CS395: Selected CS1 (Introduction to Machine Learning)

Associate Prof. Wessam El-Behaidy

Fall 2021

References:

https://www.coursera.org/learn/machine-learning (Andrew Ng)

Machine learning A to Z: **Kirill Eremenko ©superdatascience**

Normal Equations

Better values for θ

- To solve for θ analytically → normal equation
- $\theta \in \mathbb{R}^{n+1}$

$$J(\theta_0, \theta_1, \dots, \theta_n) = J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Minimize $J(\theta)$

$\frac{\partial}{\partial \theta_j} J(\theta)$= …=0 (set to zero)

for $every\ j$ solve for $\theta_0, \theta_1, \dots, \theta_n$

Example: m=4

x_0	$Size\ (feet)^2$ x_1	$Number\ of\ bedrooms$ x_2	$Number\ of\ floors$ x_3	$Age\ of\ home\ (years)$ x_4	$Price\ (\$1000)$ y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix} \qquad \theta = (X^T X)^{-1} X^T y$$

Features matrix $m \times (n+1)$ $\qquad$ $m - \dim vector$

m **examples** $(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$ **; n features.**

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ . \\ . \\ . \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1}$$

$m \; samples \;, n \; features$

$$X = \begin{bmatrix} \cdots (x^{(1)})^T \cdots \\ (x^{(2)})^T \\ . \\ \cdots \quad . \quad \cdots \\ . \\ \cdots (x^{(m)})^T \cdots \end{bmatrix}$$

$\boldsymbol{Design\, matrix}$

m x (n+1)

$Example\colon one\ feature$

$$x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & x_1^1 \\ 1 & x_1^2 \\ \vdots & \vdots \\ 1 & x_1^m \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ . \\ . \\ . \\ y^{(m)} \end{bmatrix}$$

$$\theta = (X^T X)^{-1} X^T y$$

Gradient Descent vs Normal Equation

Gradient Descent	**Normal Equation**
Need to choose α	No need to choose α
Needs many iterations	No need to iterate
$O\ (kn^2)$	$O\ (n^3)$
Works well when n is large	Slow if n is very large

Example 2:

age (x_1)	height in cm (x_2)	weight in kg (y)
4	89	16
9	124	28
5	103	20

What is X (design matrix) and y?

$$X = \begin{bmatrix} 1 & 4 & 89 \\ 1 & 9 & 124 \\ 1 & 5 & 103 \end{bmatrix}, \; y = \begin{bmatrix} 16 \\ 28 \\ 20 \end{bmatrix}$$

Check?

Suppose you have $m = 25$ training examples with $n = 6$ features. The normal equation is $\theta = (X^T X)^{-1} X^T y$

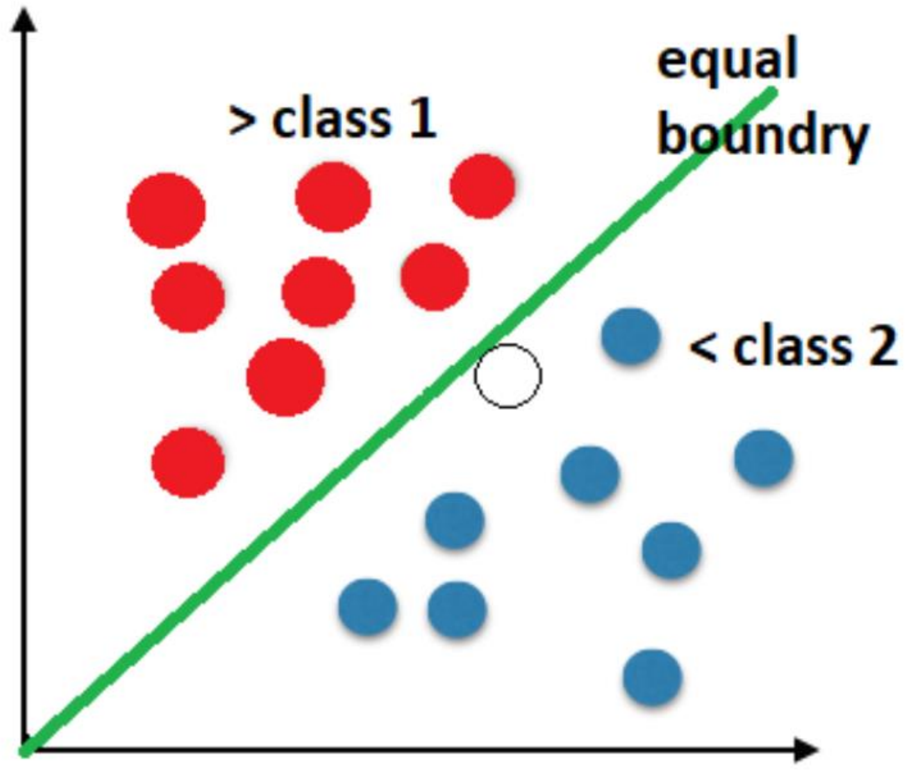
For the given values of m and n, what are the dimensions of θ, X, and y in this equation?

