

Supervised learning: linear regression and SVM

Shikui Tu

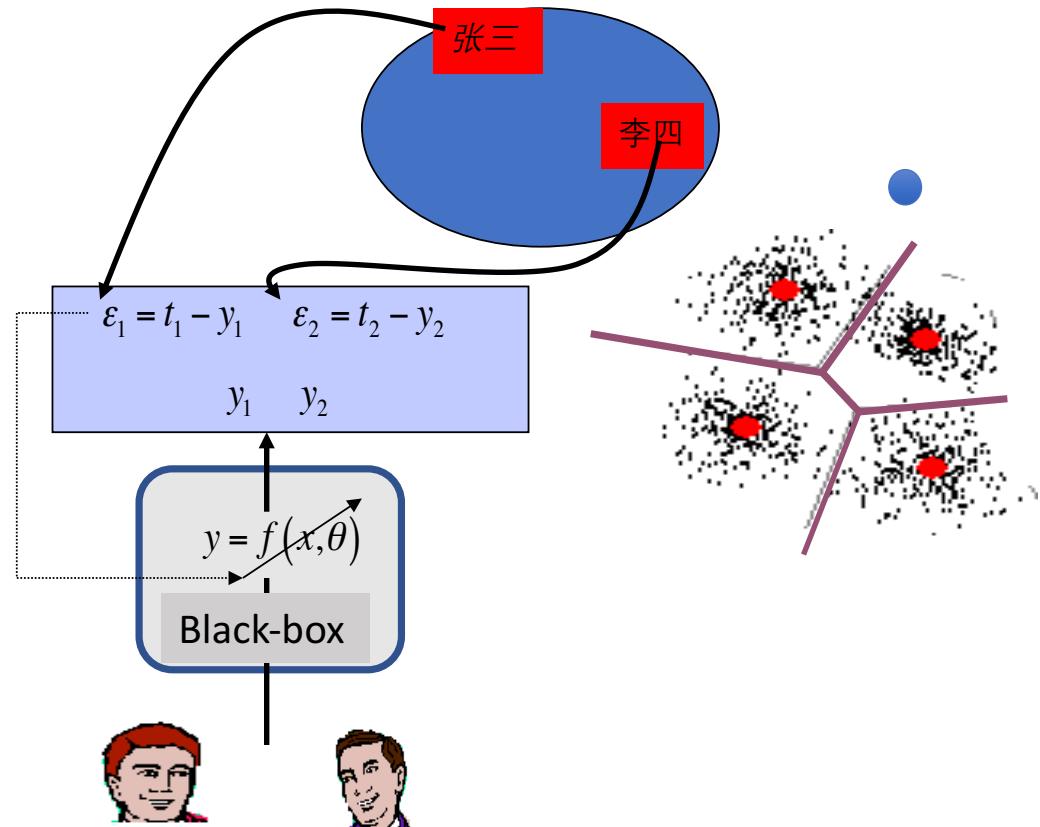
Department of Computer Science and
Engineering, Shanghai Jiao Tong University

2019-04-09

Outline

- **Supervised learning**
- Linear regression
- Logistic regression
- Support Vector Machine (SVM)

Supervised learning



Models and algorithms

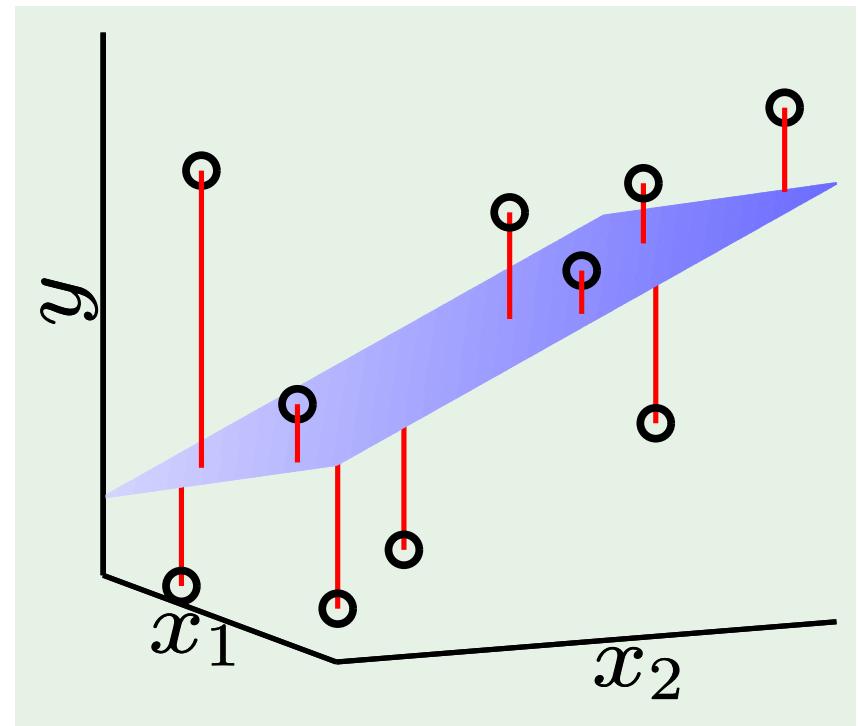
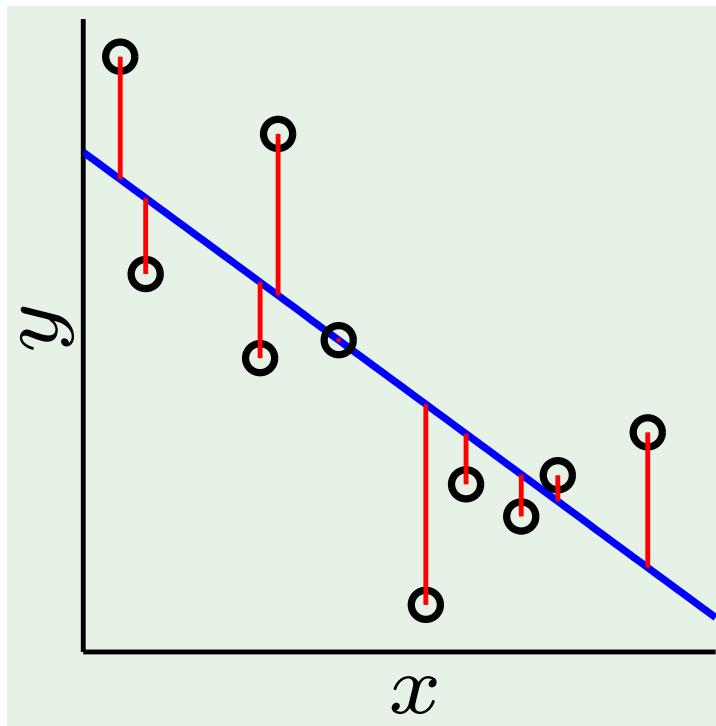
- linear regression
- logistic regression
- Support Vector Machines
- naive Bayes
- linear discriminant analysis
- decision trees
- k-nearest neighbor algorithm
- Neural Networks (Multilayer perceptron)
- ...

Outline

- Supervised learning
- **Linear regression**
- Logistic regression
- Support Vector Machine (SVM)

Linear regression

“Regression” usually means “real-value output”.



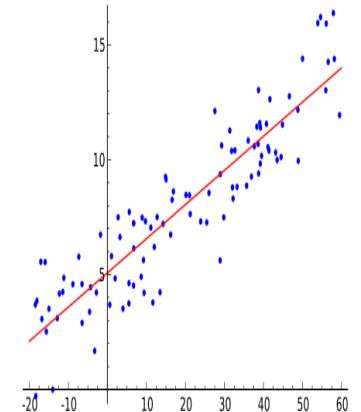
Basic concepts of linear regression

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

the *regressand, endogenous variable, response variable, measured variable, criterion variable, or dependent variable*

$$y_i = \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i = \mathbf{x}_i^T \boldsymbol{\beta} + \varepsilon_i, \quad i = 1, \dots, n,$$

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ x_{21} & \cdots & x_{2p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$



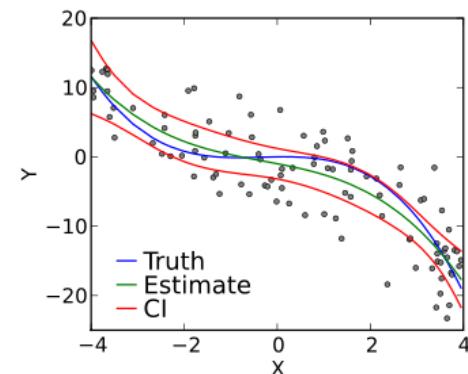
regressors, exogenous variables, explanatory variables, covariates, input variables, predictor variables, or independent variables

