# Supervised learning: linear regression, SVM, Neural Networks

**Shikui Tu**

**Department of Computer Science and Engineering, Shanghai Jiao Tong University**

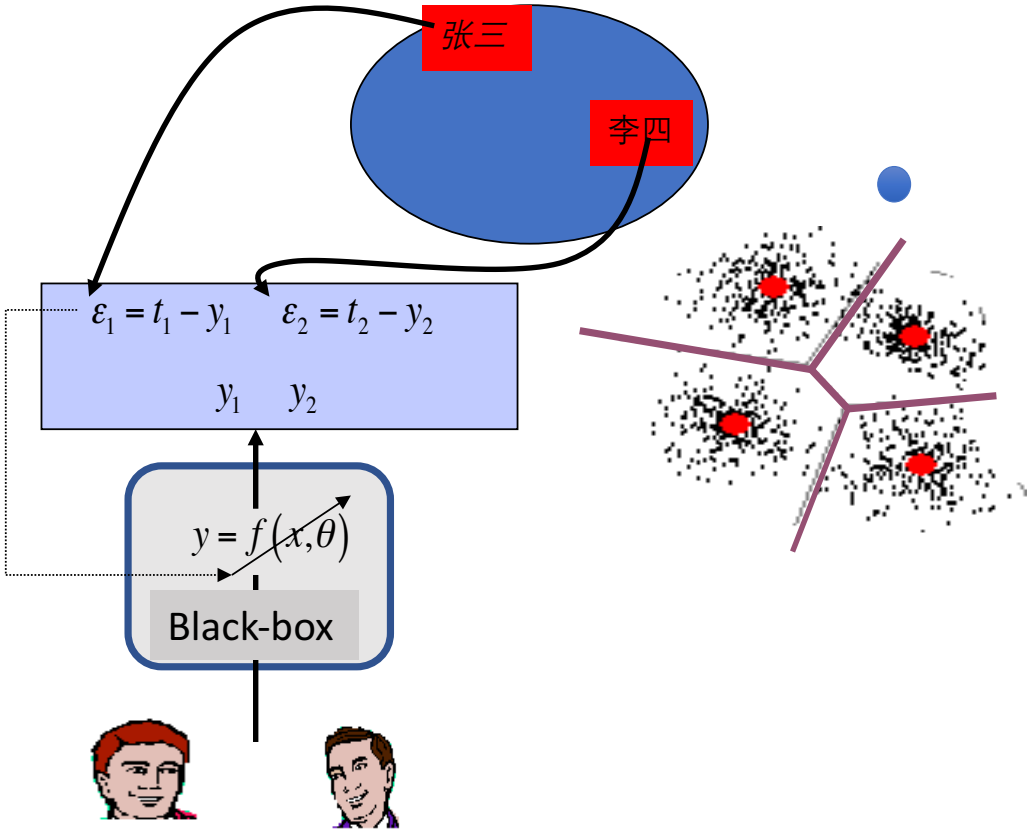
**2021-05-07**

# Outline

- **Supervised learning**
- Linear regression
- Logistic regression
- Neural Networks

