COMP90051 **Statistical Machine Learning**

Semester 2, 2015

Workshop 2

THE UNIVERSITY OF MELBOURNE

Linear Regression

- Model representation
  - Single variable
    - $h_{(a,b)}(x) = a + bx$
  - Multiple variable
    - $h_w(\boldsymbol{x}) = \sum_{j=0}^{n}(w_j x_j)$
      $= \boldsymbol{w}'\boldsymbol{x}$
- Cost function
  - $J(\boldsymbol{w}) = \sum_{i=1}^{m}(h_w(\boldsymbol{X_i}) - y_i)^2$
- The objective is to minimise $J(\boldsymbol{w})$
  - Normal equation: $\boldsymbol{\hat{w}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$
  - Gradient descent (not this week)

X: 3 × 4 matrix

m instances

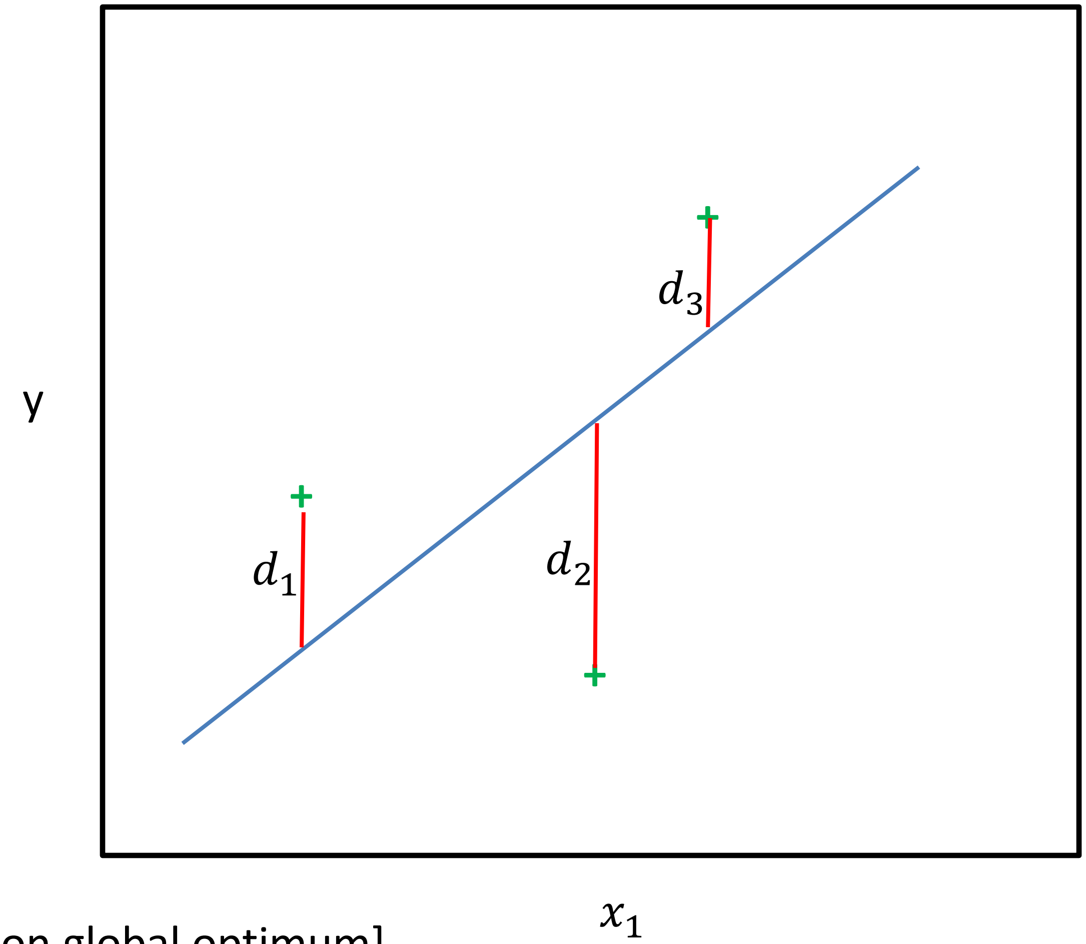
x_0	x_1	x_2	y
1	20	85	85
1	10	80	90
1	35	83	86
1	85	55	98

$n + 1$ features

[comments on matrices, vector, scalar; hypothesis vs cost function]

Cost Function: Square Error

$$J(\boldsymbol{w}) = \sum_{i=1}^{m}(h_w(\boldsymbol{X_i}) - y_i)^2 \ = d_1^2 + d_2^2 + d_3^2$$



[comments on global optimum]