The matrix \mathbf{X} is sometimes called the [design matrix](#).

$$h_i = \beta_1 t_i + \beta_2 t_i^2 + \varepsilon_i,$$

$$\mathbf{x}_i = (x_{i1}, x_{i2}) = (t_i, t_i^2)$$

$$h_i = \mathbf{x}_i^T \boldsymbol{\beta} + \varepsilon_i.$$



Least Square Estimation

Matrix form: $Y = X\beta + \epsilon$

Minimize the sum of squared residuals (errors):

$$\min_{\beta} \varepsilon^T \varepsilon = (Y - X\beta)^T (Y - X\beta) \quad \longleftrightarrow \quad \min_w \sum_t \|e_t(x_t, w)\|^2$$

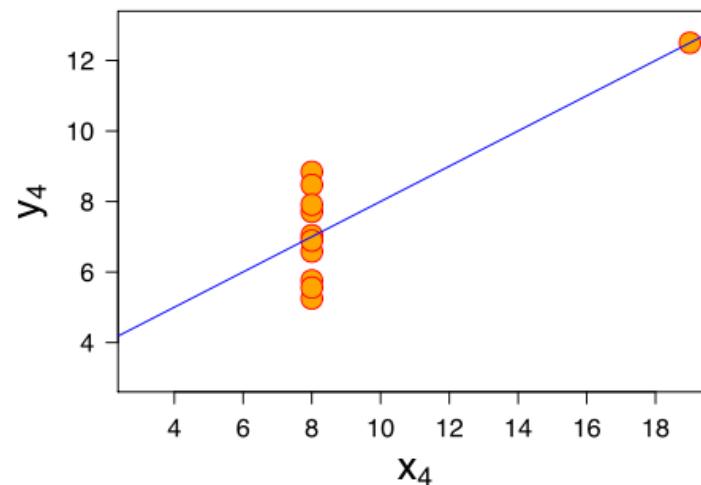
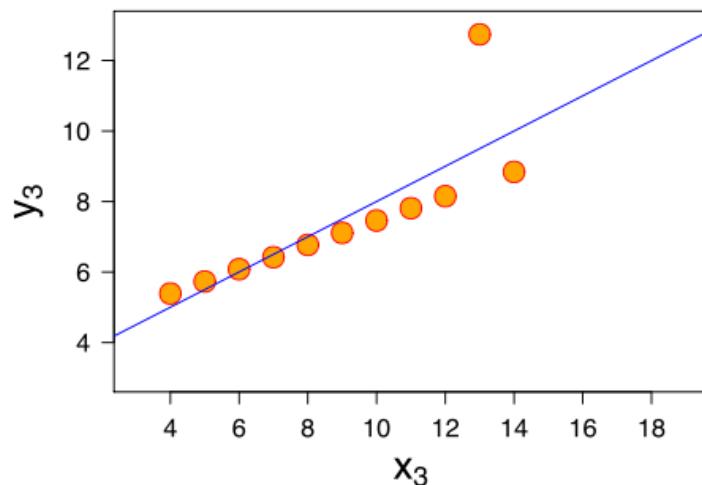
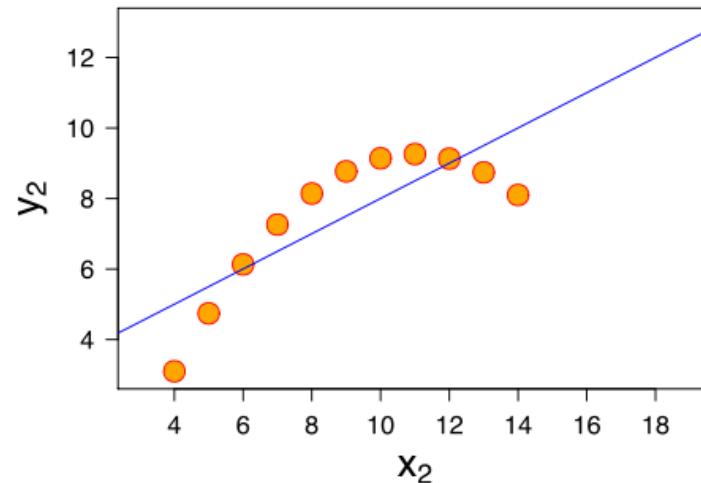
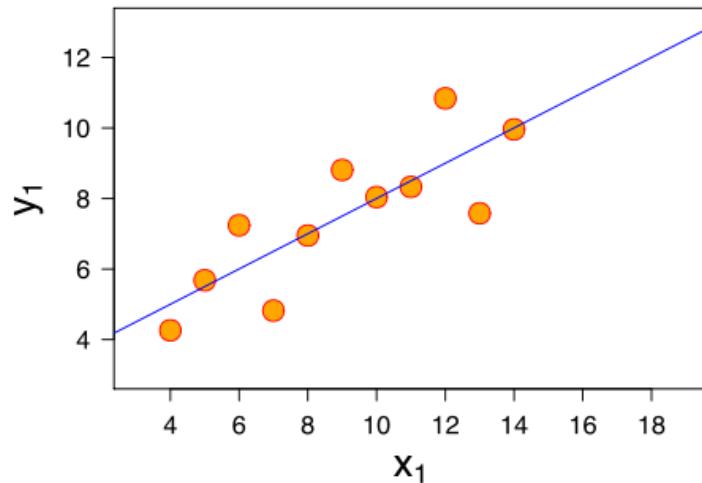
By taking the derivative to zero, we have:

$$\beta = (X^T X)^{-1} X^T Y$$

The pseudo-inverse $X^+ = (X^T X)^{-1} X^T$

A diagram illustrating the dimensions of the matrices in the equation $X^+ = (X^T X)^{-1} X^T$. The matrix $(X^T X)^{-1}$ is shown as a green square bracket with a dimension of $d+1 \times d+1$ indicated by blue brackets below it. To its right is a green bracket with a dimension of $d+1 \times N$ indicated by blue brackets below it. Below the first green bracket is another green bracket with a dimension of $d+1 \times N$ indicated by blue brackets below it.

