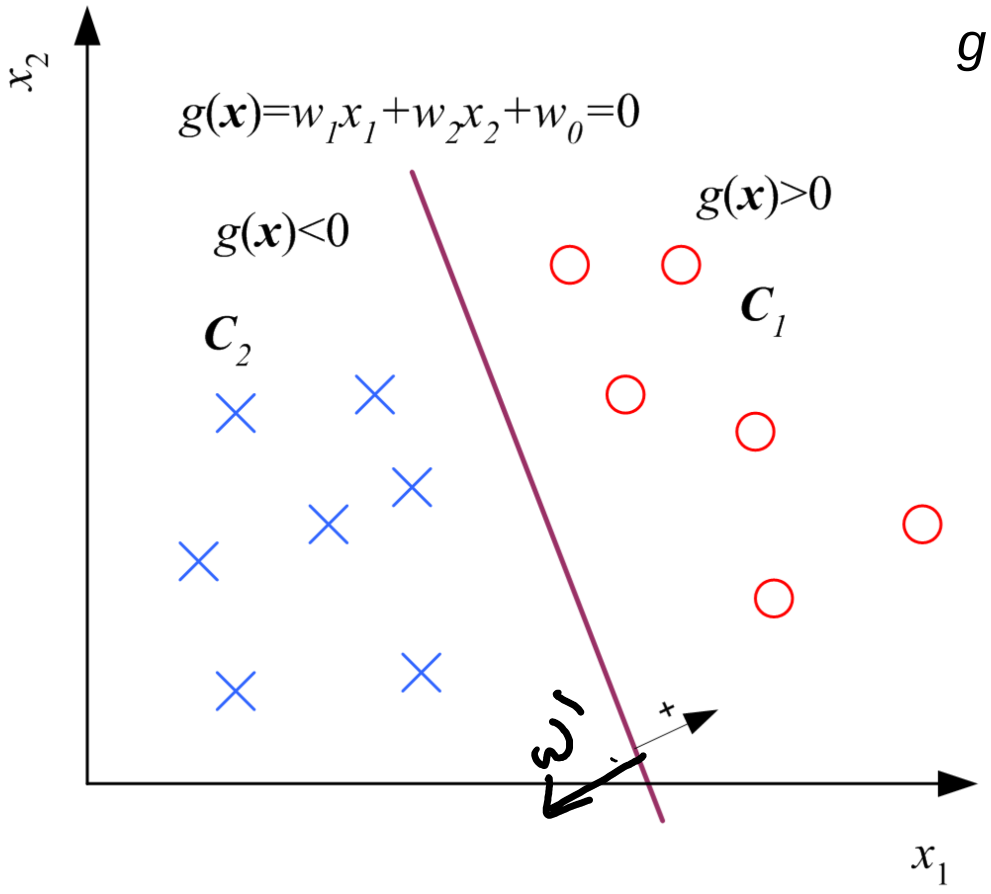
Unlike linear regression we want to divide the region into relevant spaces