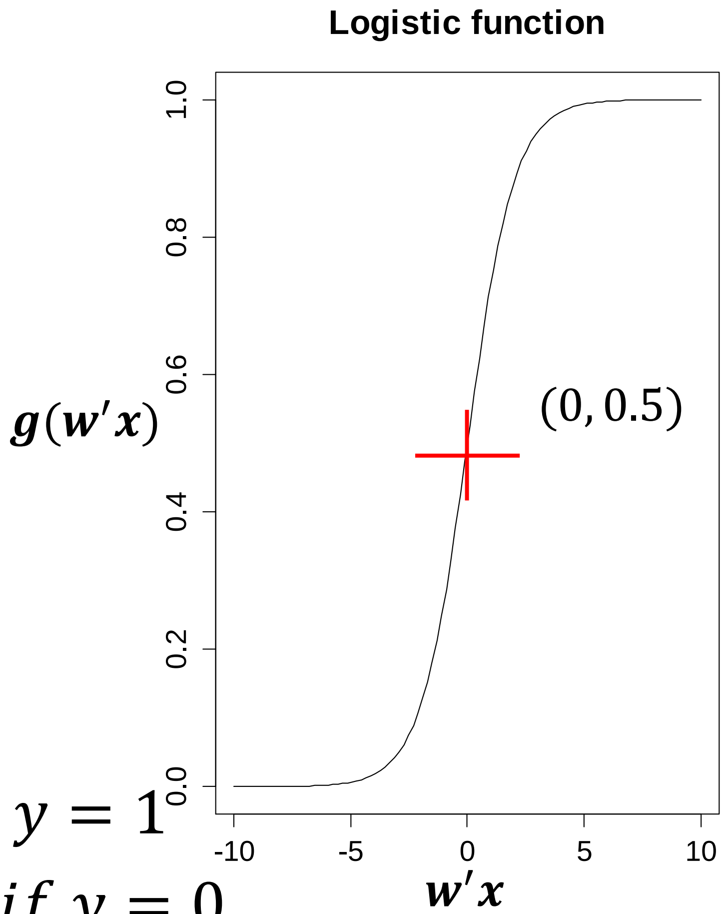
Example

- [Matlab: train a linear regression model using the normal equation, step by step]
- [R: lm() function]

[Matlab and R demo]

Logistic Regression

- Model representation
  * $h_w(\boldsymbol{x}) = g(\boldsymbol{w}'\boldsymbol{x})$
  * $g(z) = \frac{1}{1+e^{-z}}$
  * $\boldsymbol{w}'\boldsymbol{x} > 0 \rightarrow g(z) > 0.5$

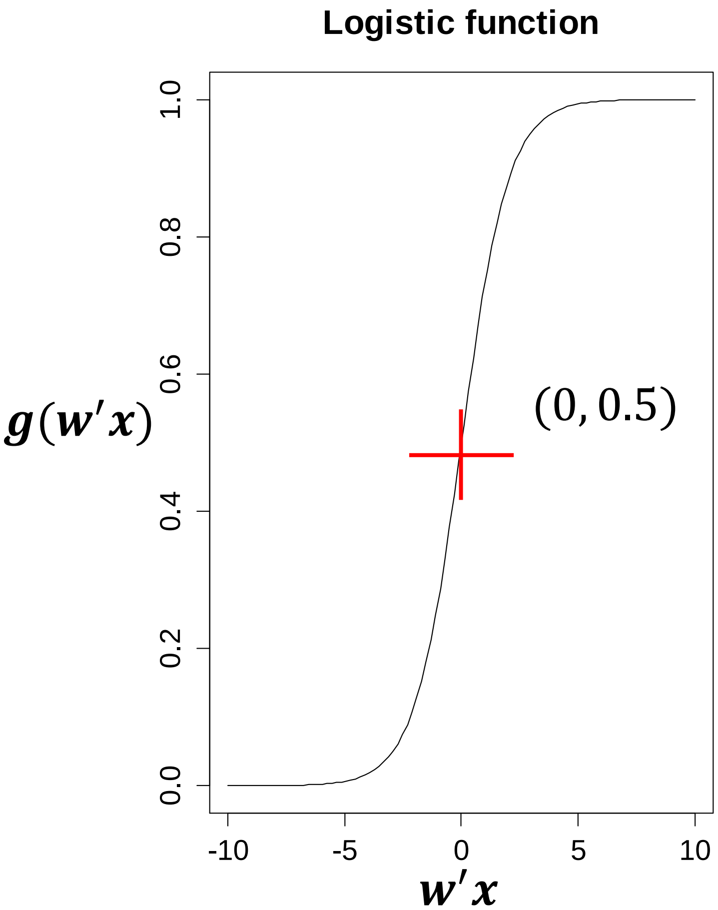


- Cost function
  * $y = \{0, 1\}$
  * $J(\boldsymbol{w}, x) = \begin{cases} -\log(h_w(\boldsymbol{x})), if\ y = 1 \\ -\log\left(1 - h_w(\boldsymbol{x})\right), if\ y = 0 \end{cases}$

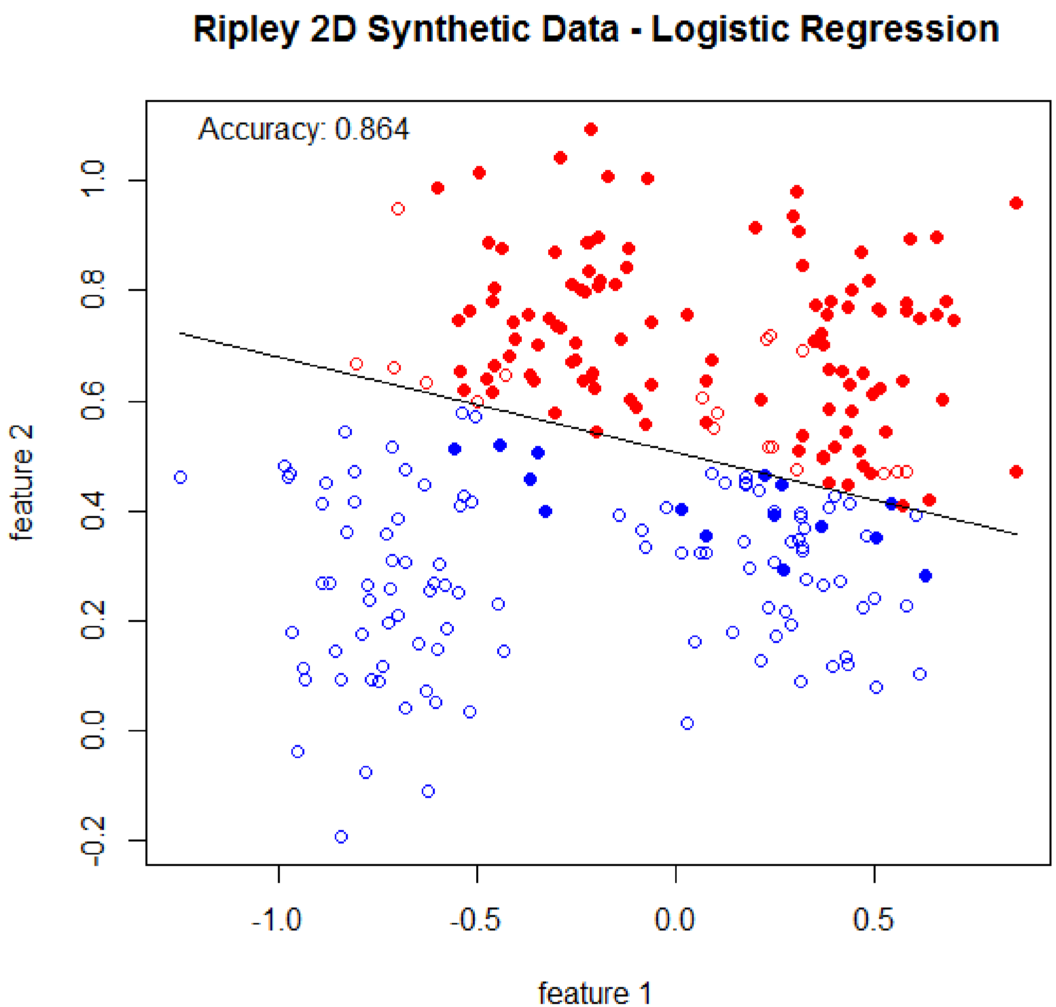
[comments on the decision boundary; the cost function; why not re-using square error]