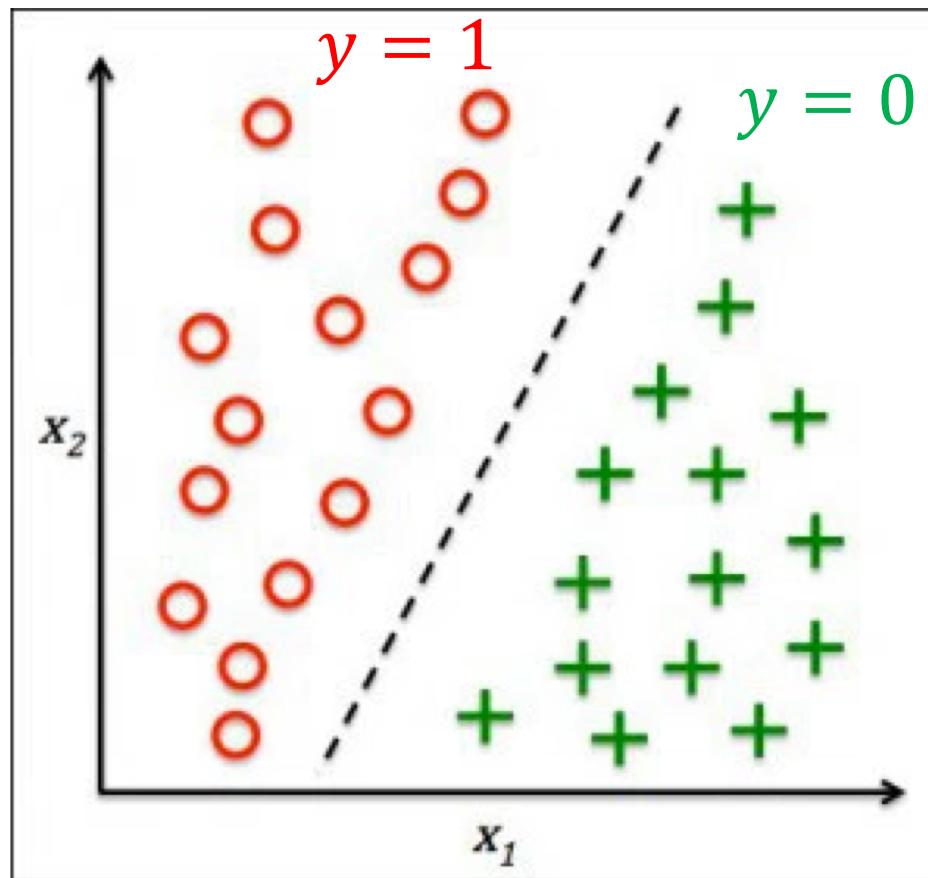
Limitations of linear regression



Outline

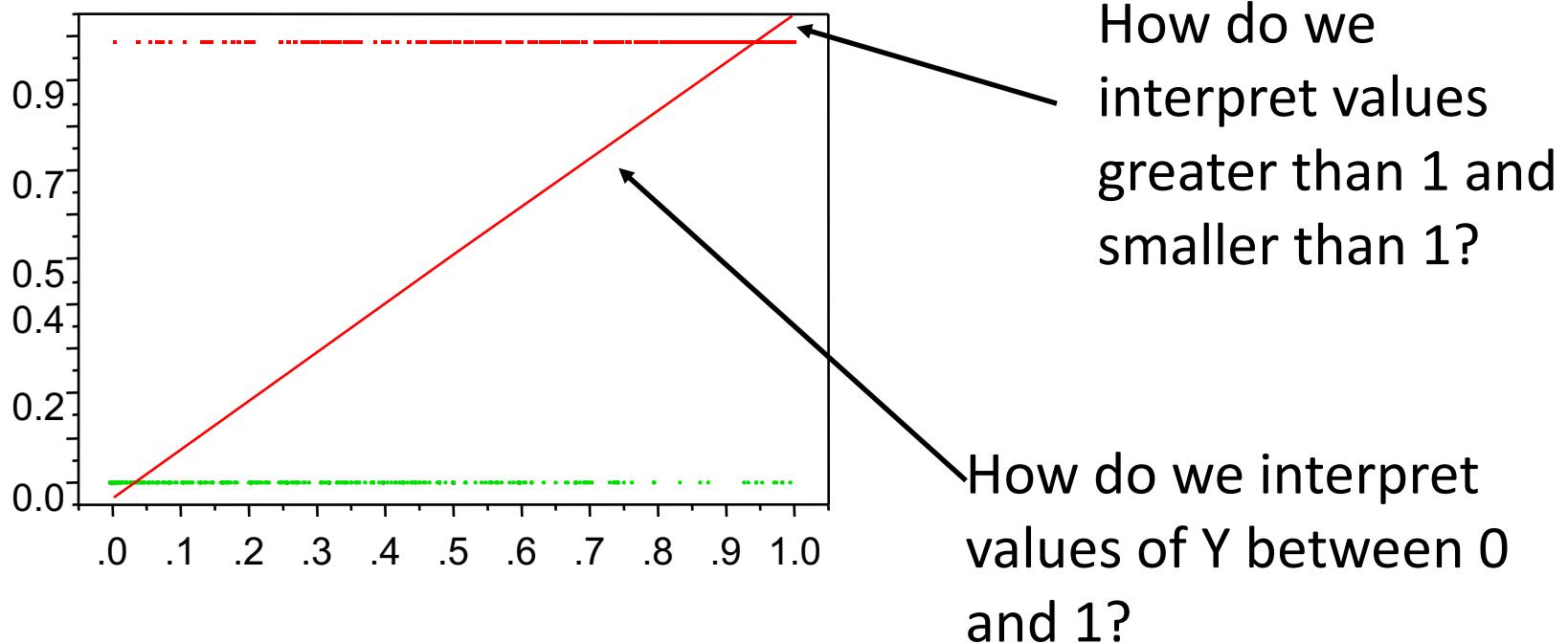
- Supervised learning
- Linear regression
- **Logistic regression**
- Support Vector Machine (SVM)

Classification



Why not Linear Regression?

- For classification, Y only takes on values of 0 and 1



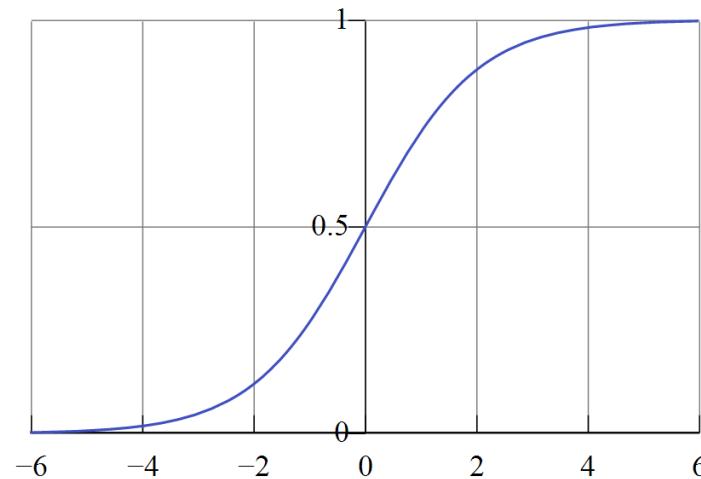
Problems

- The regression line $\beta_0 + \beta_1 X$ can take on any value between negative infinity and positive infinity.
- In the classification problem, output Y can only take on two possible values: 0 or 1.
- Therefore the regression line almost always predicts the wrong value for output Y in classification problems

Solution: Use Logistic Function

- Instead of trying to predict Y , let's try to predict $P(Y = 1)$
- Model $P(Y = 1)$ using a function that gives outputs between 0 and 1 → logistic function!

$$\text{Sigmoid}(x) = \text{Logistic}(x) = \frac{1}{1 + e^{-x}}$$



Logistic Regression

$$p(x) = P(Y = 1|x) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

- Euler's number $e = 2.71828\dots$
- no matter what values 0, 1 or X take, $p(X)$ will have values between 0 and 1

$$\log\left(\frac{p(x)}{1 - p(x)}\right) = \beta_0 + \beta_1 X$$

Log-odds or logit transformation of $p(X)$ gives a linear model.

Interpreting β_1

- Not very easy with logistic regression → we are predicting $P(Y)$, not Y directly
 - If $\beta_1 = 0 \rightarrow$ no relationship between Y and X
 - If $\beta_1 > 0 \rightarrow$ when X gets larger so does the probability that $Y = 1$
 - If $\beta_1 < 0 \rightarrow$ when X gets larger, the probability that $Y = 1$ gets smaller
 - How much bigger or smaller depends on where we are on the slope

Logistic Regression with Multiple Variables

$$p(X) = \frac{e^{\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p}}.$$

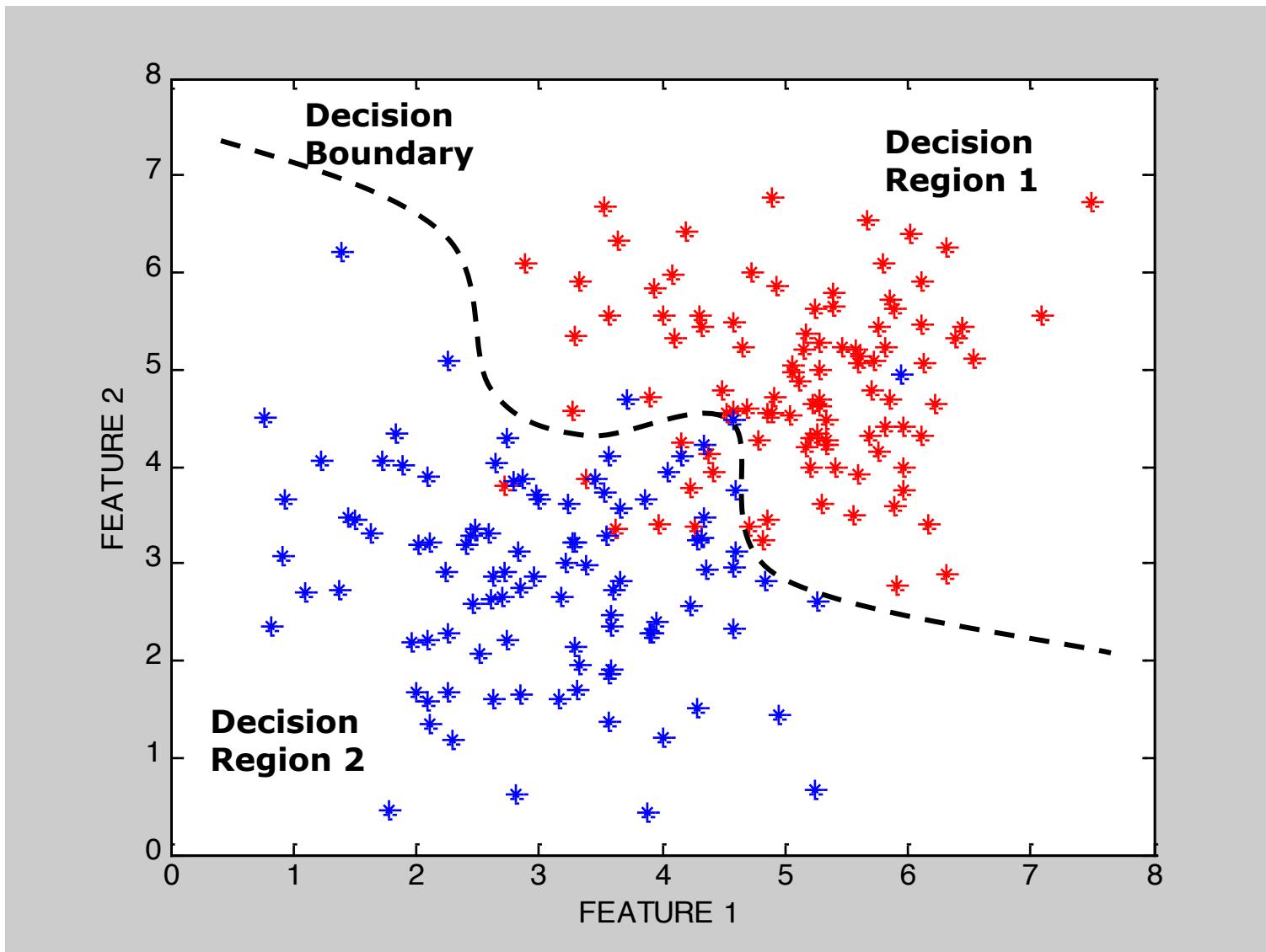
Logistic regression with more than two classes