Classification

Logistic regression

Classification

- Discrete outcomes.
- Binary $y \in \{0,1\}$, 0 negative , 1 positive (normal / abnormal)
- Multi-class: a telescope that identifies whether an object in the night sky is a galaxy, star, or planet. $y \in \{0,1,2,3\}$



Hypothesis Representation

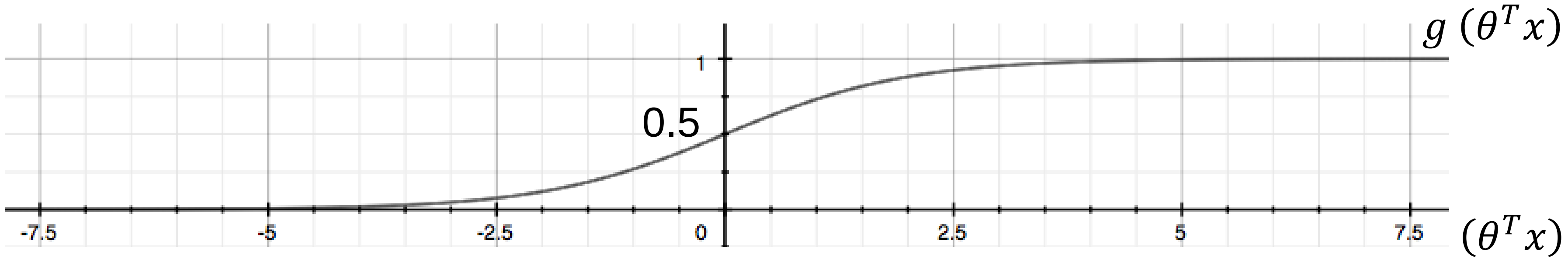
- We want our classifier to output values between 0 and 1
  - When using linear regression we did $h_\theta(x) = (\theta^T x)$
  - For classification hypothesis representation we do $h_\theta(x) = g((\theta^T x))$

Logistic regression model

$$0 \leq h_\theta(x) \leq 1$$

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

Sigmoid Function
Logistic Function



Interpretation of hypothesis Output

- $h_\theta(x)$ will give us the **probability** that our output is 1.
- For example, $h_\theta(x) = 0.7$ gives us a probability of 70% that our output is 1.
- Our probability that our prediction is 0 is just the complement of our probability that it is 1 (e.g. if probability that it is 1 is 70%, then the probability that it is 0 is 30%).

Interpretation of hypothesis Output

- $h_\theta(x)$ will give us the **probability** that our output is 1.
- For example,

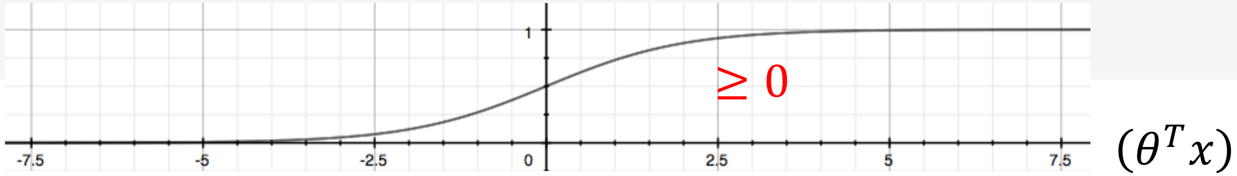
$$x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumourSize} \end{bmatrix},\ h_\theta(x) = 0.7 \ldots\ y = 1$$

70% chance of a tumor being malignant.