Logistic Regression

- Model representation
  - $h_w(\boldsymbol{x}) = g(\boldsymbol{w}'\boldsymbol{x})$
  - $g(z) = \frac{1}{1+e^{-z}}$
  - $\boldsymbol{w}'\boldsymbol{x} > 0 \rightarrow g(z) > 0.5$
- Cost function
  - $y = \{0, 1\}$
  - $J(\boldsymbol{w}) = \sum_{i=1}^{m}[-y_i \log(h_w(X_i)) - (1-y_i)\log(1-h_w(X_i))]$

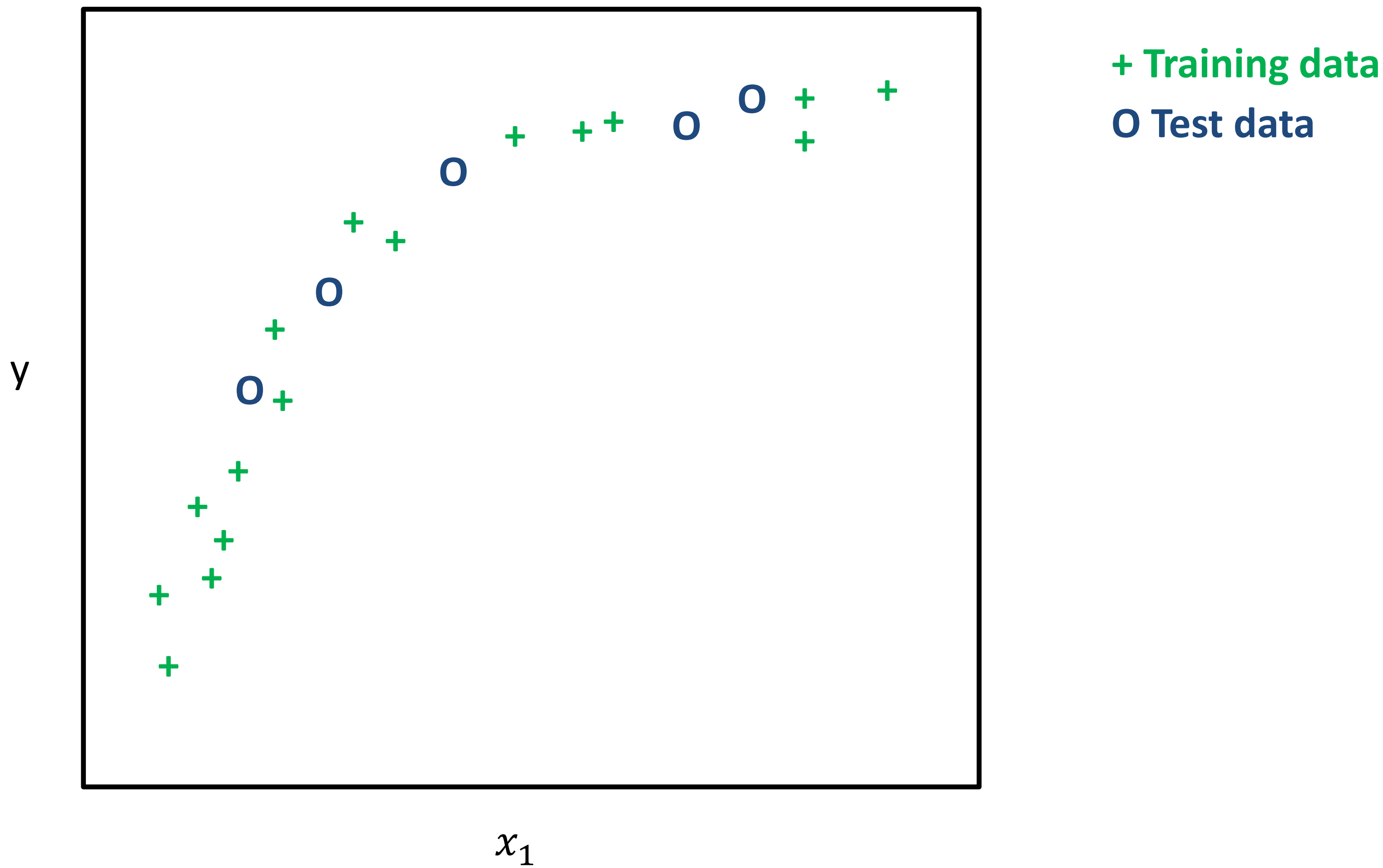


Logistic Regression: Example



[explain what does the line mean here; different meaning from linear regression; how to compute the decision boundary here]

