HALMSTAD UNIVERSITY

Learning Systems (DT8008)

Classification

Dr. Mohamed-Rafik Bouguelia
mohbou@hh.se

Halmstad University

Classification

- The variable y that you want to predict (the output variable) is discrete.

- Examples (with two classes)
  - Email: Spam / Not Spam?
  - Online Transactions: Fraudulent (Yes/No)?
  - Tumor: Malignant / Benign
    - $y \in \{0, 1\}$ 0: "Negative Class" (e.g., benign tumor)
      1: "Positive Class" (e.g. malignant tumor)

- We will first start talking about **binary classification** (with two classes).
- Then, we will talk more about **multi-class classification** (with more than two classes), $y \in \{0, 1, 2, 3, \dots, c\}$

HALMSTAD UNIVERSITY

Some Applications of Classification

Some applications of classification

Example: Spam Filter

- Input: email
- Output: spam/ham
- Setup:
  - Get a large collection of example emails, each labeled "spam" or "ham"
  - Note: someone has to hand label all this data!
  - Want to learn to predict labels of new, future emails
- Features: The attributes used to make the ham / spam decision
  - Words: FREE!
  - Text Patterns: $dd, CAPS
  - Non-text: SenderInContacts
  - …

Dear Sir.

First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial and top secret. …

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES FOR ONLY $99

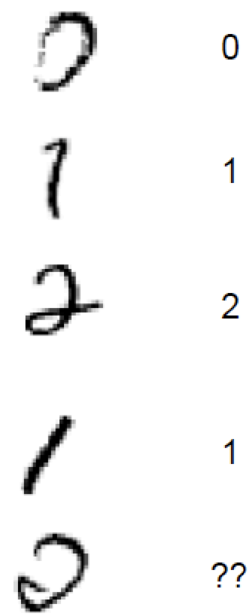
Ok, Iknow this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Some applications of classification