# 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

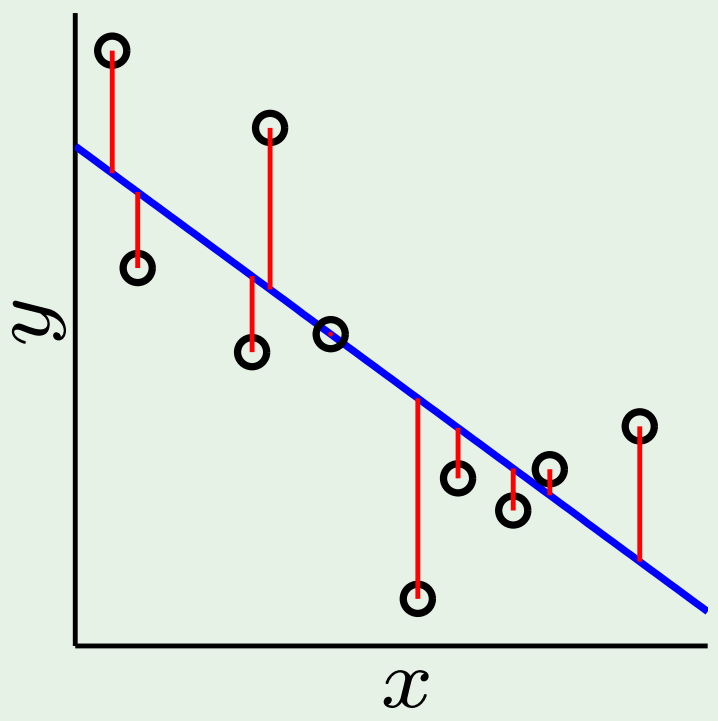
- **Linear regression**

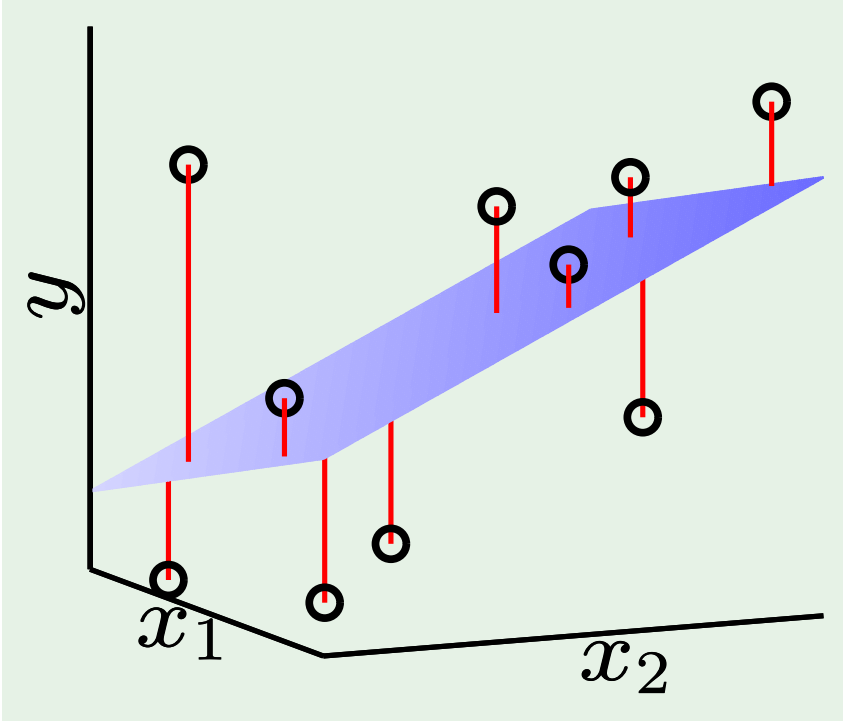
- Logistic regression

- Neural Networks

# Linear regression

"Regression" usually means "real-value output".

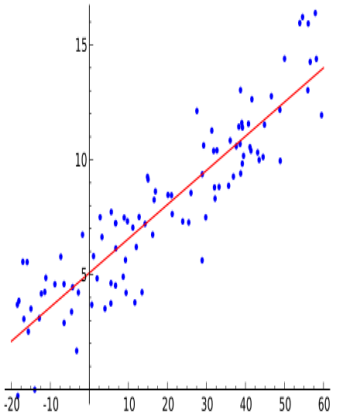


# Basic concepts of linear regression

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

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




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

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

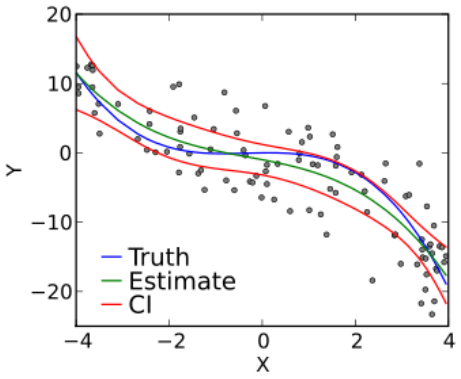
*regressors*, *exogenous variables*, *explanatory variables*, *covariates*, *input variables*, *predictor variables*, or *independent variables*

The matrix ***X*** is sometimes called the [design matrix.]

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

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

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



# Least Square Estimation

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

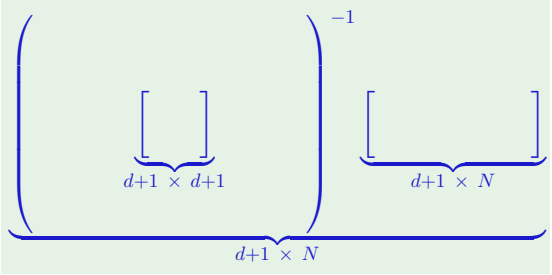
Minimize the sum of squared residuals (errors):

$$\min_{\beta} \varepsilon^T \varepsilon = (Y - X\beta)^T (Y - X\beta) \quad \Longleftrightarrow \quad \min_{w} \sum_{t} \left\| e_t \left( x_t, w \right) \right\|^2$$

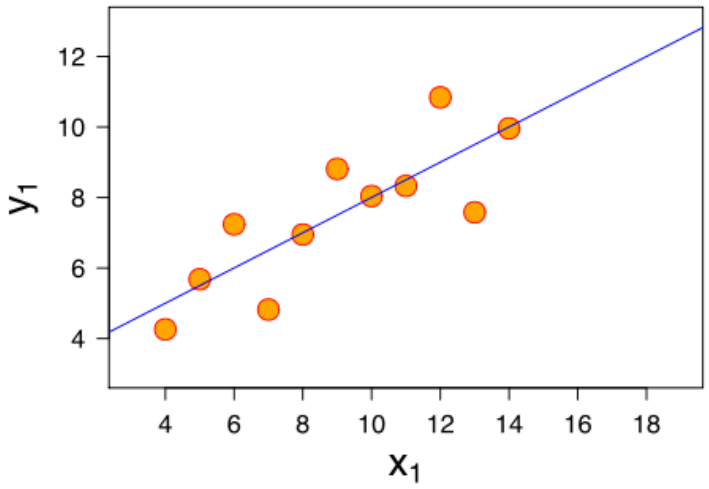
By taking the derivative to zero, we have:

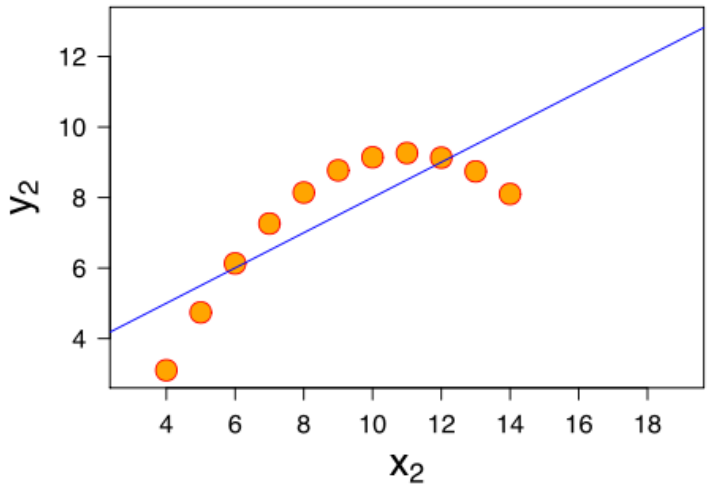
$$\beta = (X^T X)^{-1} X^T Y$$

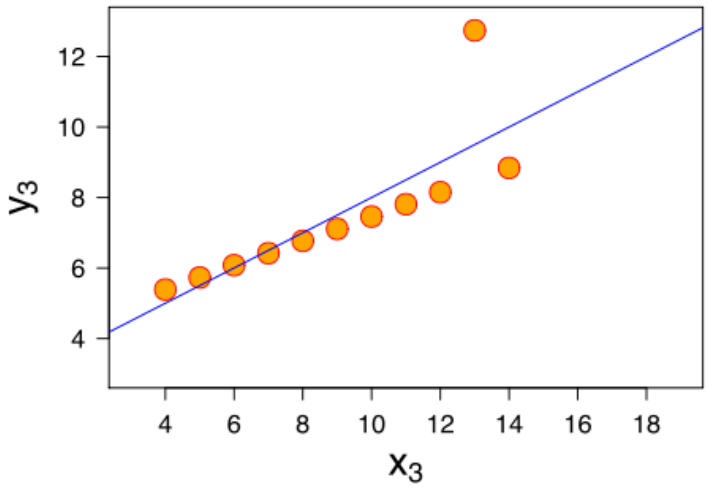
The pseudo-inverse $X^+ = (X^T X)^{-1} X^T$