More “Complex” Linear Regression



[hand draw examples to explain how we can increase the degree of polynomial to fit a “complex” dataset; Under-fitting and over-fitting]

Polynomial Feature Mapping

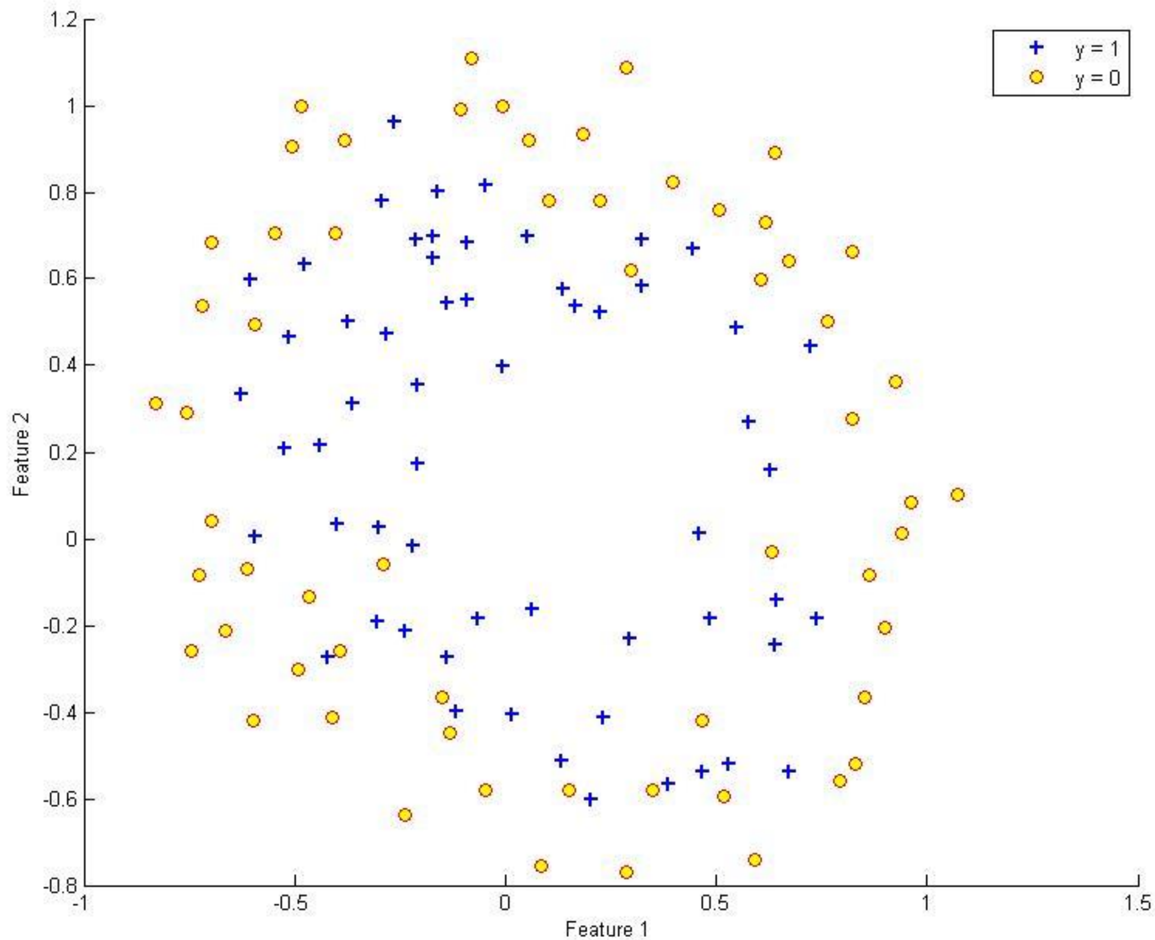
- Single feature

$$y = w_0 x_0 + w_1 x_1 + w_2 x_1^2 + w_3 x_1^3 + \cdots$$

- Multiple features

$$y = w_0 x_0 + w_1 x_1 + w_2 x_2 + w_3 x_1 x_2 + w_4 \, x_1^2 + w_5 x_2^2 + \cdots$$

More “Complex” Logistic Regression (Classification)