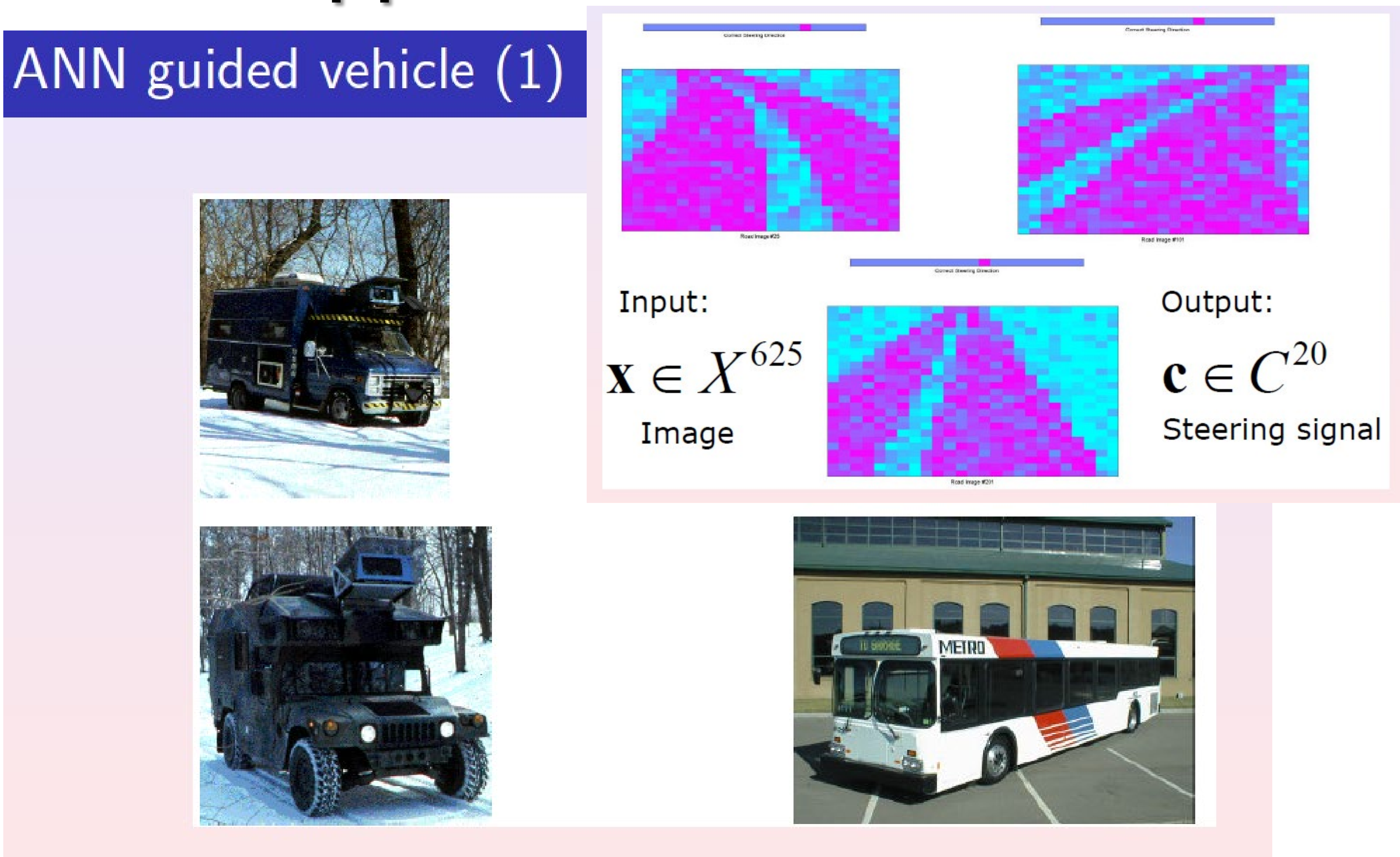
Example: Digit Recognition

- Input: images / pixel grids
- Output: a digit 0-9
- Setup:
  - Get a large collection of example images, each labeled with a digit
  - Note: someone has to hand label all this data!
  - Want to learn to predict labels of new, future digit images

- Features: The attributes used to make the digit decision
  - Pixels: (6,8)=ON
  - Shape Patterns: NumComponents, AspectRatio, NumLoops
  - …



Some applications of classification



Some applications of classification

Classify the Lego pieces into red, blue, and yellow.

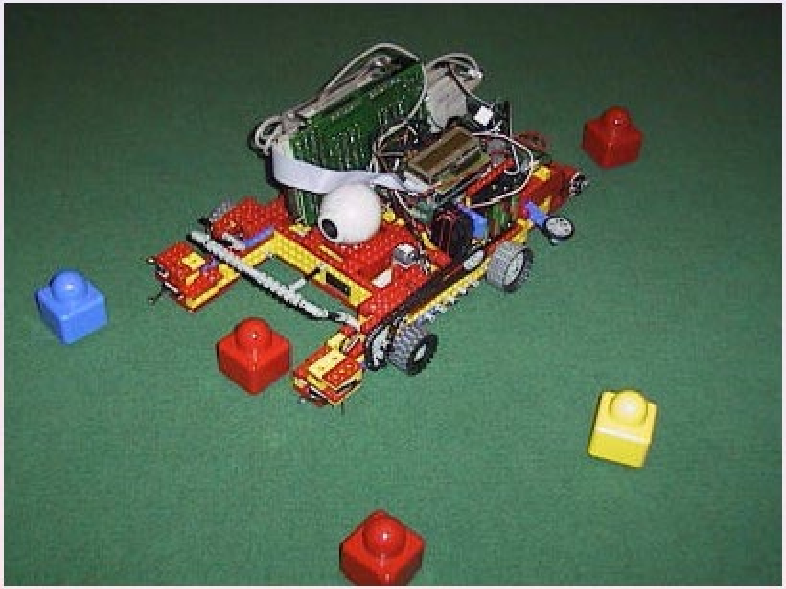
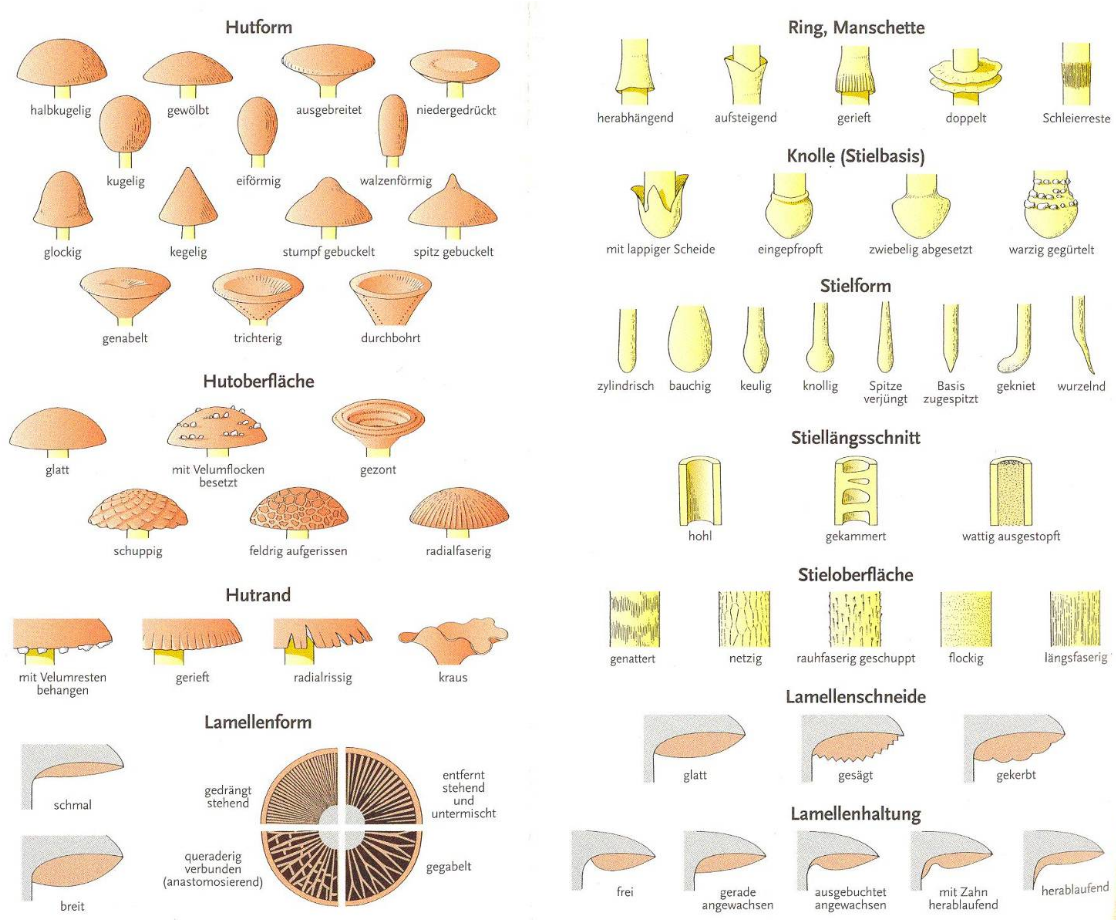