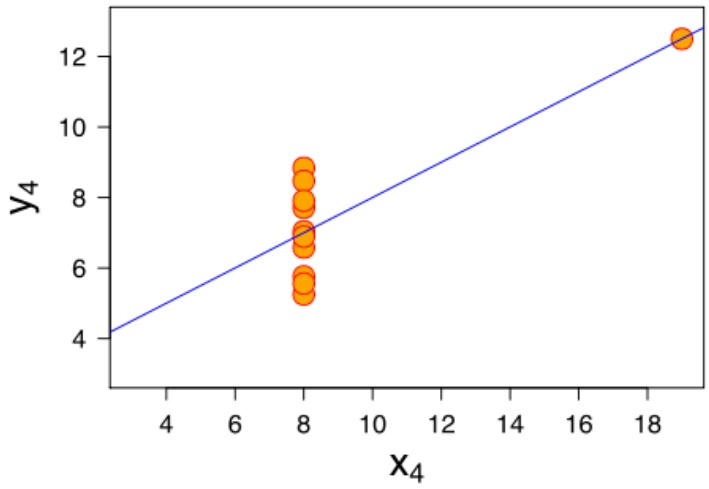


# Limitations of linear regression



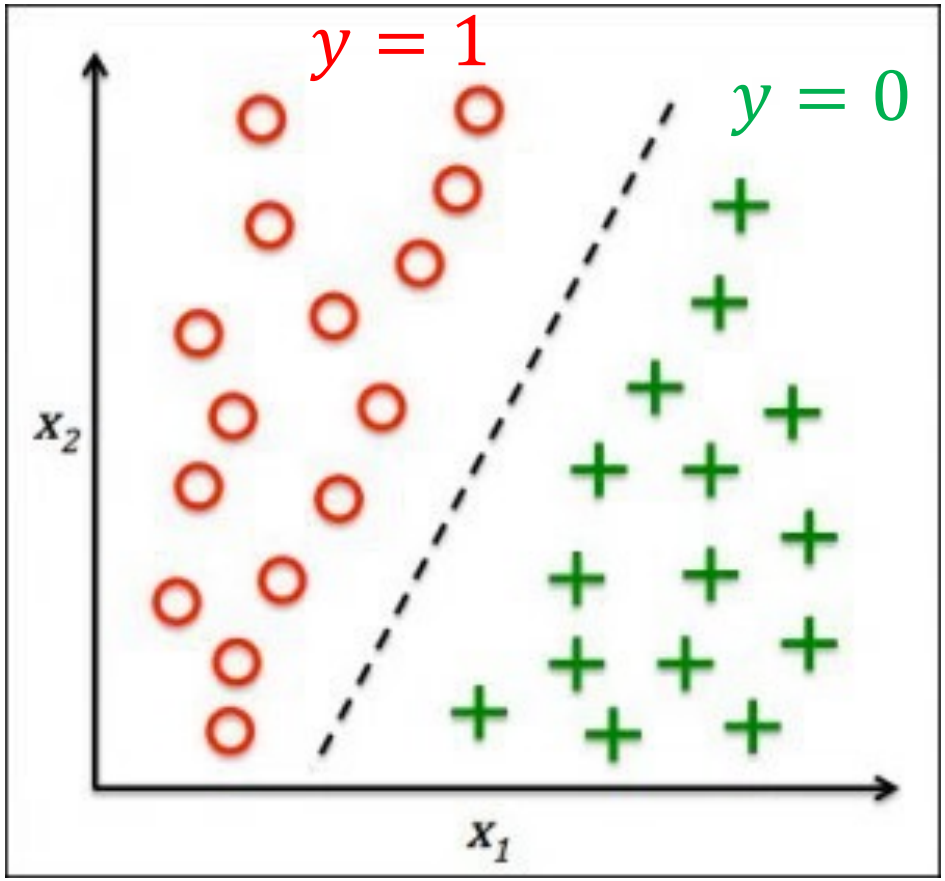



# Outline

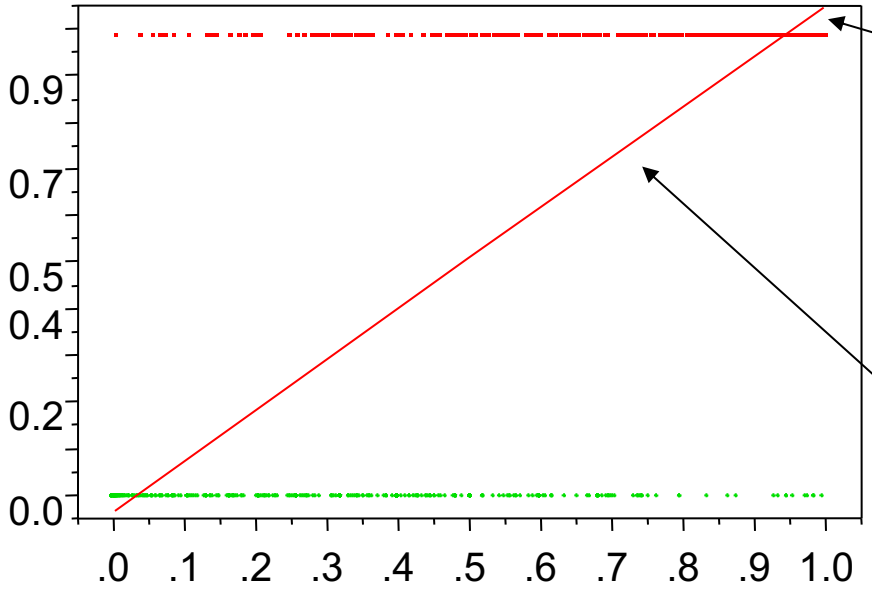
- Supervised learning
- Linear regression
- **Logistic regression**
- Neural Networks

# Classification



# Why not Linear Regression?

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




How do we interpret values greater than 1 and smaller than 1?

How do we interpret values of Y between 0 and 1?

# Problems

- The regression line $\beta_0+\beta_1 X$ can take on any value between negative infinity and positive infinity.

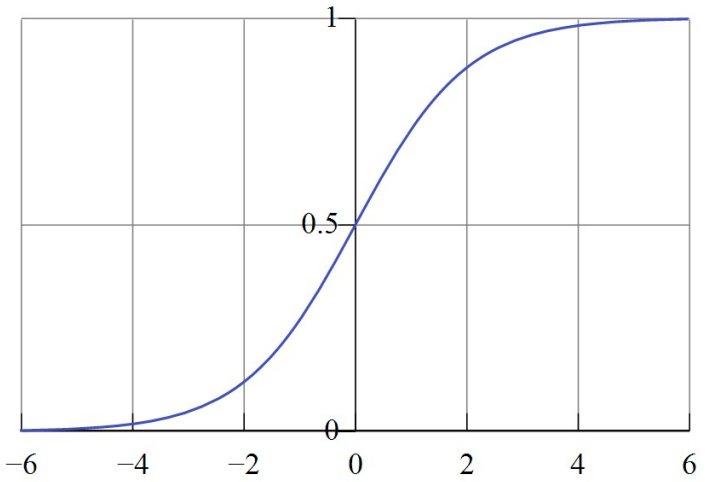
- In the classification problem, output Y can only take on two possible values: 0 or 1.

- 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!

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




Wikipedia

# Logistic Regression

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

- Euler's number e = 2.71828…
- 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 $\beta_1$