$$h_\theta(x) = P(y = 1|x;\theta) = 1 - P(y = 0|x;\theta)$$

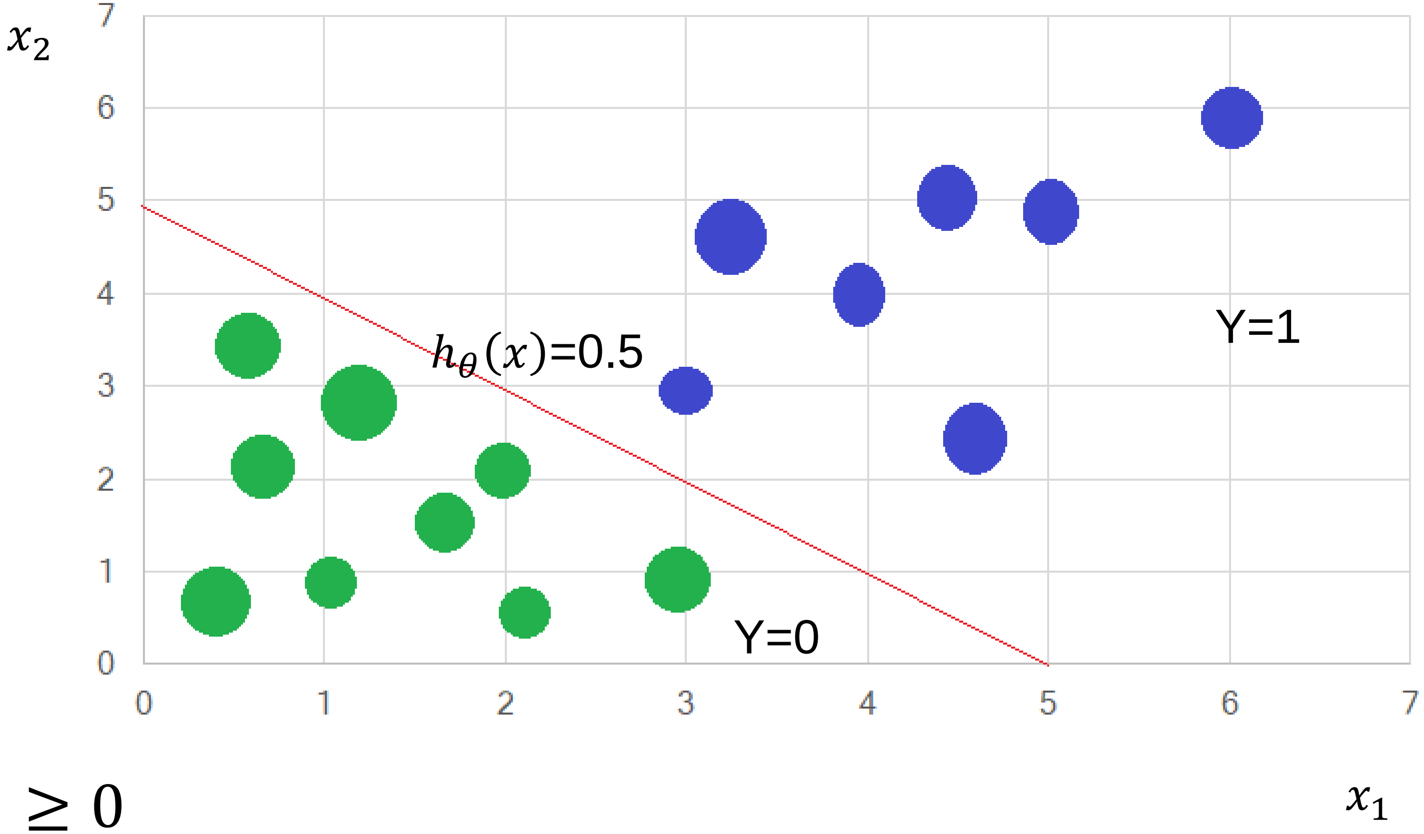
Probability that $y = 1$, given x, parameterized by θ

Decision Boundary



$$h_\theta(x) = g\,(\theta^T x) = \; g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$
$$= g(-5 + 1x_1 + 1x_2)$$