- Logistic regression easily generalized to more than two classes

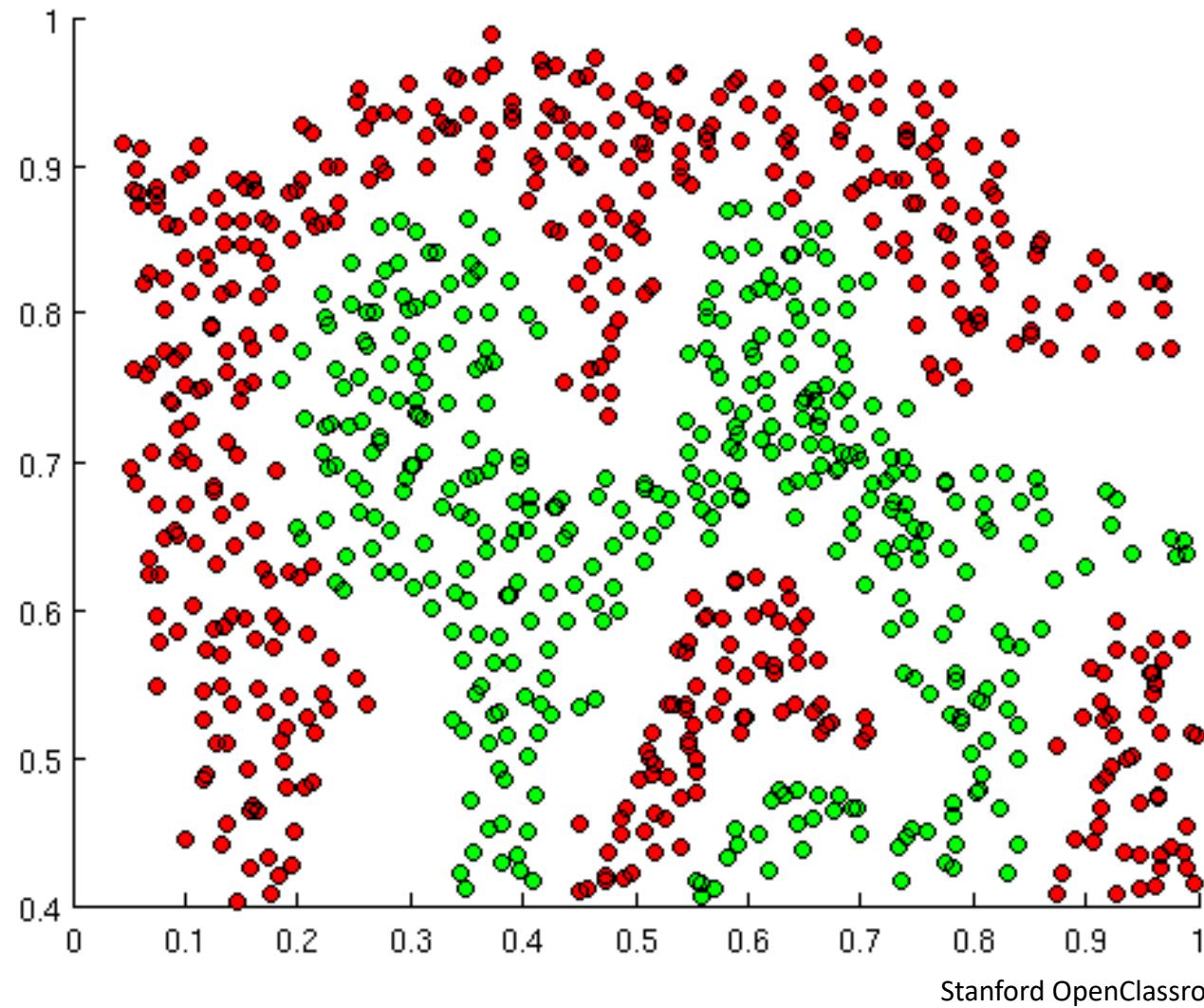
$$P(Y = k \mid \mathbf{X}) = \frac{e^{\beta_{0k} + \beta_{1k}X_1 + \dots + \beta_{pk}X_p}}{1 + e^{\beta_{0k} + \beta_{1k}X_1 + \dots + \beta_{pk}X_p}}$$

- Linear function for each class
- Multiclass logistic regression is also referred to as *multinomial regression*