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

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

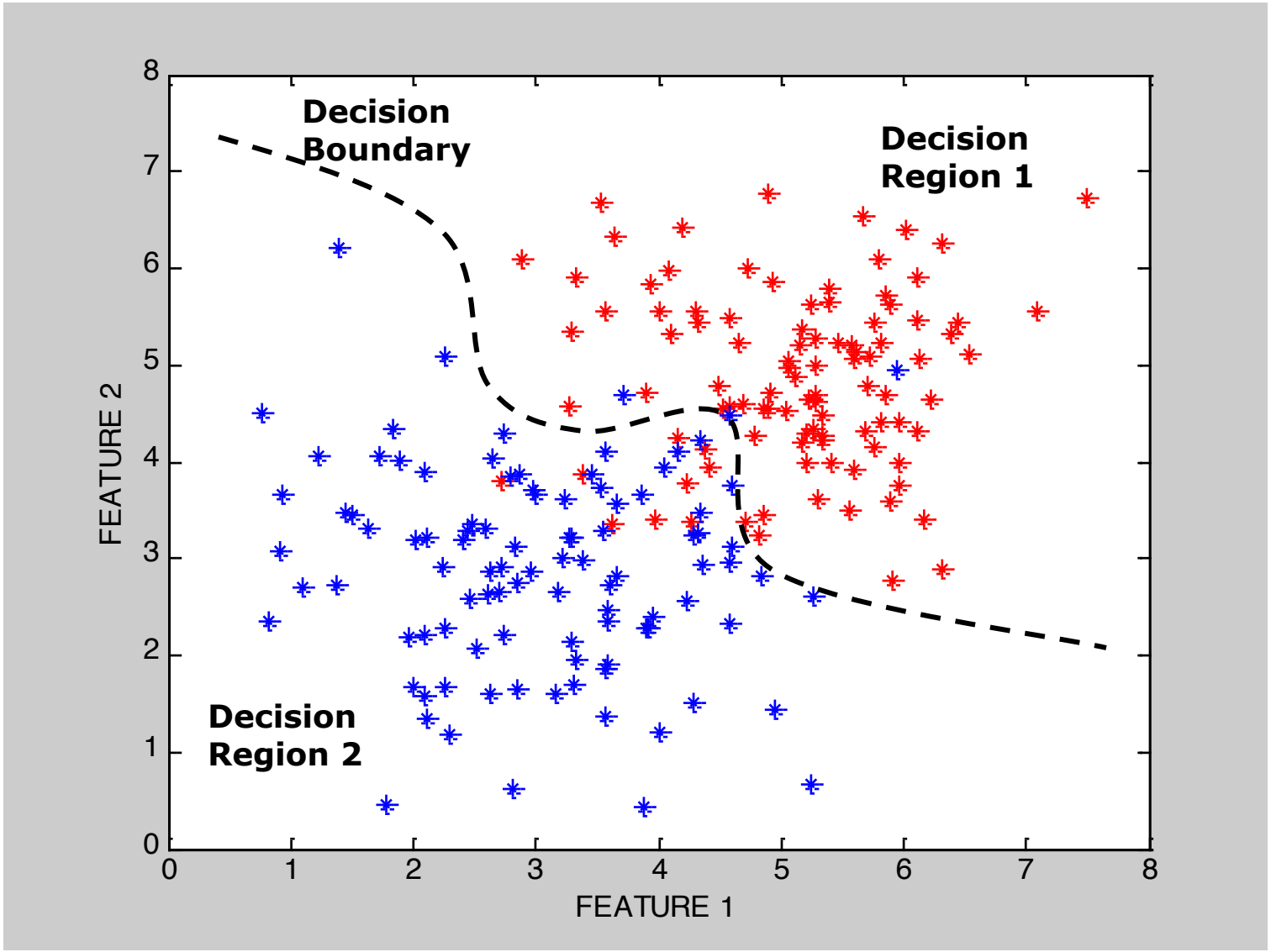
# Logistic regression with more than two classes

- Logistic regression easily generalized to more than two classes

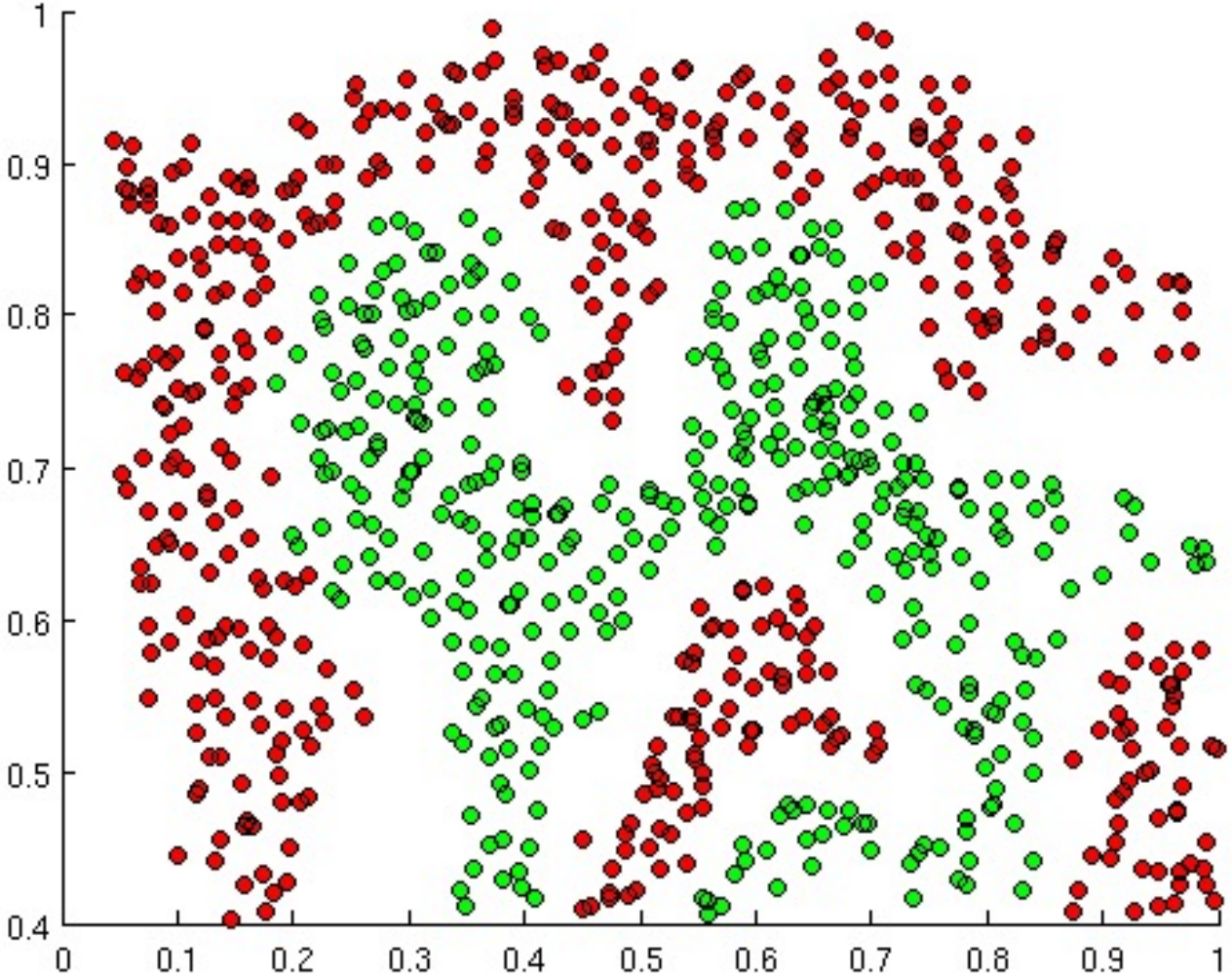
$$P(Y = k \mid \mathrm{X}) = \frac{e^{\beta_{0k} + \beta_{1k} X + \ldots + \beta_{pk} X}}{1 + \sum_{k=1}^{n} e^{\beta_{0k} + \beta_{1k} X + \ldots + \beta_{pk} X}}$$