Polynomial Kernel $\varphi(x)$

$X_1, X_2, X_1^2, X_2^2, X_1X_2$

Two Classes



g

$$\text{choose } \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}$$

$C_2 \quad g'(x) > 0$

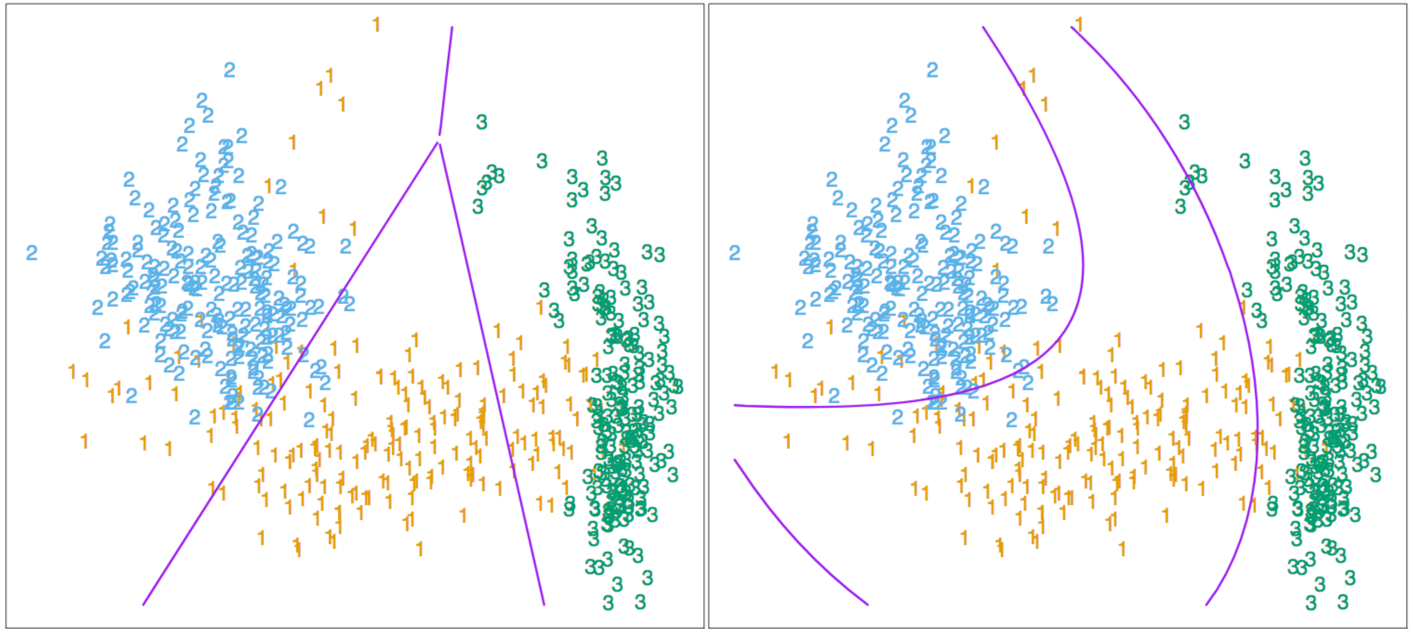
C_1 otherwise

Quadratic discriminants

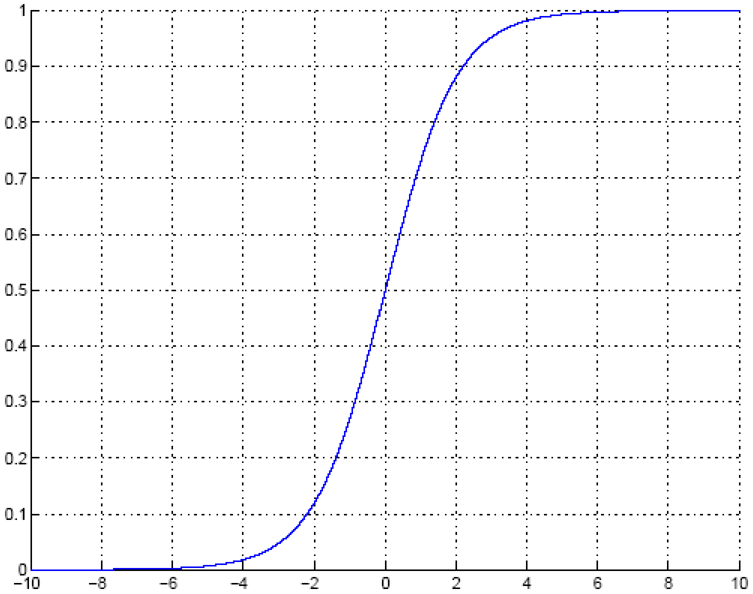
Here, we can see two different models

- X_1, X_2
- $X_1, X_2, X_1X_2, X_1^2, X_2^2$

For only two variables, it is easy to generate the interaction/polynomial terms