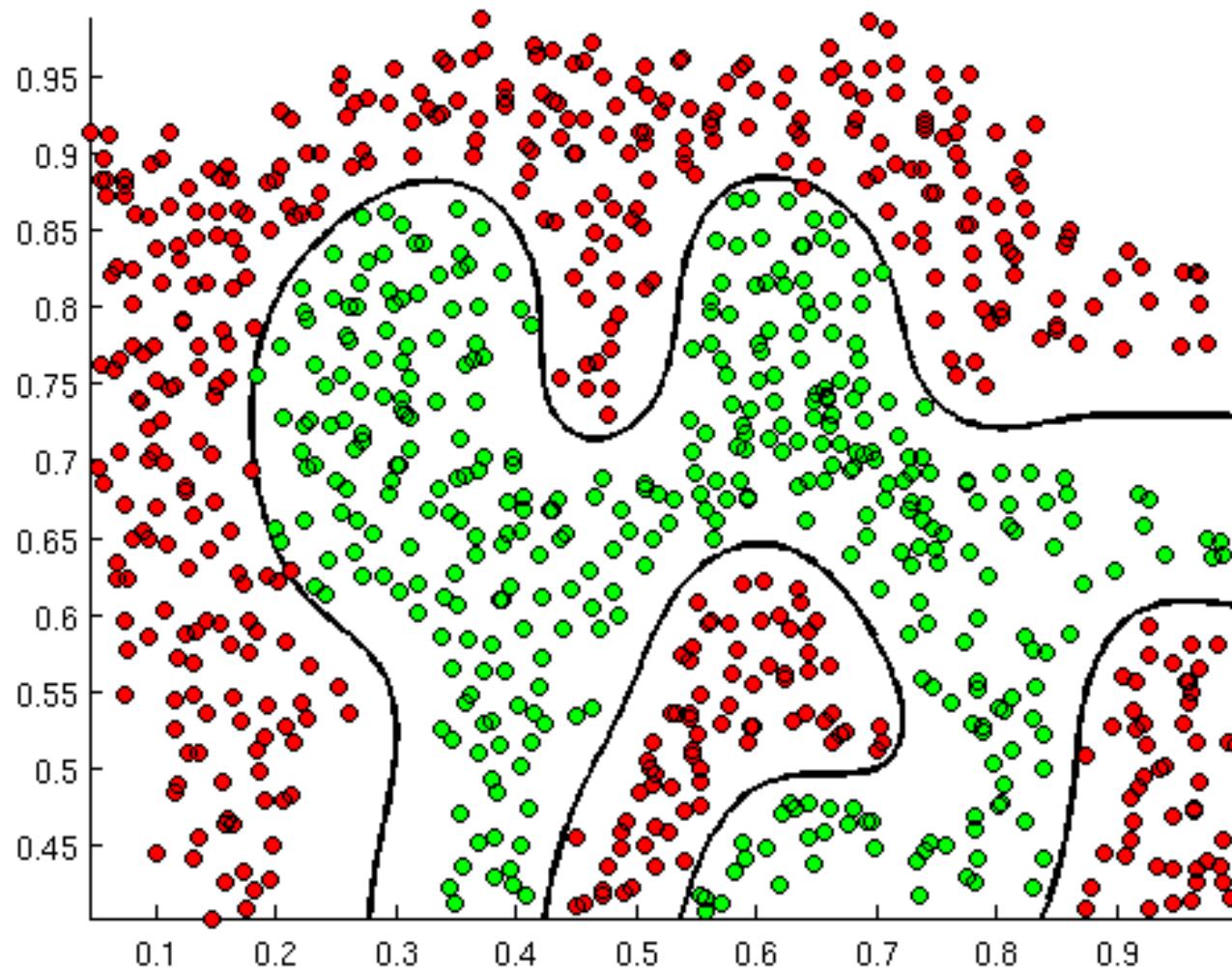
Decision Boundaries



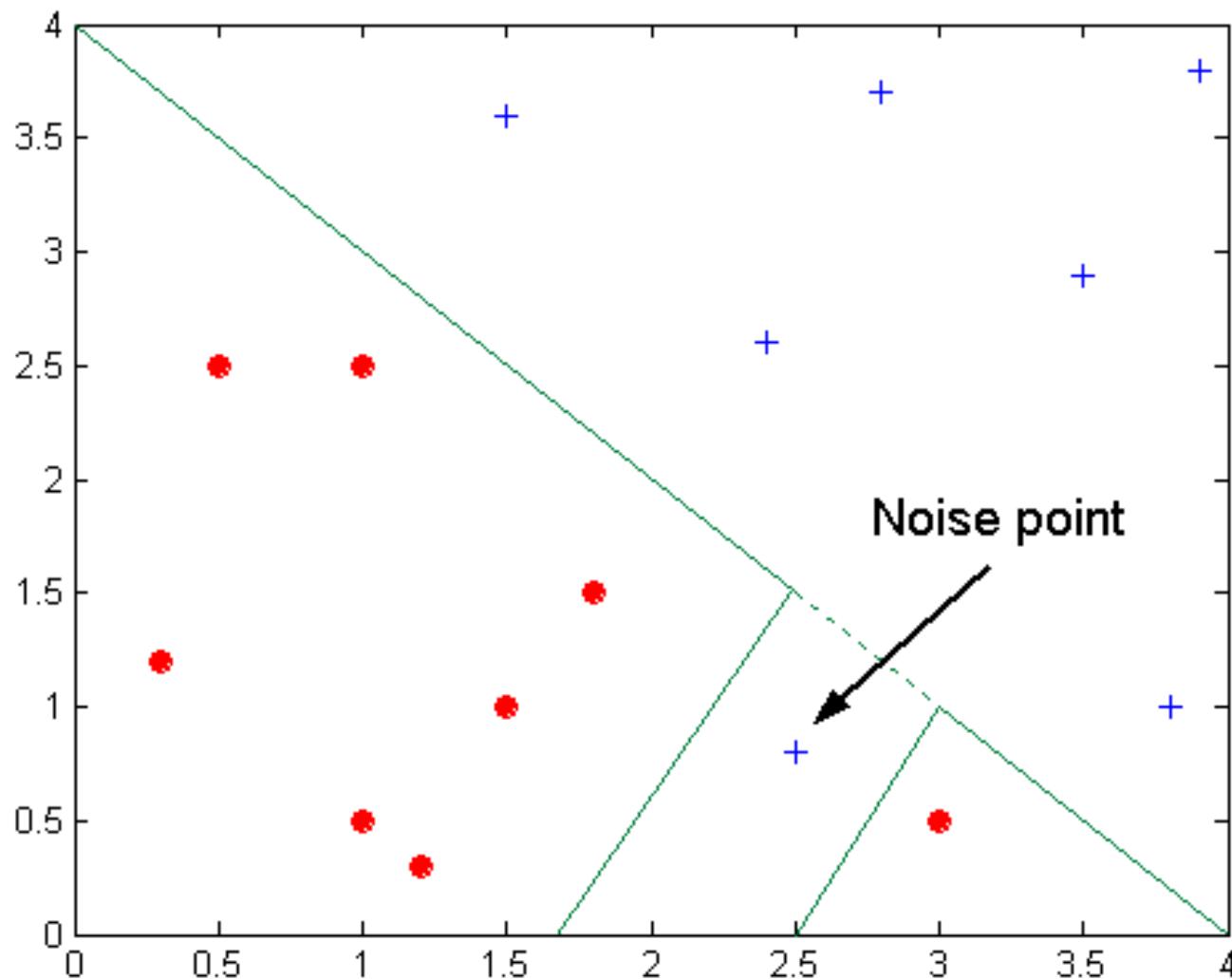
Non-linear Decision Boundaries



Non-linear Decision Boundaries



Overfitting Decision Boundary



Logistic Regression Regularization

- Regularization (shrinking) is a method that can be used to prevent overfitting
- It works by adding a term to the cost function that penalizes for extreme parameter values

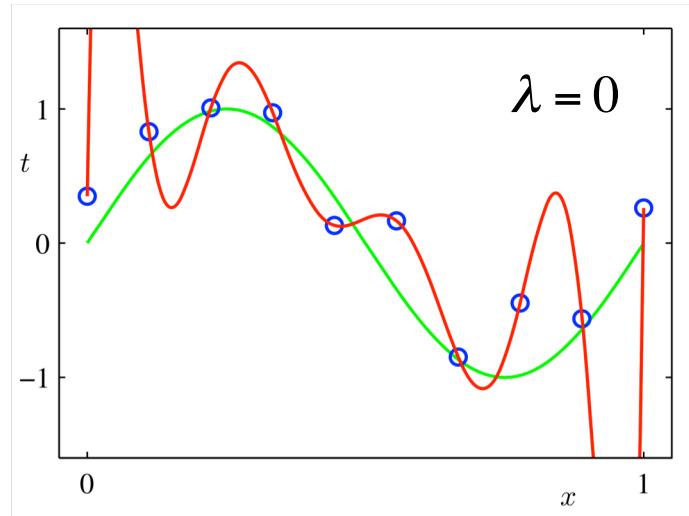
$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right]$$

Regularization parameter

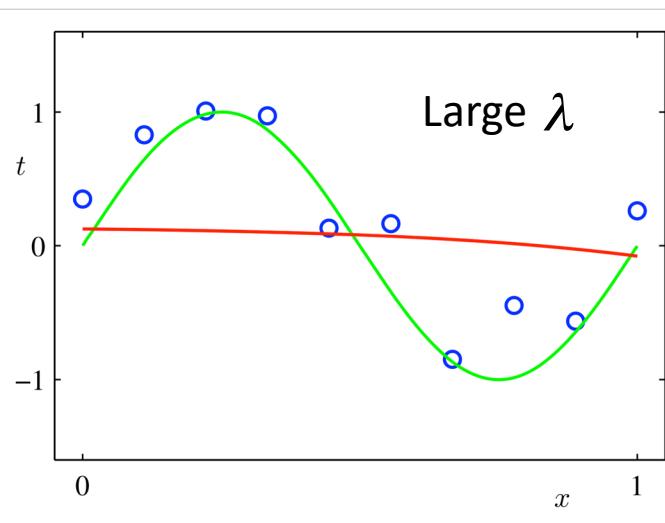
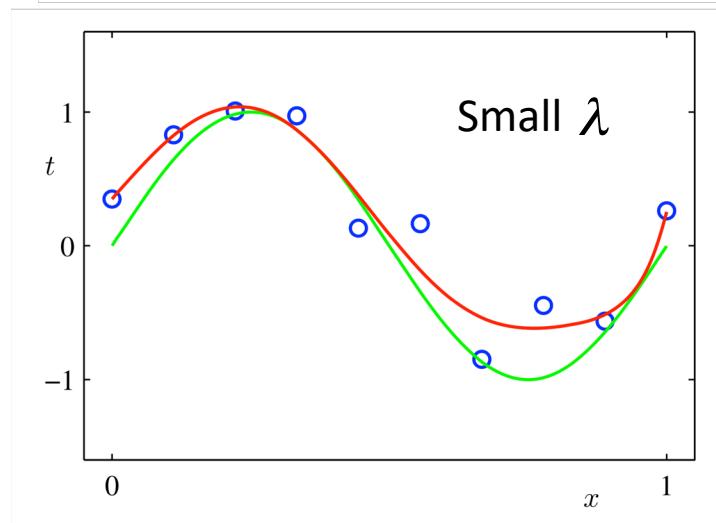
 

squared L2-norm
(≈magnitude of θ)