$$\theta = \begin{bmatrix} -5 \\ 1 \\ 1 \end{bmatrix}$$



$$Predict\ y = 1\ if\ \ -5 + x_1 + x_2 \geq 0$$
$$x_1 + x_2 \geq 5$$

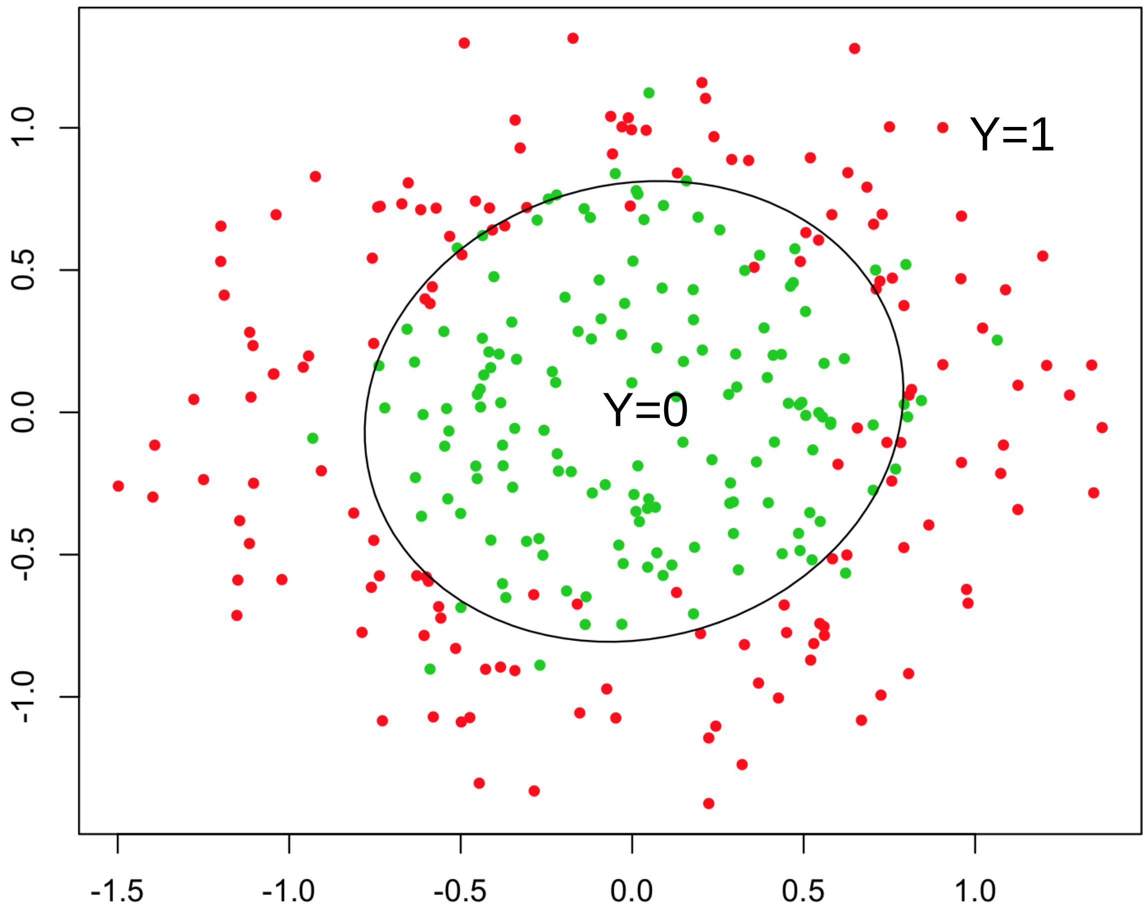
Non-linear decision boundaries

$$h_\theta(x) = g\,(\theta^T x) = \; g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$= \; -0.6 \; + \; 0\, x_1 \; + \; 0\, x_2 \; + \; 1\, x_1^2 \; + 1\, x_2^2$$

$$y = 1, if \; - 0.6 + x_1^2 \; + \; x_2^2 \geq 0$$

$$x_1^2 \; + \; x_2^2 \geq 0.6$$



How to choose parameter θ

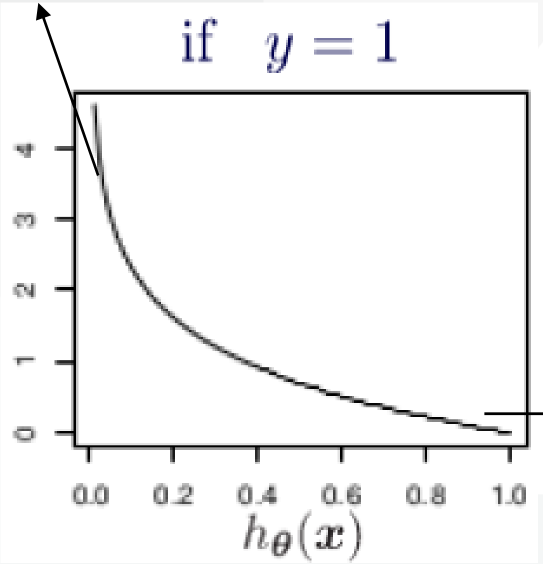
Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

$$h_\theta(x) = g\,(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

$$x \in \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ . \\ . \\ x_n \end{bmatrix}$$