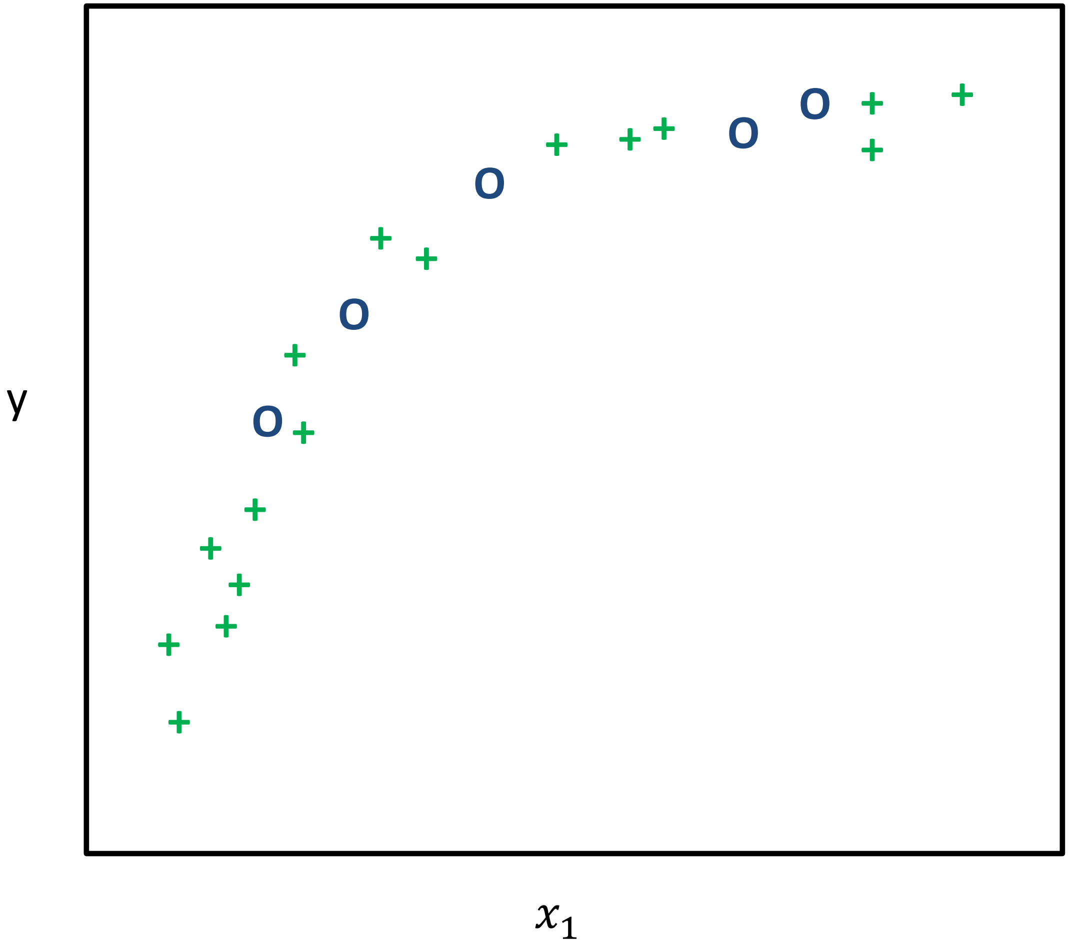


[comments on the feature mapping again]

Figure and dataset derived from http://cs229.stanford.edu/ by Andrew Ng

Bias vs Variance

- Under fitting → High Bias
- Over fitting → High Variance



[comments: hand draw figures to explain how to diagnose if a mode is under-fitting / over-fitting the data, using the error (training/test)-model complexity relationship figure]

Regularization

- Linear Regression

$$J(\boldsymbol{w}) = \sum_{i=1}^{m}(h_w(X_i) - y_i)^2 + \lambda\sum_{j=1}^{n} w_j^2$$

- Logistic Regression

$$J(\boldsymbol{w}) = \sum_{i=1}^{m}[-y_i\log(h_w(X_i)) - (1-y_i)\log(1-h_w(X_i))] + \lambda\sum_{j=1}^{n} w_j^2$$

[comments: explain how λ can control the complexity of the trained model]

More "Complex" Logistic Regression (Classification)