The effect of regularization



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



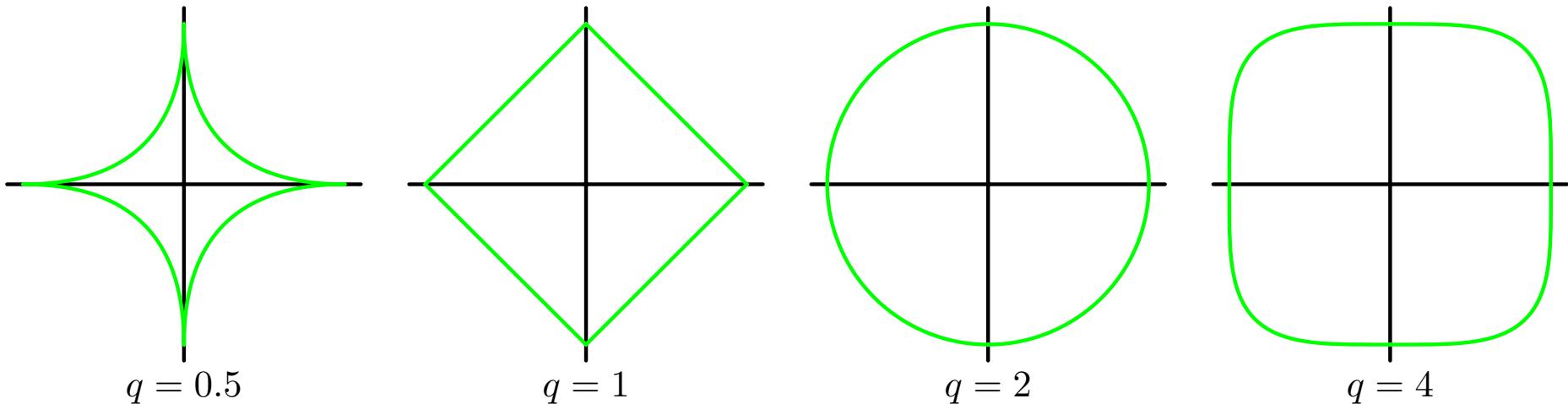
Logistic Regression Regularization

- Several types
 - L1 (LASSO) (L1-norm)
 - somewhat indifferent to very correlated predictors
 - will tend to pick one and ignore the rest
 - expects many coefficients to be close to zero, and a small subset to be larger and nonzero
 - tends to sparse models → coefficients for irrelevant features set to 0
 - L2 (Ridge) (squared L2-norm)
 - tends to shrink the coefficients of correlated predictors toward each other → allows coefficients to borrow strength from each other
 - extreme case of k identical predictors, they each get identical coefficients with $1/k^{\text{th}}$ the size that any single one would get if fit alone

Logistic Regression Regularization

- *Elastic net* regularization uses a linear combination of L1 and L2 norms
 - two regularization parameters, λ_1 and λ_2
 - performs much like the lasso
 - removes any degeneracies and wild behavior caused by extreme correlations.
 - creates a useful compromise between ridge and lasso

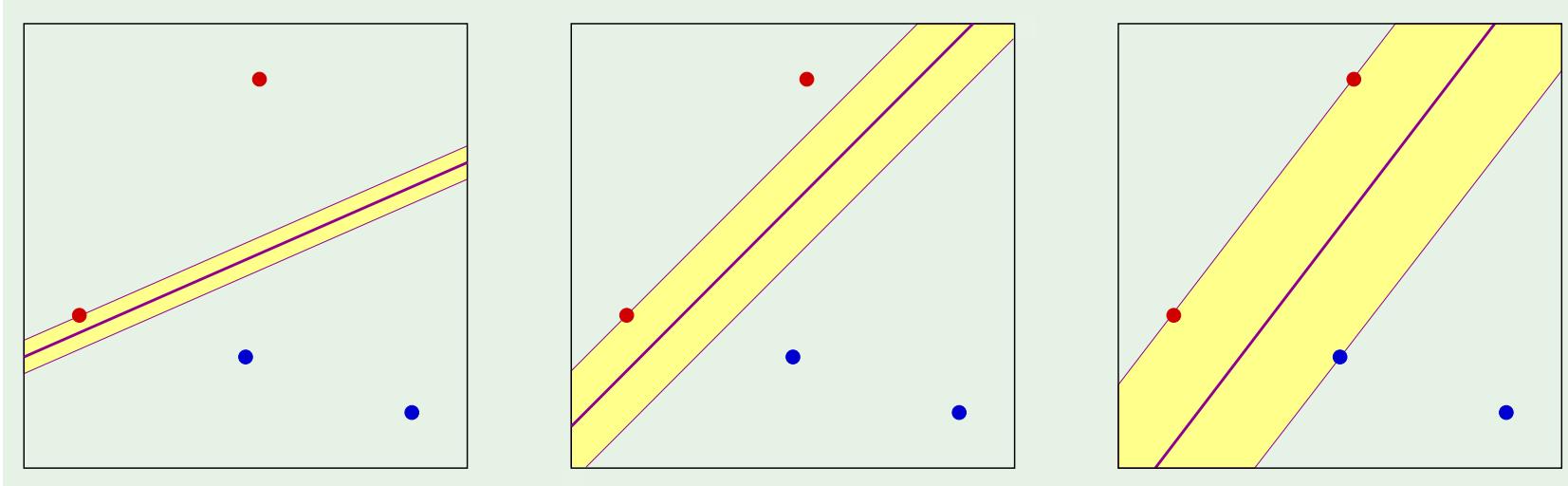
Contours of regularization term $|\theta|^q$



Outline

- Supervised learning
- Linear regression
- Logistic regression
- Support Vector Machine (SVM)

Which is a better linear separation



Two questions:

1. Why is bigger margin better?
2. Which \mathbf{w} maximizes the margin?

Finding \mathbf{w} with large margin

Let \mathbf{x}_n be the nearest data point to the plane $\mathbf{w}^\top \mathbf{x} = 0$. How far is it?

2 preliminary technicalities:

1. Normalize \mathbf{w} :

$$|\mathbf{w}^\top \mathbf{x}_n| = 1$$

2. Pull out w_0 :

$$\mathbf{w} = (w_1, \dots, w_d) \quad \text{apart from } b$$

The plane is now $\boxed{\mathbf{w}^\top \mathbf{x} + b = 0}$ (no x_0)

Computing the distance

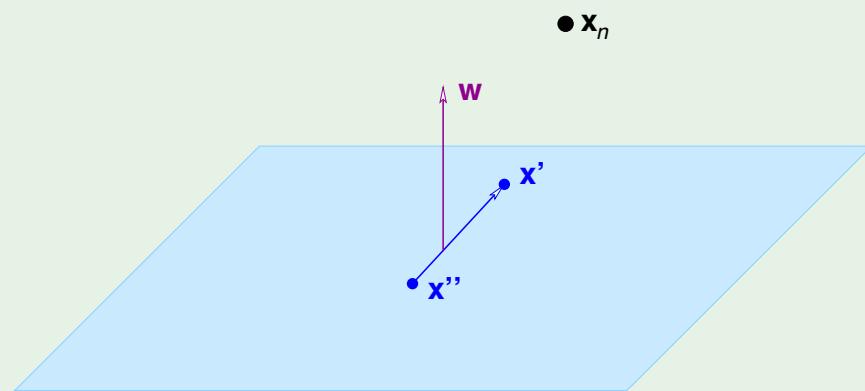
The distance between \mathbf{x}_n and the plane $\mathbf{w}^\top \mathbf{x} + b = 0$ where $|\mathbf{w}^\top \mathbf{x}_n + b| = 1$

The vector \mathbf{w} is \perp to the plane in the \mathcal{X} space:

Take \mathbf{x}' and \mathbf{x}'' on the plane

$$\mathbf{w}^\top \mathbf{x}' + b = 0 \quad \text{and} \quad \mathbf{w}^\top \mathbf{x}'' + b = 0$$

$$\implies \mathbf{w}^\top (\mathbf{x}' - \mathbf{x}'') = 0$$

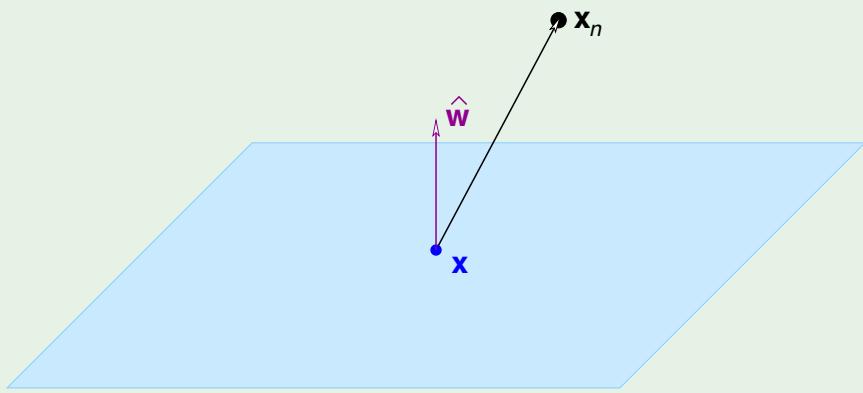


The distance from x_n to the plane

Distance between \mathbf{x}_n and the plane:

Take any point \mathbf{x} on the plane

Projection of $\mathbf{x}_n - \mathbf{x}$ on \mathbf{w}



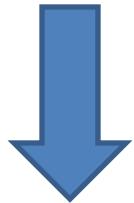
$$\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \implies \text{distance} = |\hat{\mathbf{w}}^\top (\mathbf{x}_n - \mathbf{x})|$$

$$\text{distance} = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\top \mathbf{x}_n - \mathbf{w}^\top \mathbf{x}| = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\top \mathbf{x}_n + b - \mathbf{w}^\top \mathbf{x} - b| = \frac{1}{\|\mathbf{w}\|}$$

The SVM optimization problem

$$\text{Maximize} \frac{1}{\|\mathbf{w}\|}$$

$$\text{subject to } \min_{n=1,2,\dots,N} |\mathbf{w}^\top \mathbf{x}_n + b| = 1$$



$$\text{Notice: } |\mathbf{w}^\top \mathbf{x}_n + b| = y_n (\mathbf{w}^\top \mathbf{x}_n + b)$$

$$\text{Minimize} \frac{1}{2} \mathbf{w}^\top \mathbf{w}$$

$$\text{subject to } y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 \quad \text{for } n = 1, 2, \dots, N$$

Constrained optimization

$$\text{Minimize} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w}$$

$$\text{subject to} \quad y_n (\mathbf{w}^T \mathbf{x}_n + b) \geq 1 \quad \text{for } n = 1, 2, \dots, N$$

$$\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

Lagrange? inequality constraints \implies KKT

Lagrange formulation

$$\text{Minimize } \mathcal{L}(\mathbf{w}, \mathbf{b}, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^\top \mathbf{w} - \sum_{n=1}^N \alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1)$$

w.r.t. \mathbf{w} and b and maximize w.r.t. each $\alpha_n \geq 0$



$$\nabla_{\mathbf{w}} \mathcal{L} = \mathbf{w} - \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n = \mathbf{0}$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{n=1}^N \alpha_n y_n = 0$$

Dual formulation

$$\mathbf{w} = \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n \quad \text{and} \quad \sum_{n=1}^N \alpha_n y_n = 0$$

in the Lagrangian

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^\top \mathbf{w} - \sum_{n=1}^N \alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1)$$

we get

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{x}_n^\top \mathbf{x}_m$$

Maximize w.r.t. to $\boldsymbol{\alpha}$ subject to $\alpha_n \geq 0$ for $n = 1, \dots, N$ and $\sum_{n=1}^N \alpha_n y_n = 0$

Quadratic programming (QP)

$$\min_{\alpha} \quad \frac{1}{2} \underbrace{\alpha^\top \begin{bmatrix} y_1 y_1 \mathbf{x}_1^\top \mathbf{x}_1 & y_1 y_2 \mathbf{x}_1^\top \mathbf{x}_2 & \dots & y_1 y_N \mathbf{x}_1^\top \mathbf{x}_N \\ y_2 y_1 \mathbf{x}_2^\top \mathbf{x}_1 & y_2 y_2 \mathbf{x}_2^\top \mathbf{x}_2 & \dots & y_2 y_N \mathbf{x}_2^\top \mathbf{x}_N \\ \dots & \dots & \dots & \dots \\ y_N y_1 \mathbf{x}_N^\top \mathbf{x}_1 & y_N y_2 \mathbf{x}_N^\top \mathbf{x}_2 & \dots & y_N y_N \mathbf{x}_N^\top \mathbf{x}_N \end{bmatrix}}_{\text{quadratic coefficients}} \alpha + \underbrace{(-1^\top) \alpha}_{\text{linear}}$$

subject to

$$\underbrace{\mathbf{y}^\top \alpha = 0}_{\text{linear constraint}}$$

$$\underbrace{0}_{\text{lower bounds}} \leq \alpha \leq \underbrace{\infty}_{\text{upper bounds}}$$

Solution

Solution: $\alpha = \alpha_1, \dots, \alpha_N$

$$\implies \mathbf{w} = \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n$$

KKT condition: For $n = 1, \dots, N$

$$\alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1) = 0$$

$\alpha_n > 0 \implies \mathbf{x}_n$ is a **support vector**

Support vectors

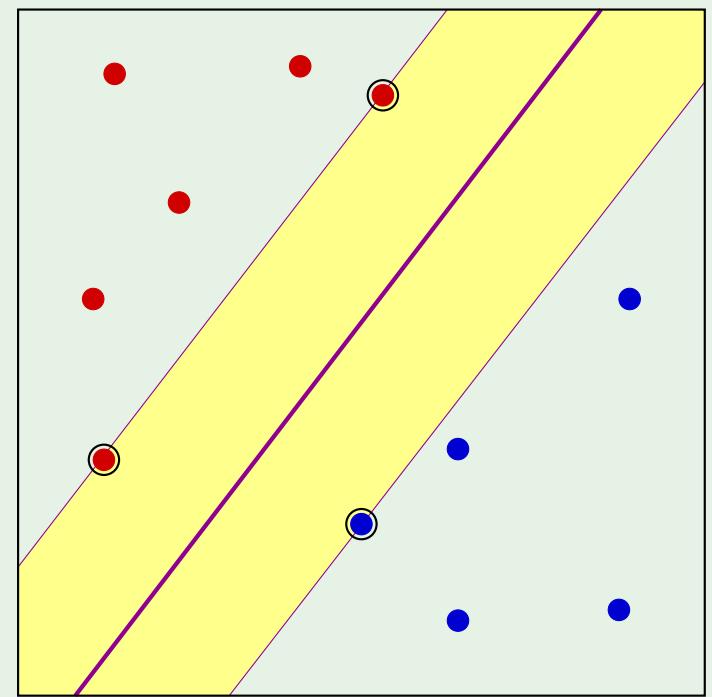
Closest \mathbf{x}_n 's to the plane: achieve the margin

$$\implies y_n (\mathbf{w}^\top \mathbf{x}_n + b) = 1$$

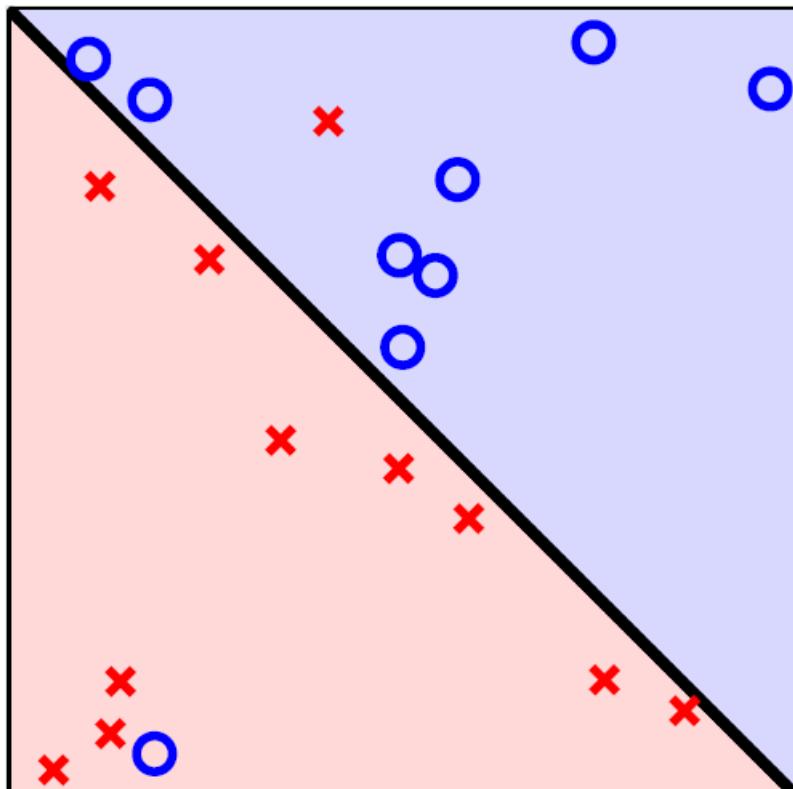
$$\mathbf{w} = \sum_{\mathbf{x}_n \text{ is SV}} \alpha_n y_n \mathbf{x}_n$$