$x_0 = 1, y \in \{0,1\}$

$Cost \rightarrow \infty\,, y \rightarrow 0$

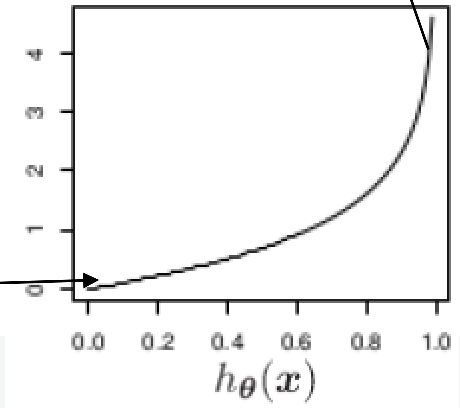


$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

$Cost = 0, y = 1$

$Cost \rightarrow \infty\,, y \rightarrow 1$

$if\ y = 0$



$Cost = 0, y = 0$

Simplified Cost Function and Gradient Descent

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

$$\equiv \ \mathrm{Cost}(h_\theta(x), y) = -y \ \log(h_\theta(x)) - (1 - y)\log(1 - h_\theta(x))$$

when y is equal to 1, then the second term will be zero and will not affect the result. If y is equal to 0, then the first term will be zero and will not affect the result.

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

$$min_\theta \ J(\theta)$$

Repeat $\{$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$\}$

$\boldsymbol{repeat\ until\ convergence}$: $\{$

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$$

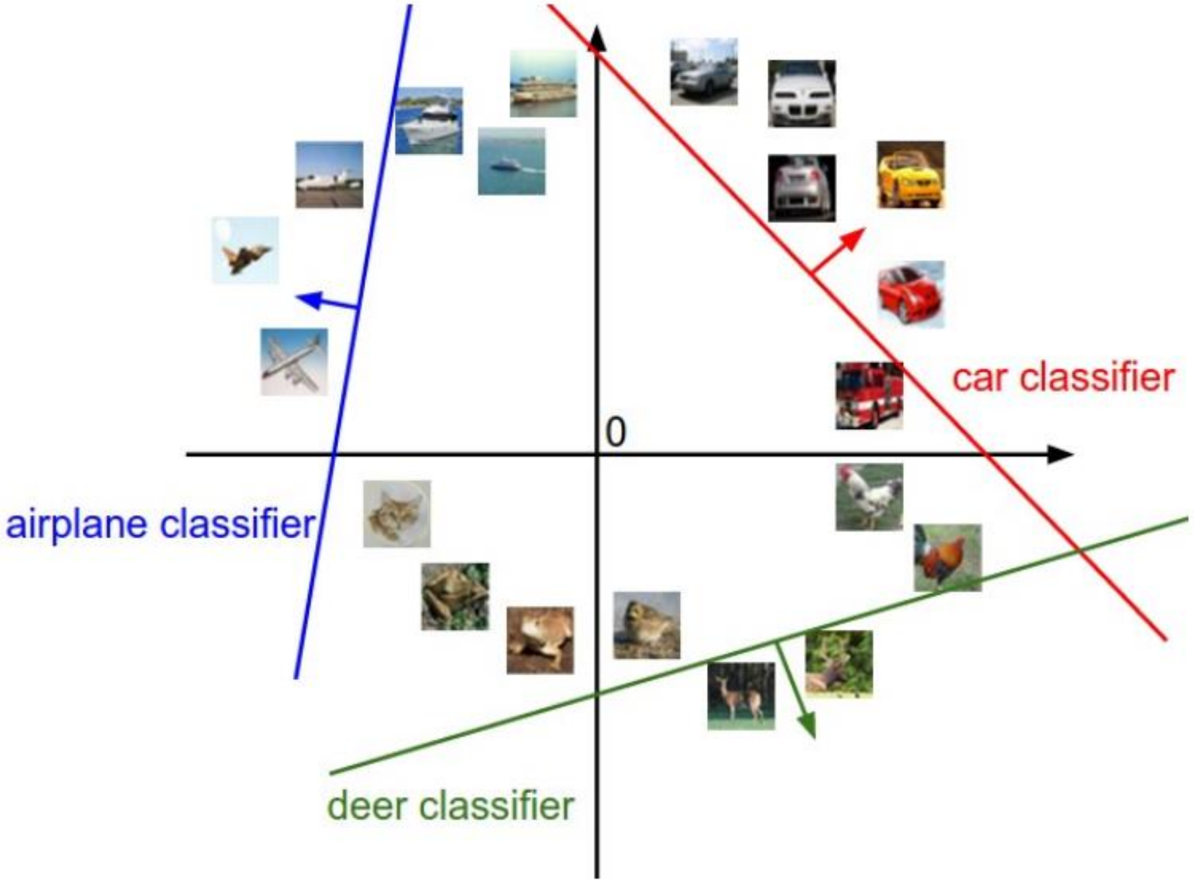
, simultaneously updates θ_j for every $j = 0,1,2, \dots, n$