Train a model using only two features but with high order polynomial mapping

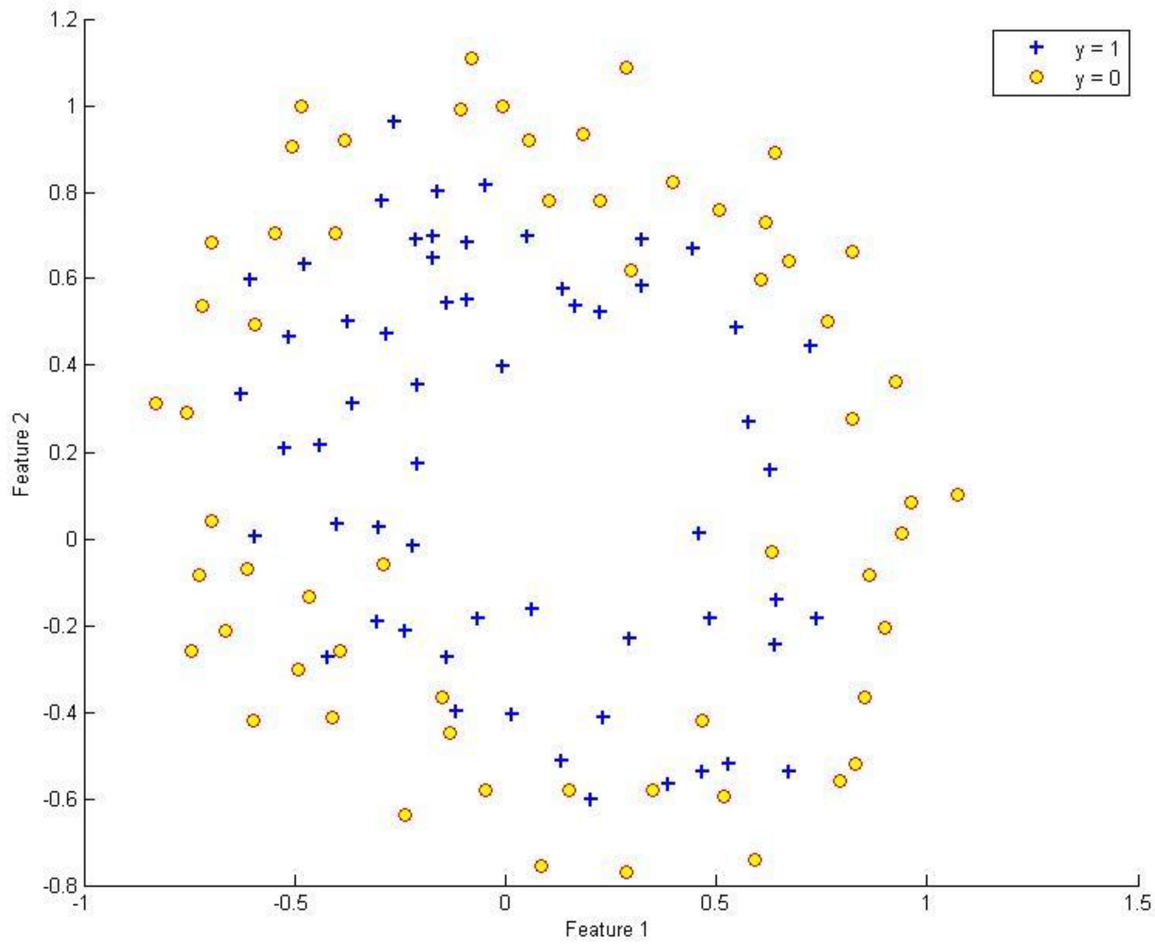


Figure and dataset derived from http://cs229.stanford.edu/ by Andrew Ng

$\lambda = 100$

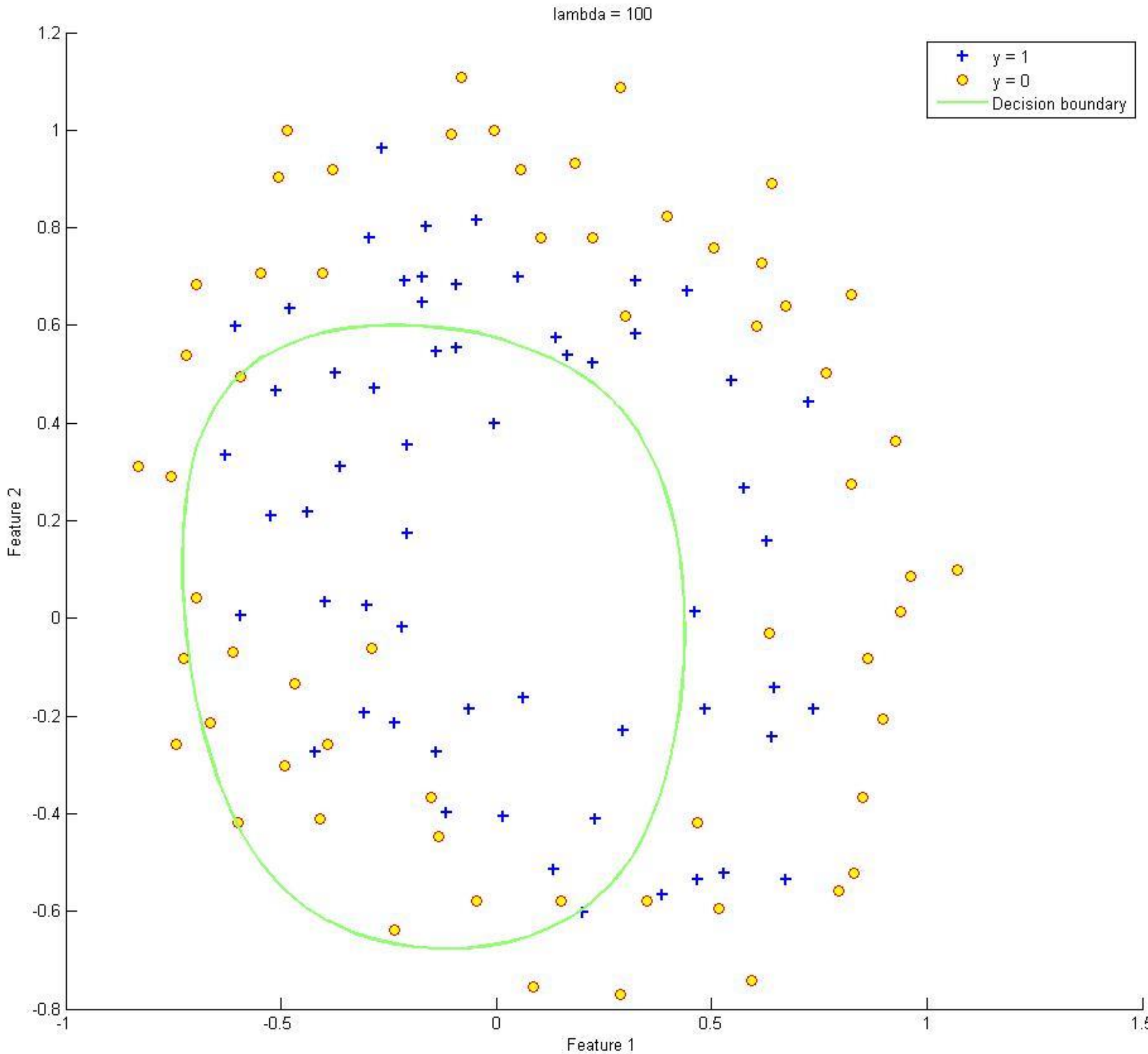