Figure: Robot and Lego pieces.

Some applications of classification

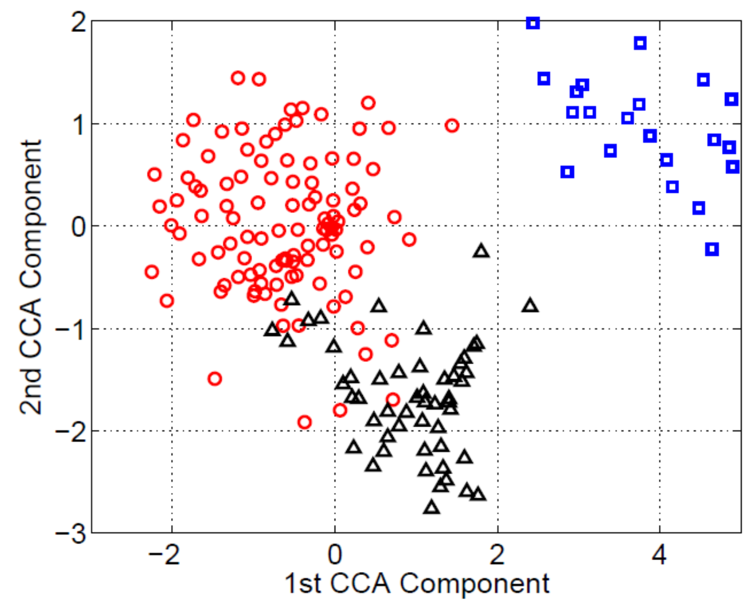


Edible or poisonous ?