Sigmoid (Logistic) Function

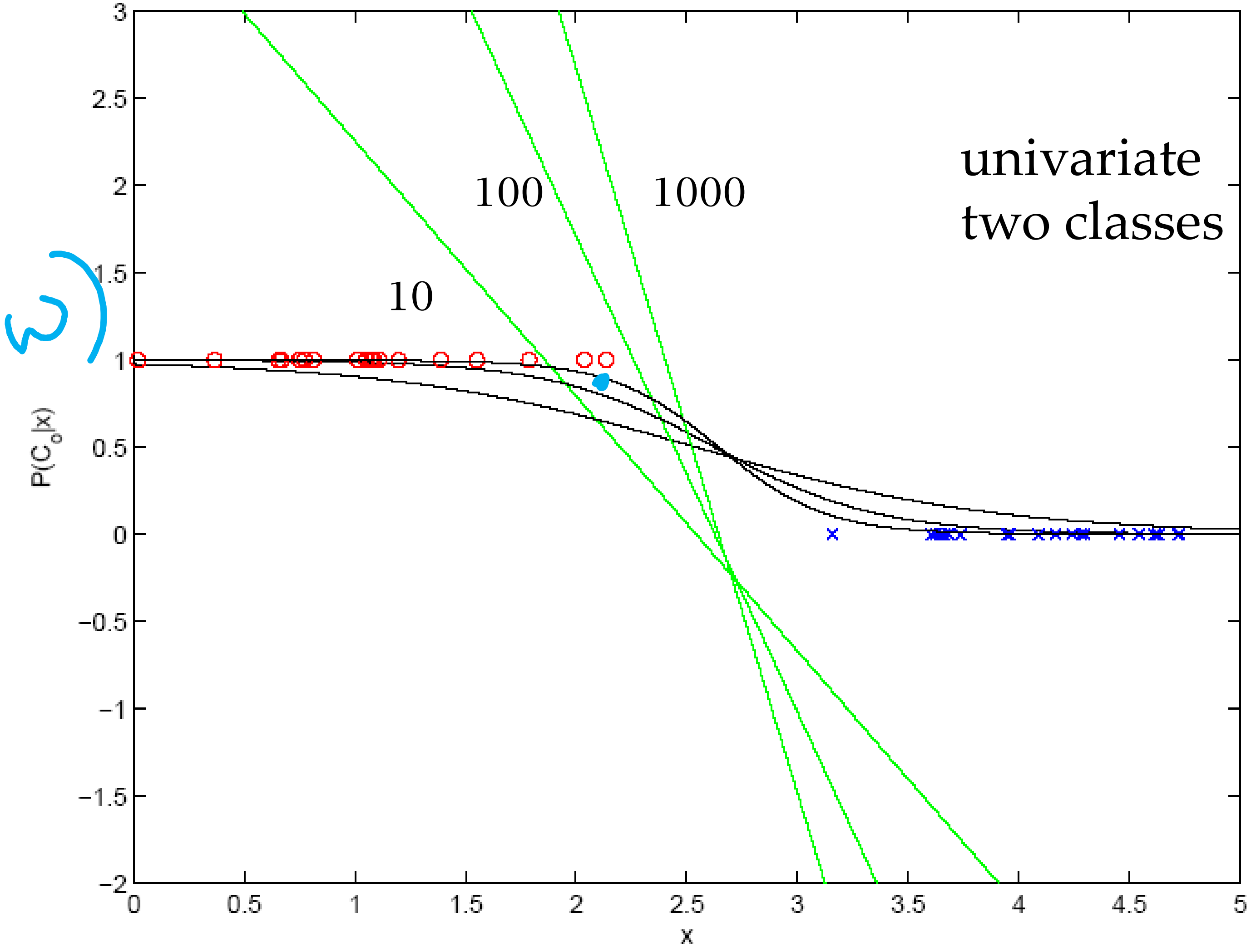


Calculate $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ and choose C_1 if $g(\mathbf{x}) > 0$, or

Calculate $y = \text{sigmoid}\left(\mathbf{w}^T\mathbf{x} + w_0\right)$ and choose C_1 if $y > 0.5$

$$= \text{sigmoid}(a), \text{where } a = \mathbf{w}^T\mathbf{x} + w_0 \qquad \frac{dy}{da} = y(1-y)$$

after 10, 100, 1000 iterations



Sigmoid($x^T w$)

=

0.9

g(x) =

Sigmoid($x^T w$)

g(x) > 0.5

red 'o'