Figure and dataset derived from http://cs229.stanford.edu/ by Andrew Ng

$\lambda = 10$

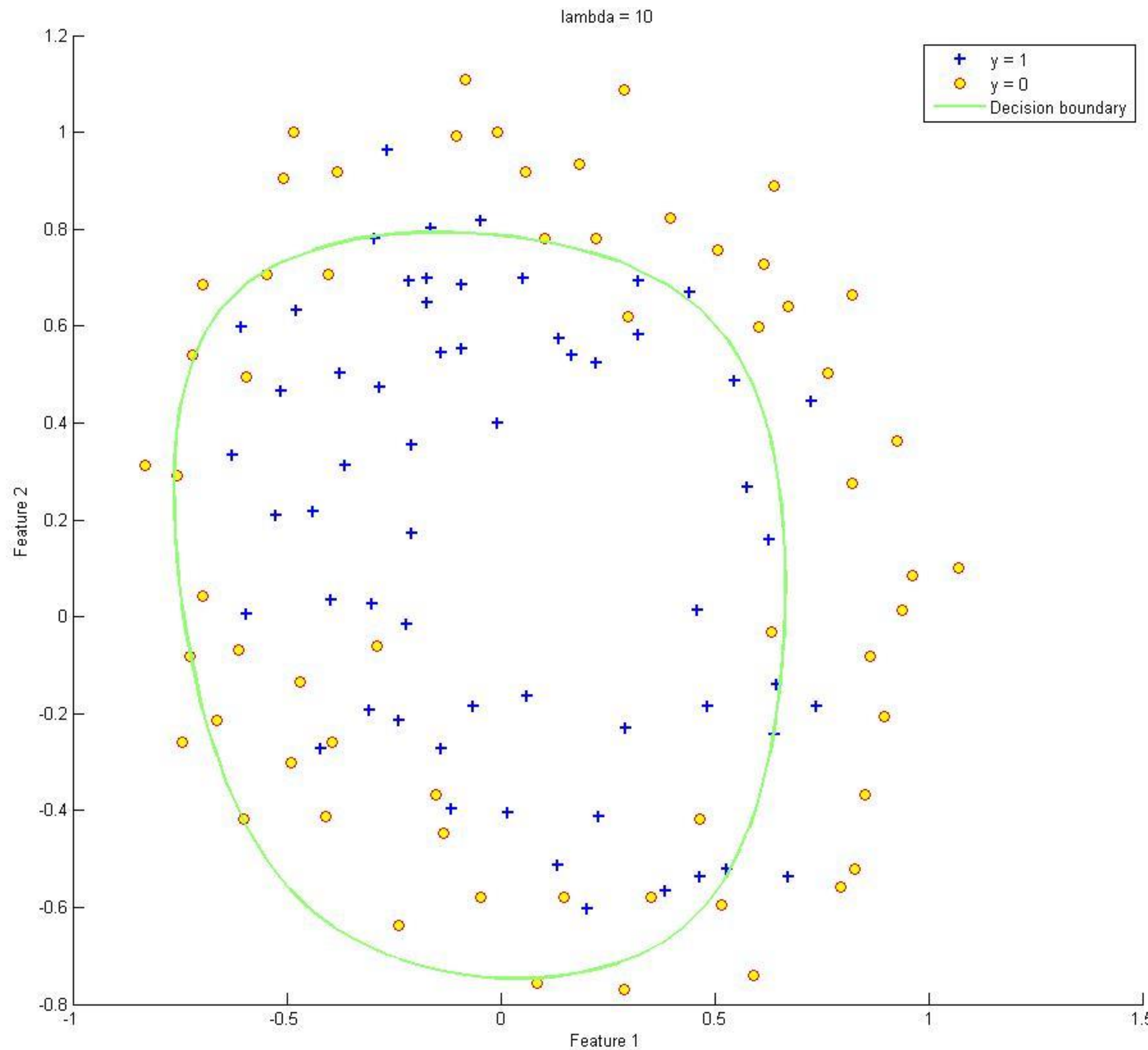


Figure and dataset derived from http://cs229.stanford.edu/ by Andrew Ng

$\lambda = 1$

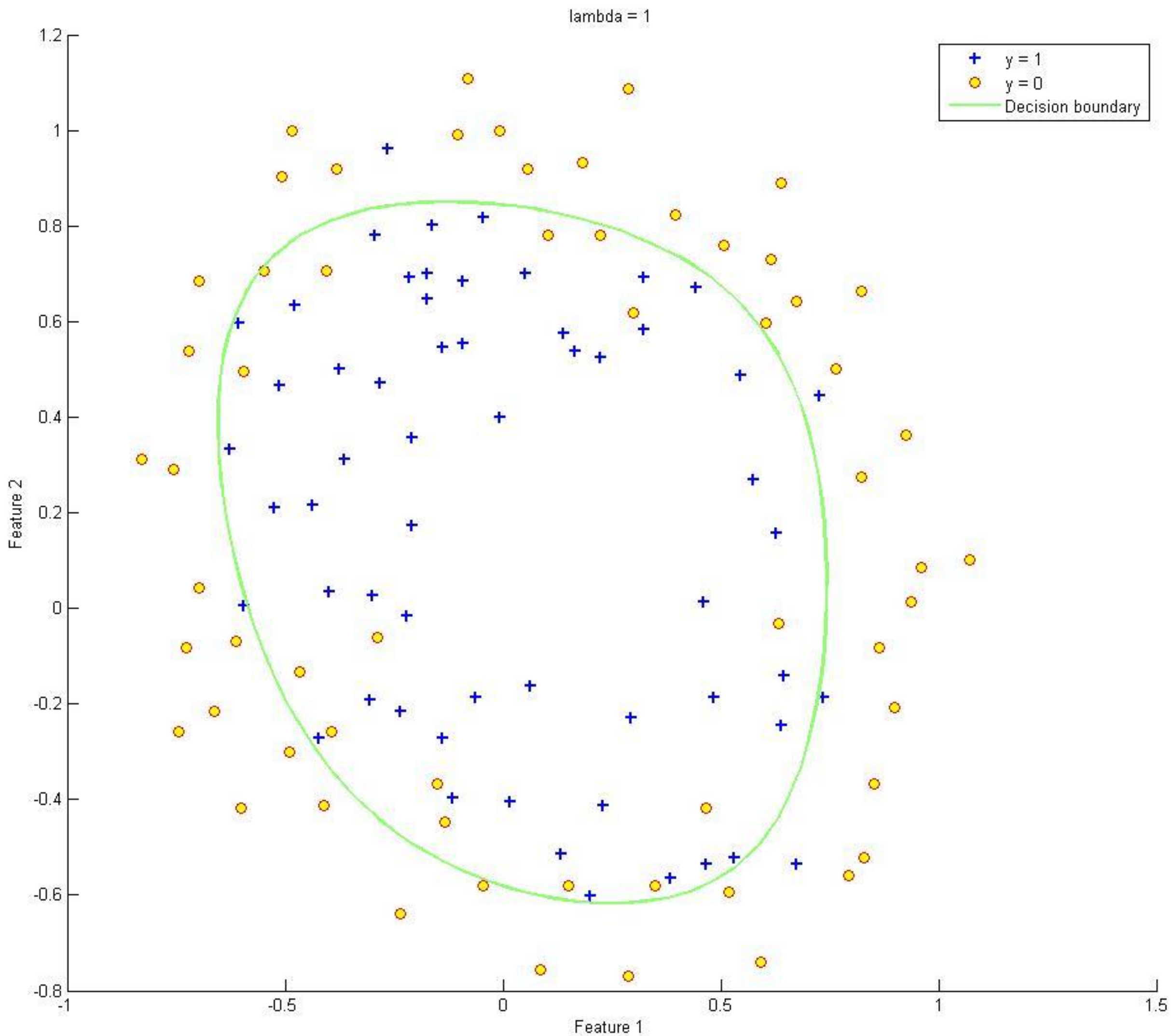


Figure and dataset derived from http://cs229.stanford.edu/ by Andrew Ng