$\}$

$$where, h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Multiclass Classification

- Train a logistic regression classifier $h_\theta(x)$ for each class



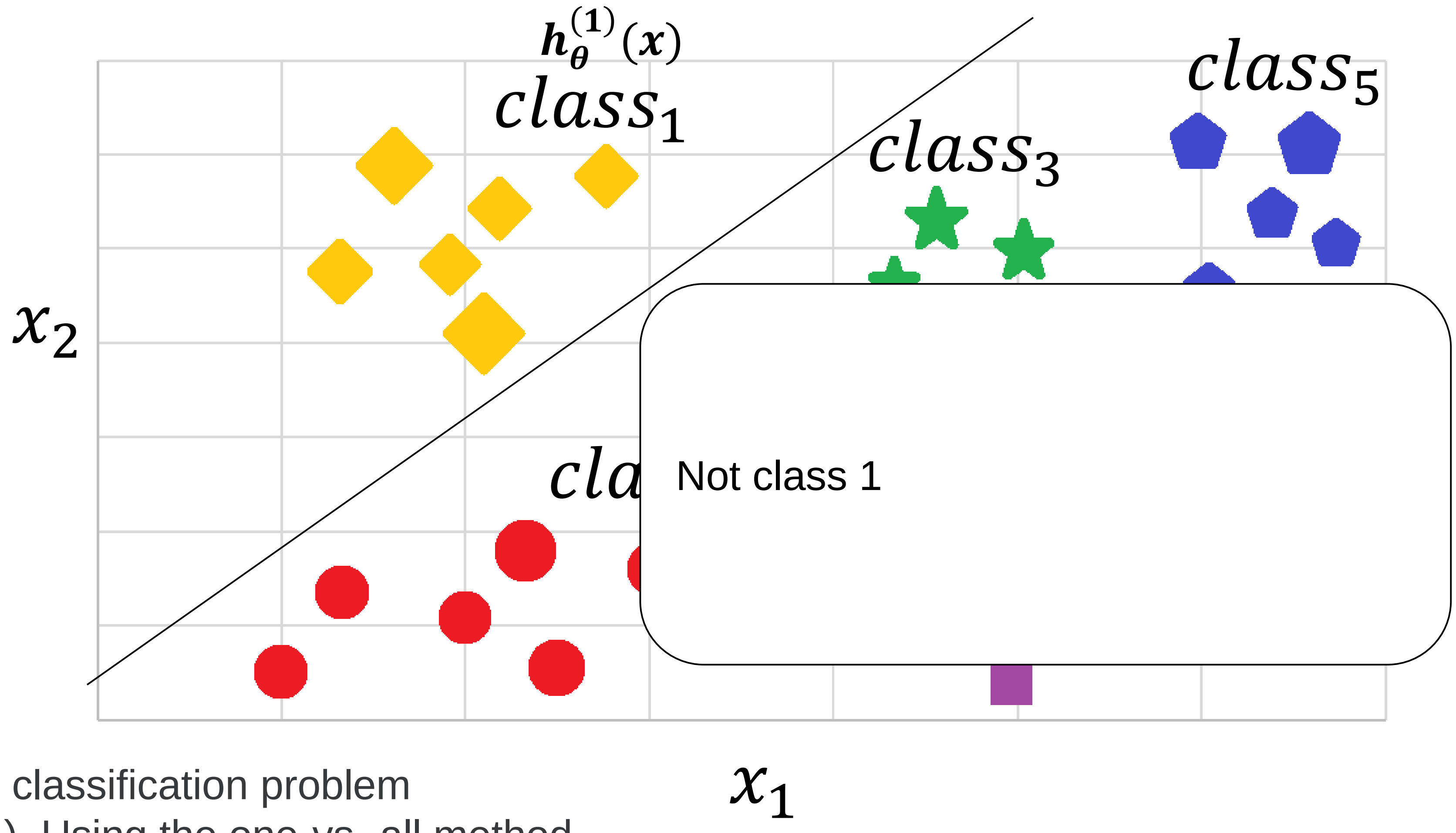
Multiclass Classification (one-vs-all)