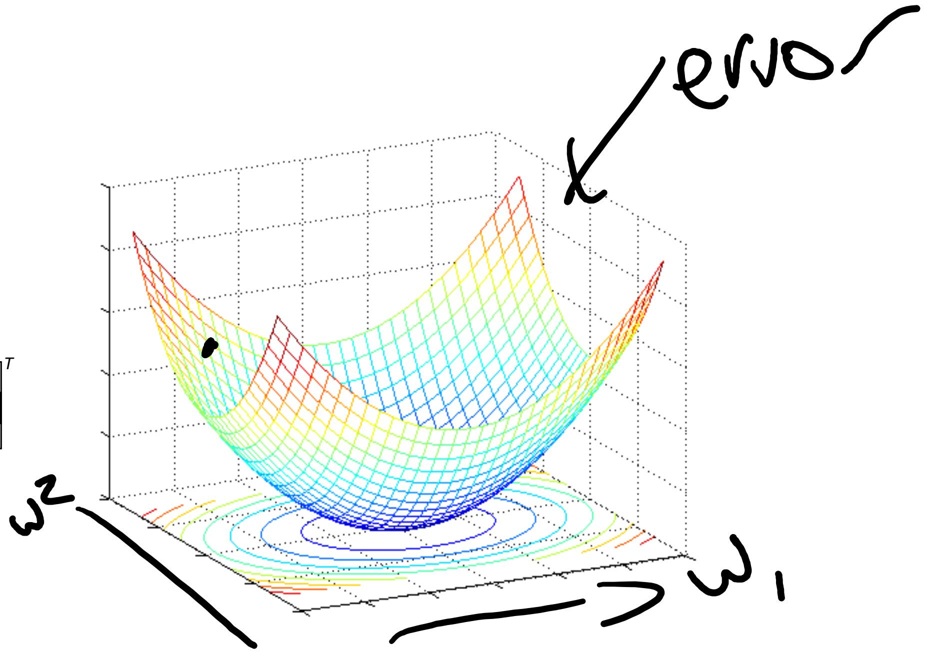
Gradient-Descent

E(w|X) is error with parameters w on sample X

$$w^* = \arg\min_w E(w \mid X)$$

Gradient

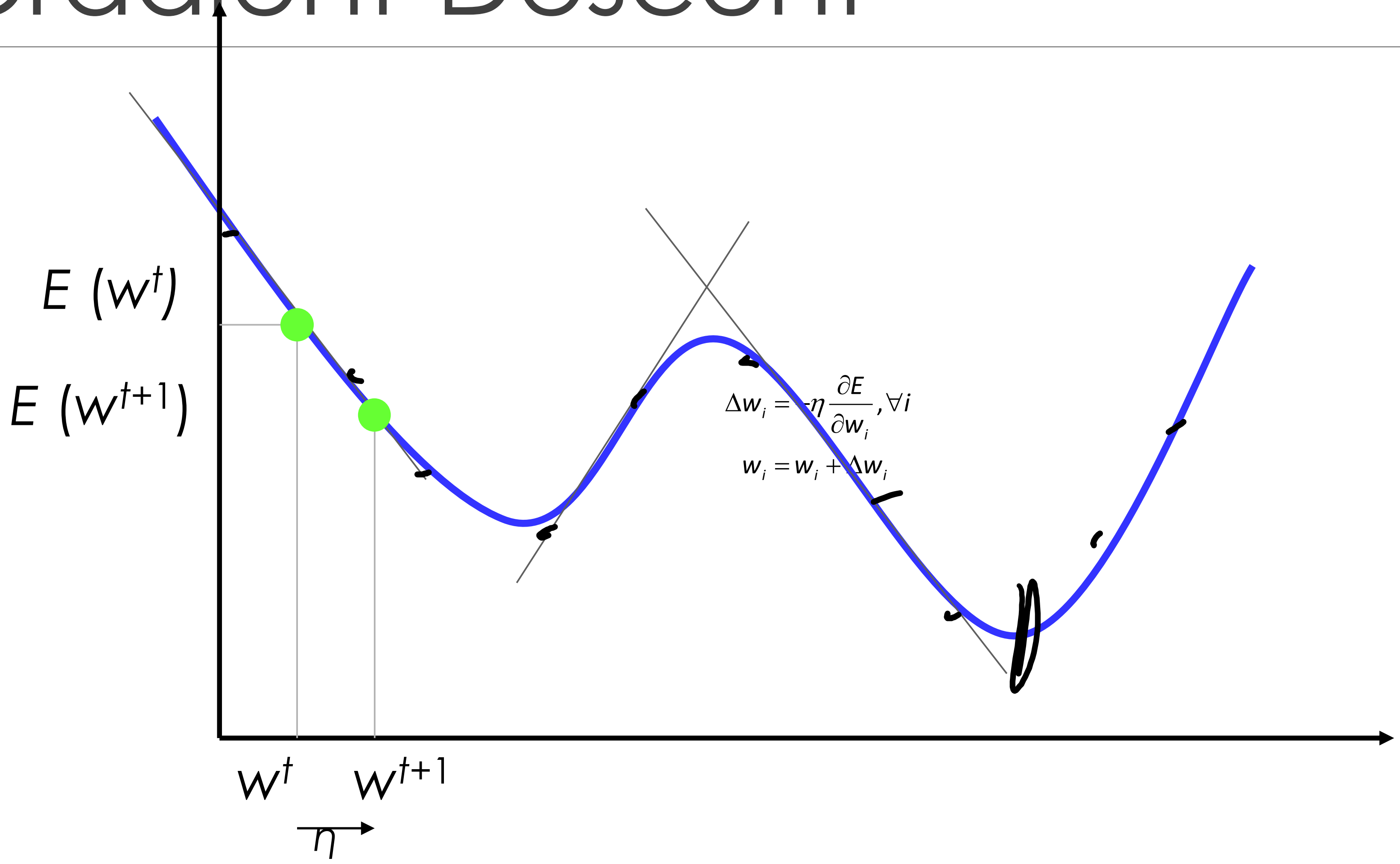
$$\nabla_w E = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_d}\right]^T$$