$\lambda = 0$

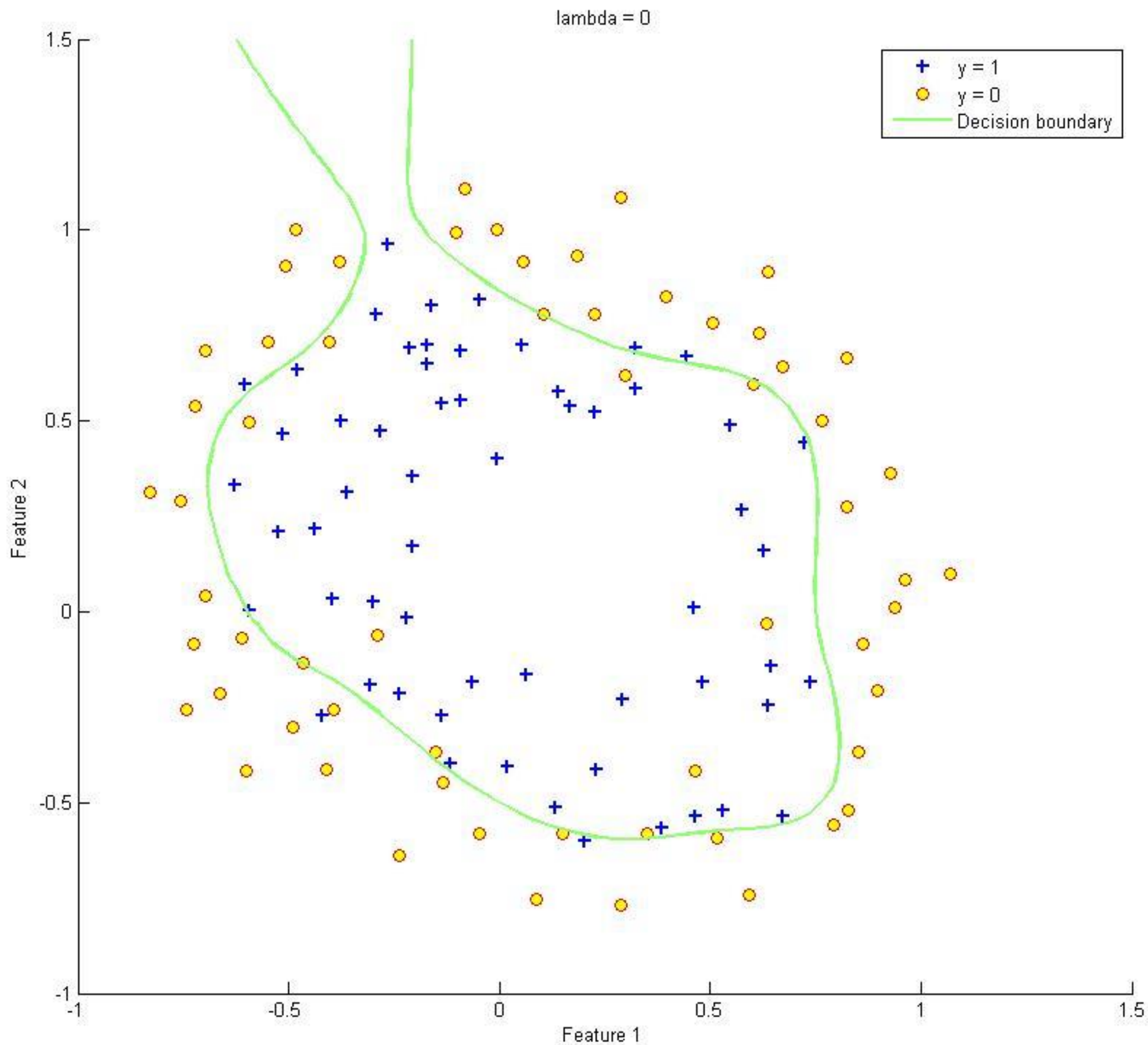


Figure and dataset derived from http://cs229.stanford.edu/ by Andrew Ng

Summary

- Linear supervised approaches
  - For regression: Linear regression
  - For classification: Logistic regression