Solve for b using any SV:

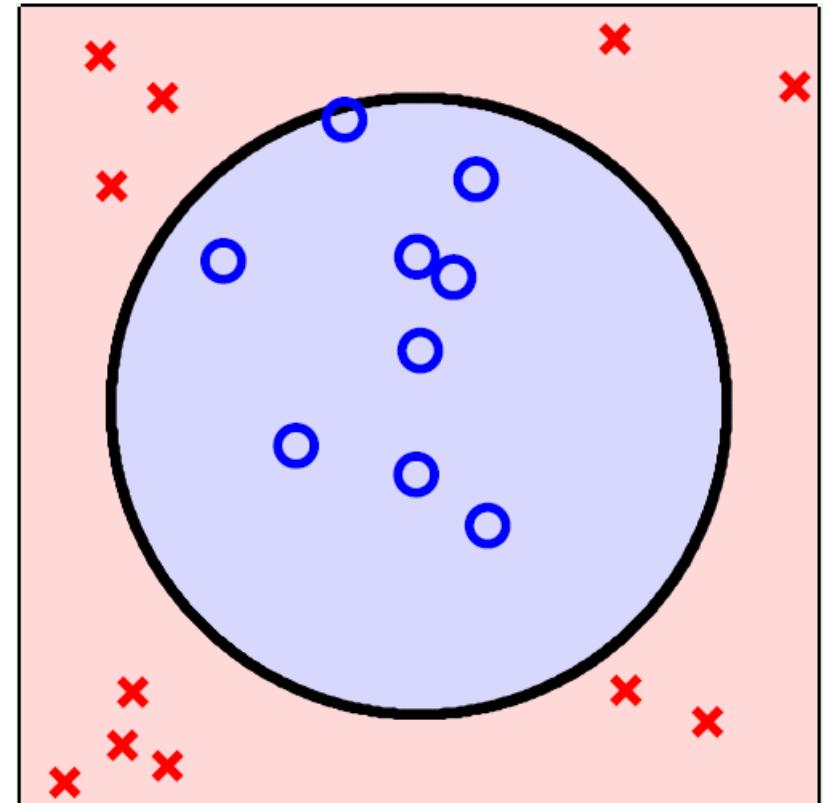
$$y_n (\mathbf{w}^\top \mathbf{x}_n + b) = 1$$



Non-separable Data



Slightly
non-separable

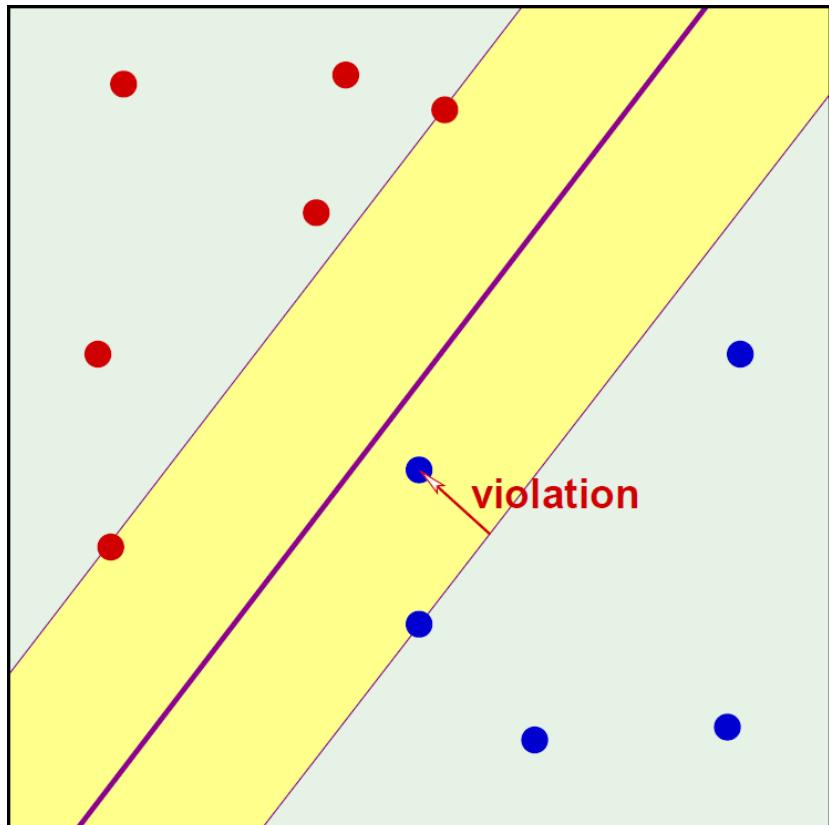


Seriously
non-separable

Soft-margin SVM

Margin violation: $y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1$ fails

Quantify: $y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 - \xi_n$



$$\text{Total violation} = \sum_{n=1}^N \xi_n$$

ξ_n

= ksi = “amount” of margin violation

$$\xi_n \geq 0$$

Soft-margin SVM optimization

- Similar to hard-margin optimization, but get compromise between maximizing the margin and allowing for violations

Still get large margin after term minimization

Allow for small violations by minimizing

Minimize

$$\frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{n=1}^N \xi_n$$

subject to

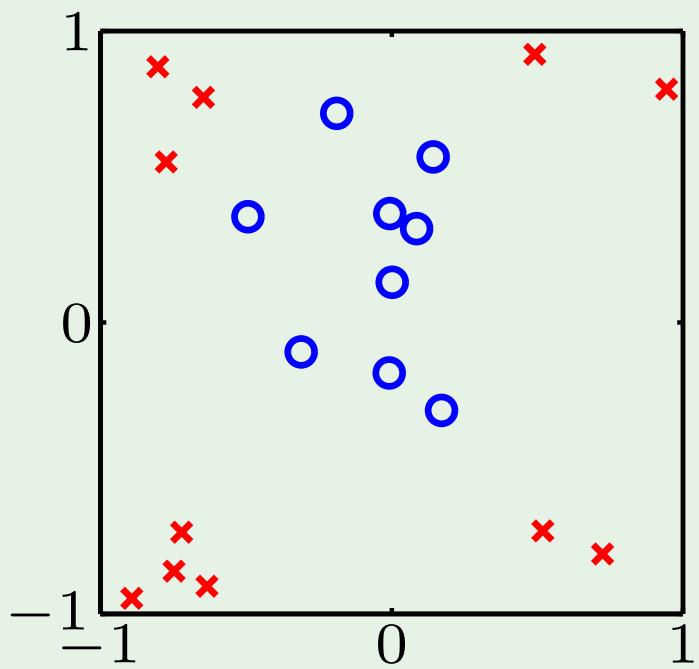
$$y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 - \xi_n$$

for $n = 1, \dots, N$

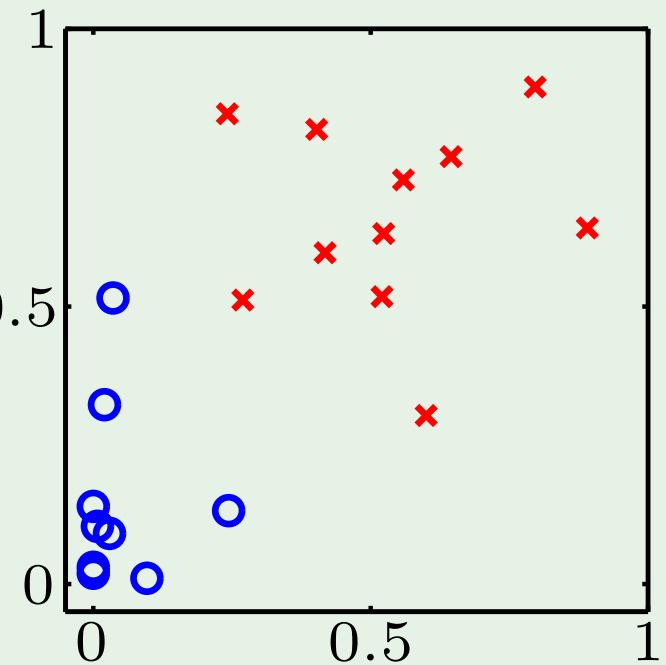
and $\xi_n \geq 0$ for $n = 1, \dots, N$

Kernel transform

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{z}_n^\top \mathbf{z}_m$$



$$\mathcal{X} \longrightarrow \mathcal{Z}$$



Thank you!