Gradient-descent:

Starts from random w and updates w iteratively in the negative direction of gradient

Gradient-Descent

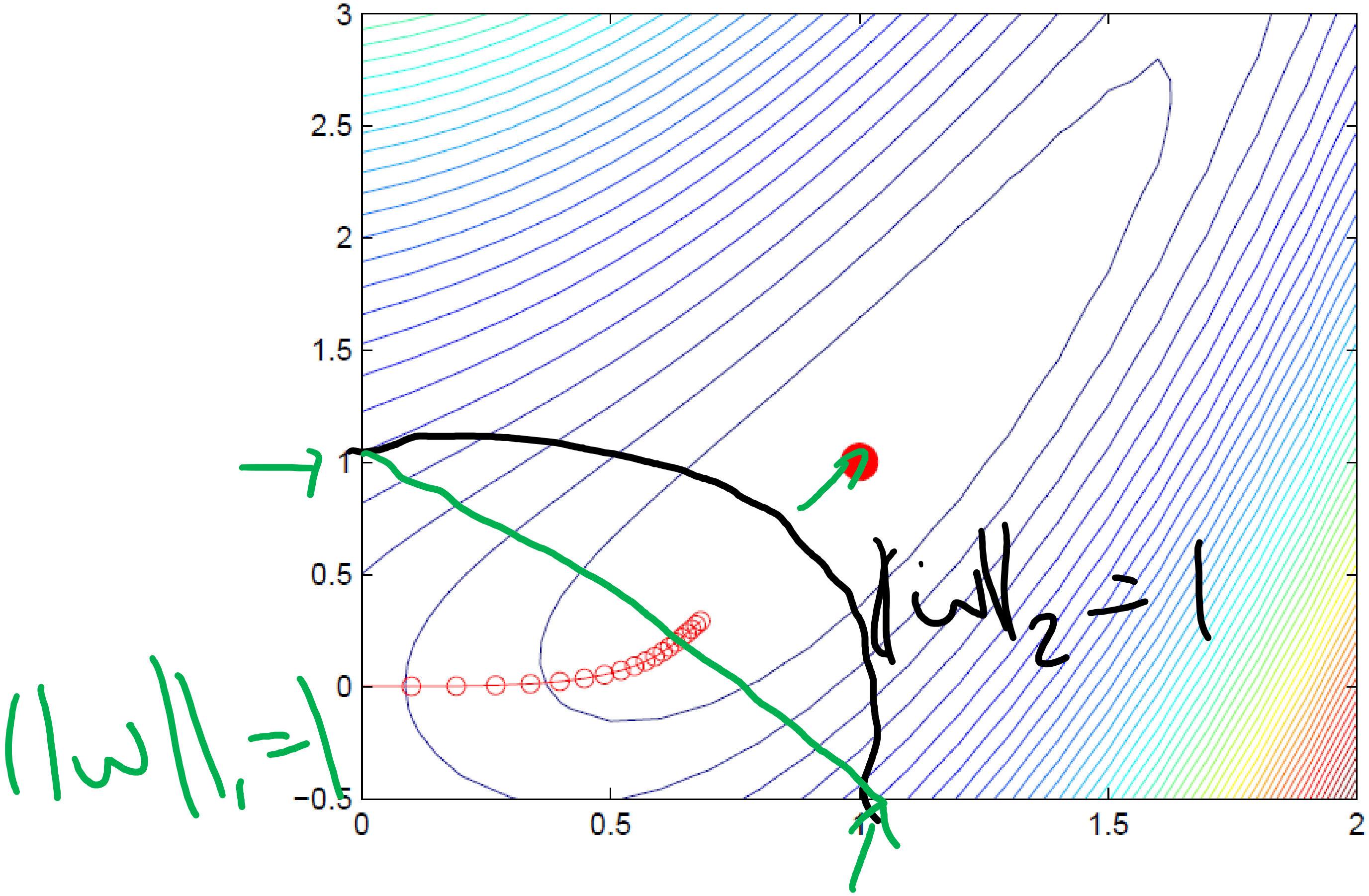


Learning for logistic regression

$\eta = 0.1$

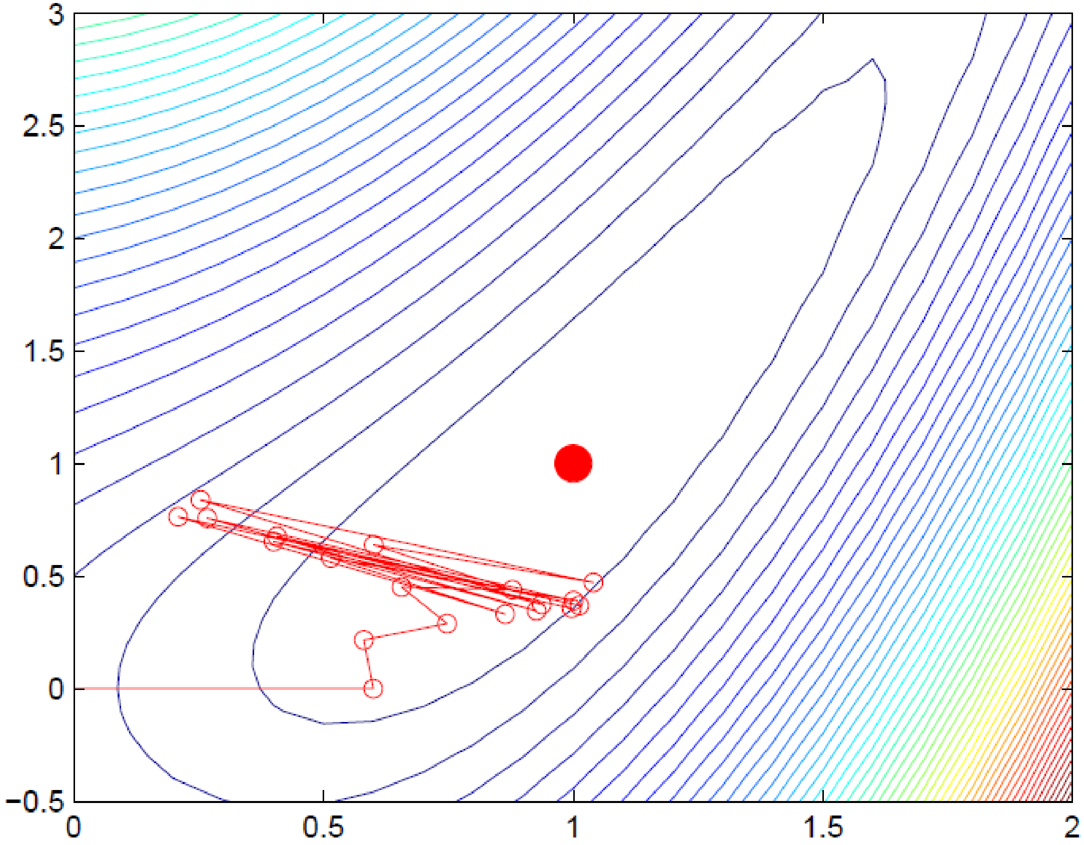
η = learning rate

(e|a)