$$h_\theta^{(i)} = P(y = i | x; \theta)$$

$$i = 1, 2, 3 \dots$$

Pick the class i that maximize

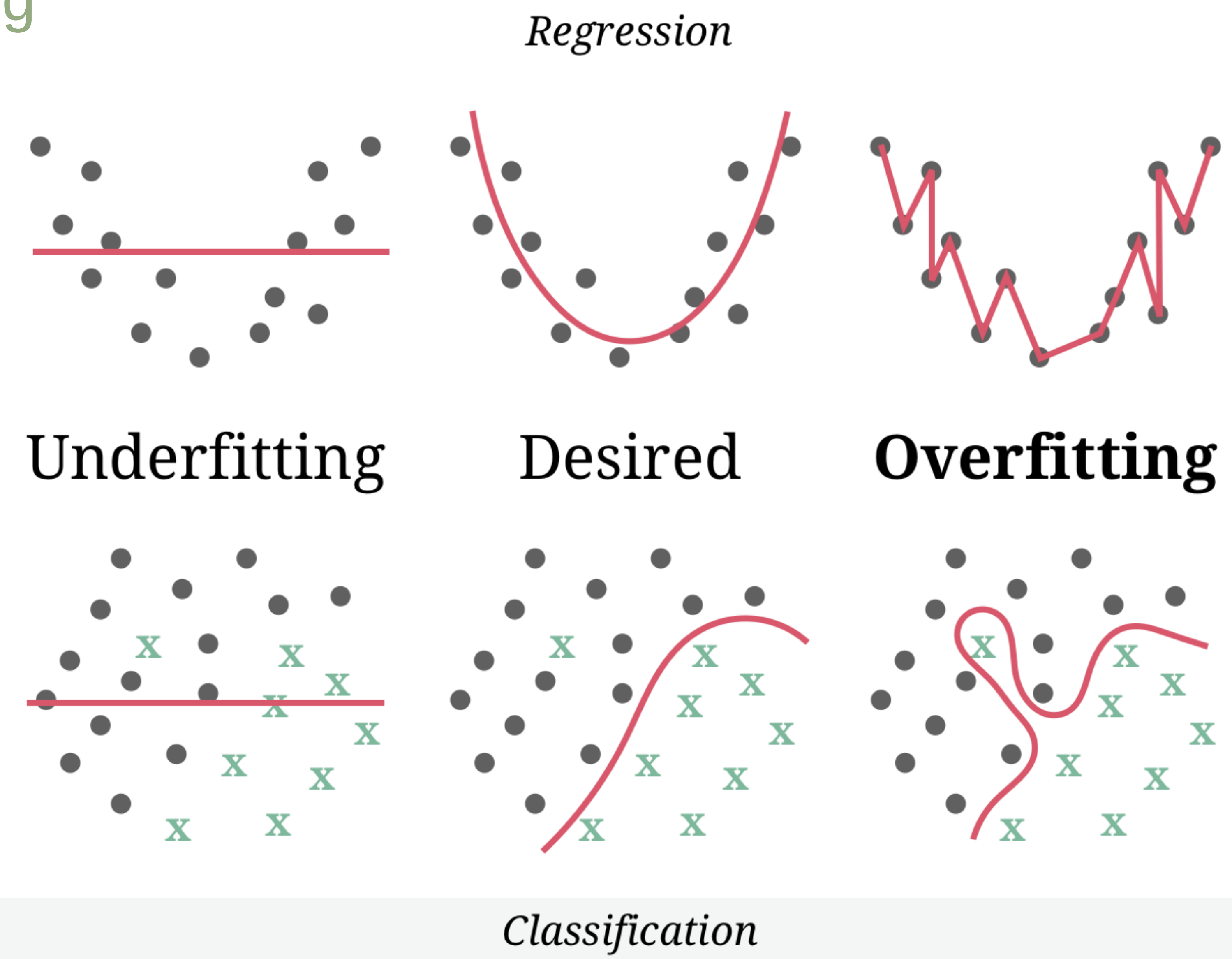
$$prediction = max_i \,(h_\theta^{(i)})$$



Suppose you have a multi-class classification problem with k classes (so $y \in \{1, 2 \cdots, k\}$). Using the one-vs.-all method, how many different logistic regression classifiers will you end up training?

The Problem of Overfitting

- Underfitting, or high bias, is when the form of our hypothesis function h maps poorly to the trend of the data. It is usually caused by a function that is too simple or uses too few features.
- At the other extreme, overfitting, or high variance, is caused by a hypothesis function that fits the available data but does not generalize well to predict new data. It is usually caused by a complicated function that creates a lot of unnecessary curves and angles unrelated to the data.