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

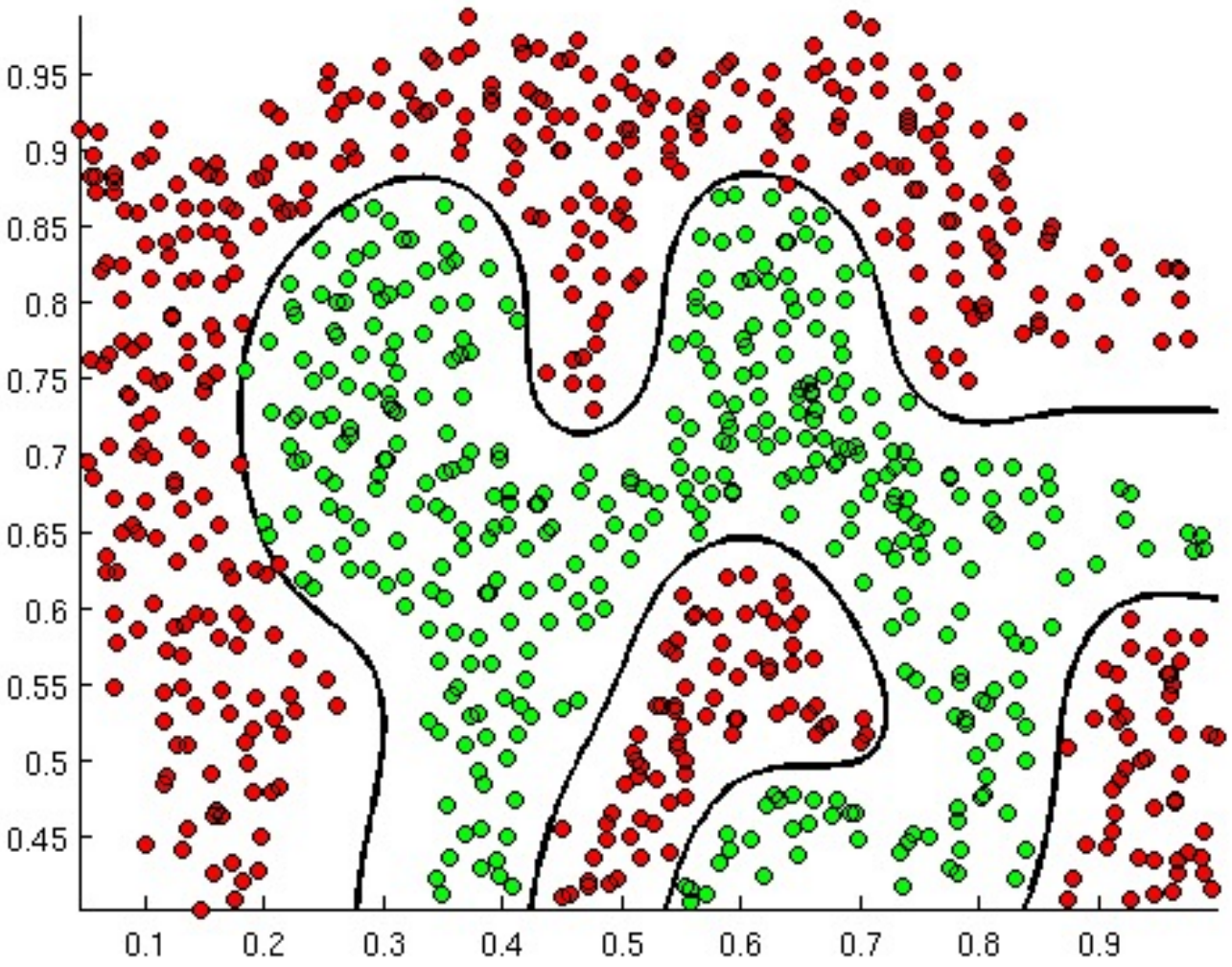
# Decision Boundaries



# Non-linear Decision Boundaries



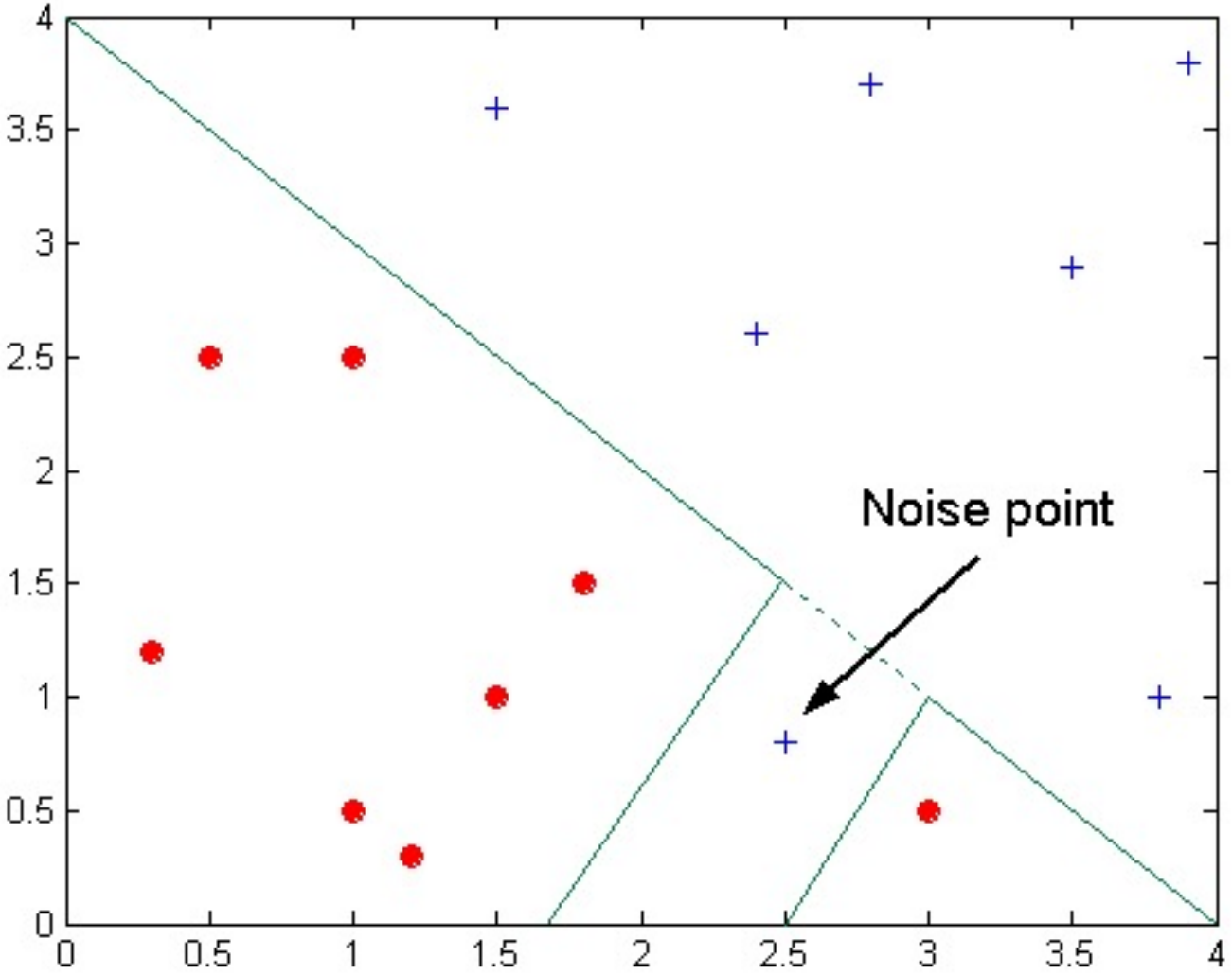
# Non-linear Decision Boundaries






Stanford OpenClassroom Machine Learning

# Overfitting Decision Boundary



# Regression Regularization

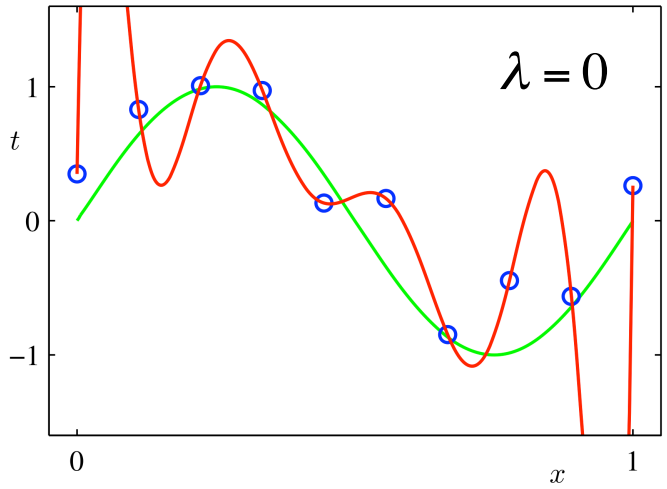
- Regularization (shrinking) is a method that can be used to prevent overfitting
- It works by adding a term to the cost function that penalizes for extreme parameter values