Learning for logistic regression

$\eta = 0.6$



Logistic regression and overfitting

Overfitting

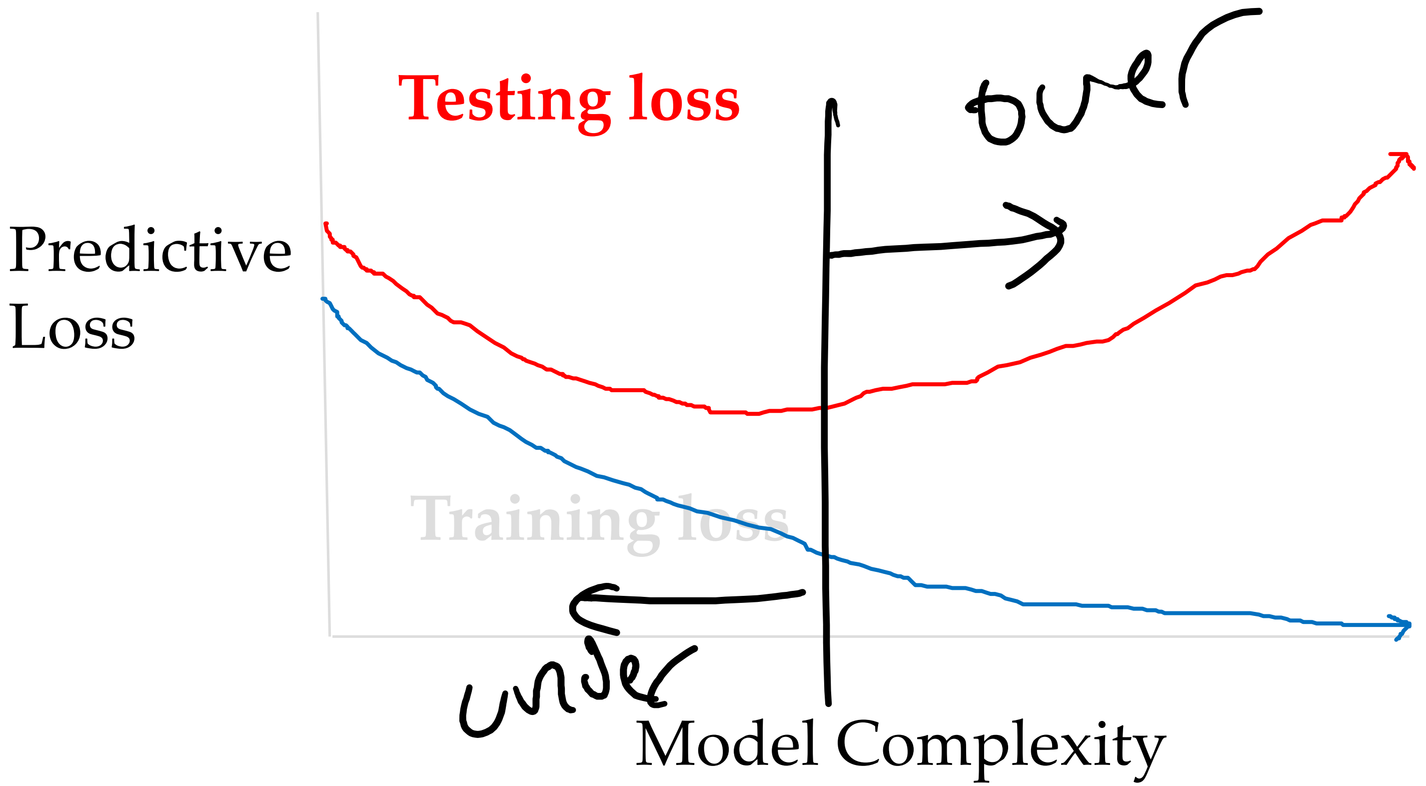
- Occurs when very few instances and feature space is high dimensional

datapoints

To avoid, a common approach is defining a prior on w

- Corresponds to Regularization
- Helps with avoiding large weights
- “Pushes” parameters to zero

Overfitting



Need to prevent complex hypotheses

Overfitting
- Occurs when very few instances and feature space is high dimensional