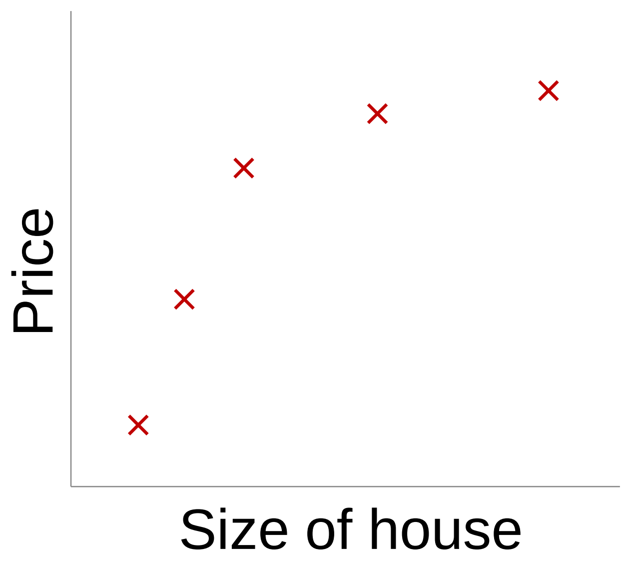
Addressing overfitting

There are two main options to address the issue of overfitting:

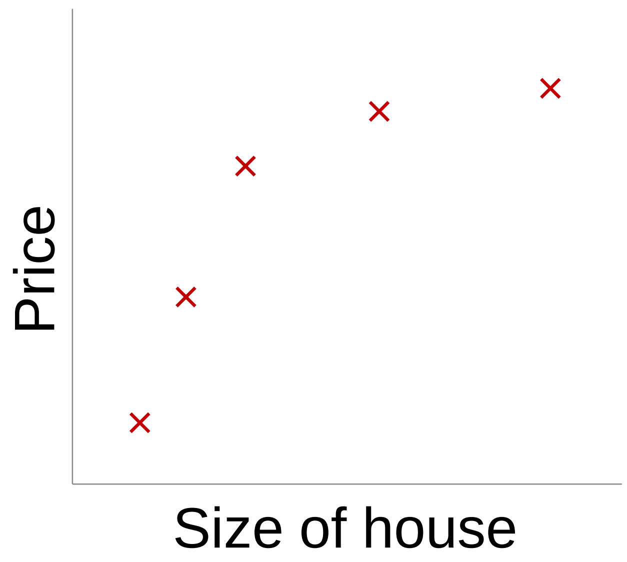
1) Reduce the number of features:
   - Manually select which features to keep.
   - Use a model selection algorithm.

2) Regularization
   - Keep all the features, but reduce the magnitude of parameters θ_j.
   - Regularization works well when we have a lot of slightly useful features.

Regularization Intuition



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



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

Suppose we penalize and make θ_3, θ_4 **<u>really small</u>**.

Small values for parameters $\theta_0, \theta_1, \ldots, \theta_n$:

- simpler hypothesis
- less prone to overfitting

Regularization for linear Regression

- Simpler hypothesis , small values of $\theta_{1,}\ \theta_{2,}\ldots\ \theta_n$.

choose θ to min

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

$$\min_{\theta} J(\theta)$$

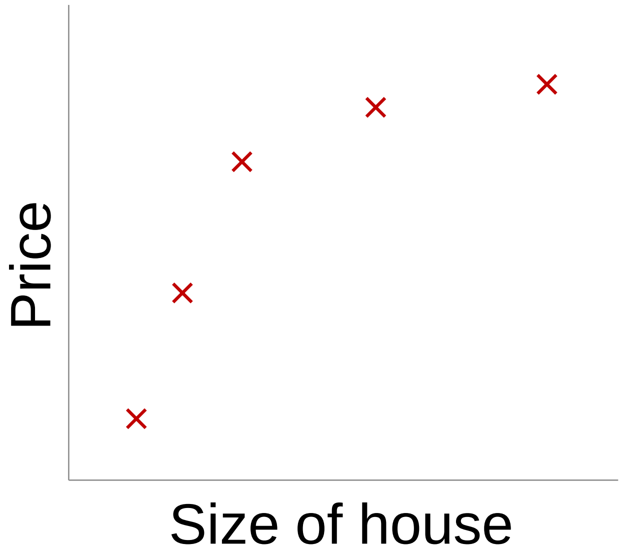
The λ, or lambda, is the **regularization parameter**.
It determines how much the costs of our theta parameters are inflated.

If λis chosen to be too large, it may smooth out the function too much and cause underfitting. ?? why

In regularized linear regression, we choose θ to minimize

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

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



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

Thanks