Regularization parameter

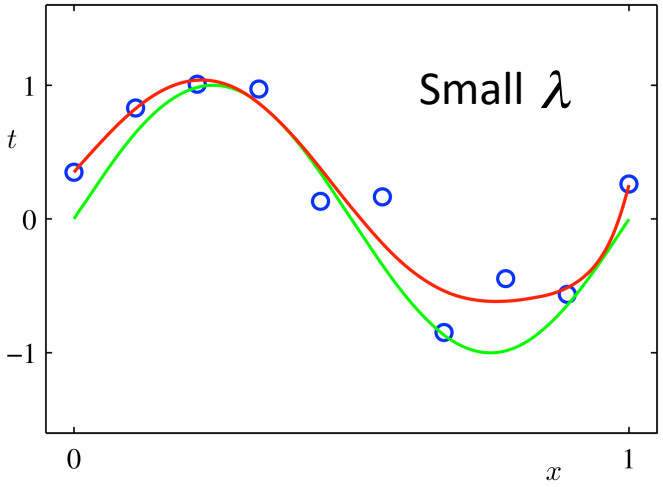
$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

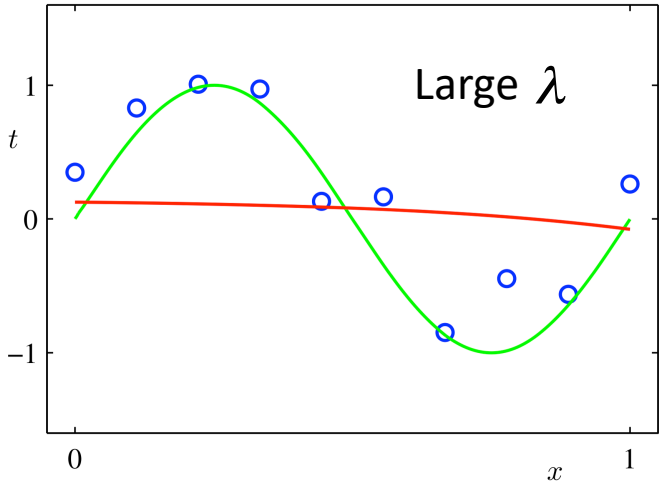
squared L2-norm
(≈magnitude of $\theta$)

# The effect of regularization




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



# 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 $1/k^{th}$ the size that any single one would get if fit alone



Friedman, J., Hastie, T., & Tibshirani, R. (2010)

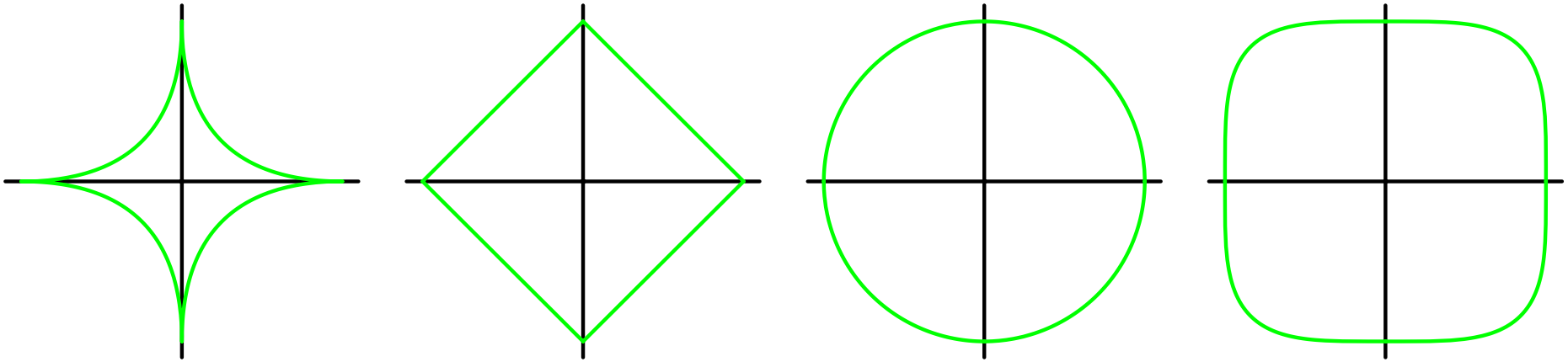
# Regression Regularization

- *Elastic net* regularization uses a linear combination of L1 and L2 norms
  - two regularization parameters, $\lambda_1$ and $\lambda_2$
  - performs much like the lasso
  - removes any degeneracies and wild behavior caused by extreme correlations.
  - creates a useful compromise between ridge and lasso



Friedman, J., Hastie, T., & Tibshirani, R. (2010)

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



$q = 0.5$ $q = 1$ $q = 2$ $q = 4$

Thank you!