Idea #1: Restrict the number of features considered
- Cross-validation

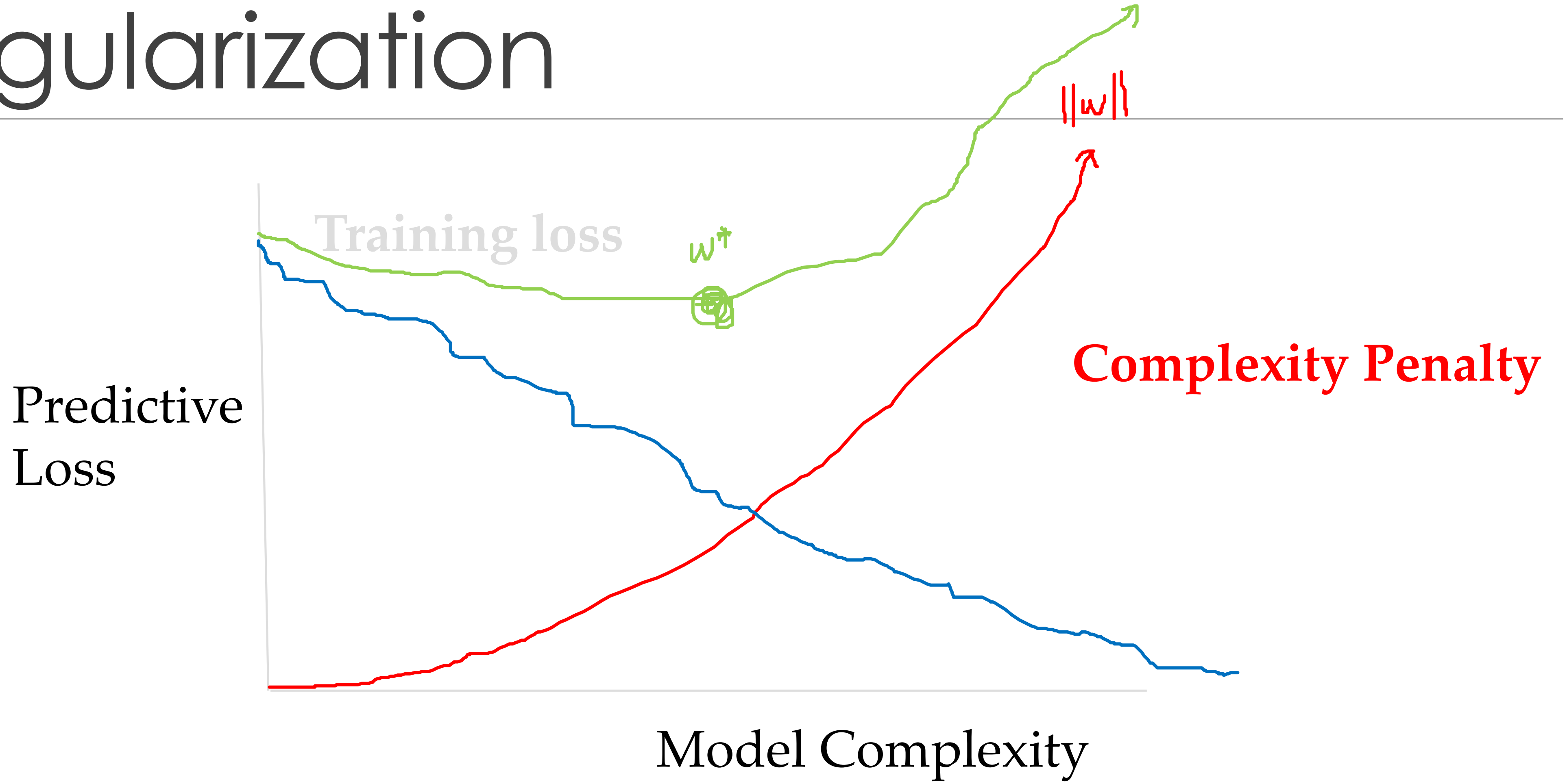
Idea #2: Penalize complex hypotheses in the model search
- Regularization!

$$g(x) = 1000x_1 - 1500x_2 + 100$$

$$h(x) = 10x_1 - 15x_2 + 1$$

$$= 0.001x_1 - 0.0015x_2 + 0.0001$$

Regularization



Regularization

Sigmoid function

Recall the objective of logistic regression:

$$E(\mathbf{w}, w_0 \mid \mathcal{X}) = -\sum_t r^t \log y^t + \left(1 - r^t\right) \log\left(1 - y^t\right)$$

L2 regularization

λ = regularization parameter

$$\underset{\mathbf{w}}{argmin} \quad E(\mathbf{w}, w_0 | X) + \lambda \sum_i w_i^2$$

L1 regularization

vector

$$\underset{\mathbf{w}}{argmin} \quad E(\mathbf{w}, w_0 | X) + \lambda \sum_i |w_i|$$

$\lambda > 0$ is a weight, chosen by, e.g., cross validation