Some applications of classification

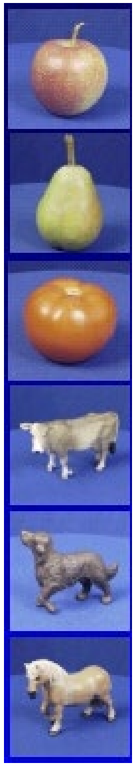
- e.g. Laryngeal disease diagnostics

- Features / Attributes:
  - Age
  - Subjectively estimated illness duration (months)
  - Education (five grades)
  - Average duration of intensive speech use (hours/day)
  - Number of days of intensive speech use (days/week)
  - Smoking (Yes/No)
  - Smoked cigarettes/day
  - Smoking duration (years);
  - Subjective voice function assessment by the patient
  - Maximal tonality duration for "aaaaaa" (sec)
  - Functional voice index (F);
  - Emotional condition index (E);
  - Physical condition index (P);
  - Voice deficiency index
  - …



Some applications of classification

Training set (labels known)



apple

pear

tomato

cow

dog

horse

Test set (labels unknown)

Linear Classification with Logistic Regression

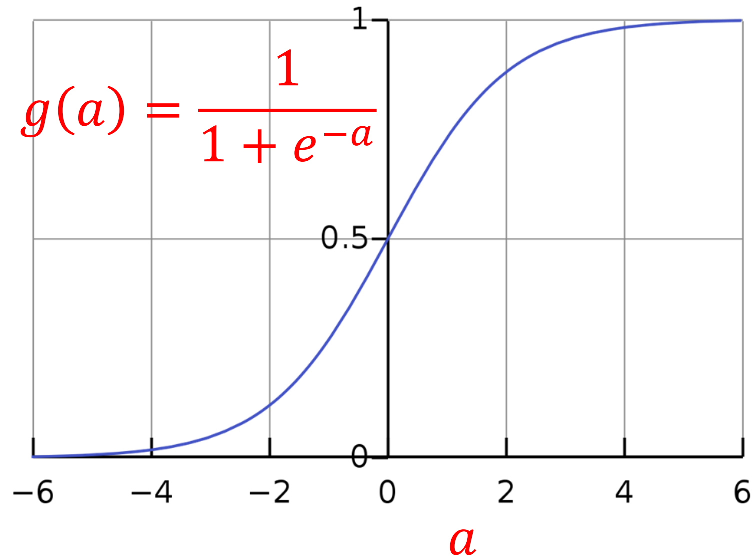
Logistic Regression

- This is a classification method (don't get confused by the name).

- In a binary classification, we want $y = 0$ or $y = 1$
  - but, if you use a simple linear regression model $h_\theta(x) = \theta^T x$, then $h_\theta(x)$ can be > 1 or < 0

- The logistic regression model is defined so that $0 \leq h_\theta(x) \leq 1$
  - $h_\theta(x) = g(\theta^T x)$, where $g(.)$ is the **sigmoid function** (or logistic function).
  - Sigmoid function: $g(a) = \frac{1}{1+e^{-a}}$

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



Logistic Regression

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

- Interpretation of the hypothesis output $h_\theta(x)$

  $h_\theta(x)$ = estimated probability that $y = 1$ on input x

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$h_\theta(x) = 0.7$ ➔ $y = 1$

70% chance of tumor being malignant

$$\left. h_\theta(x) = P(y = 1 \mid x; \theta) = 0.7 \right\}$$

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

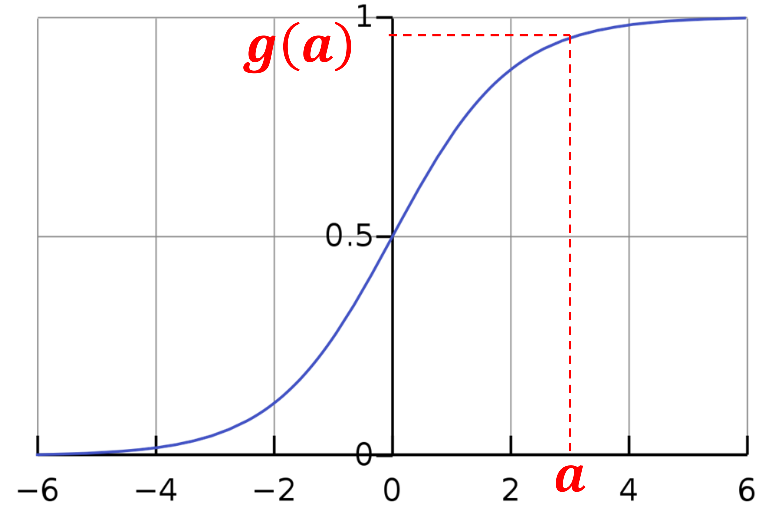
Note: since $y \in \{0,1\}$, $P(y = 1 \mid x; \theta) + P(y = 0 \mid x; \theta) = 1$

Logistic Regression

Linear decision boundary

$$h_\theta(x) = g(\theta^T x) = P(y = 1 \mid x, \theta)$$

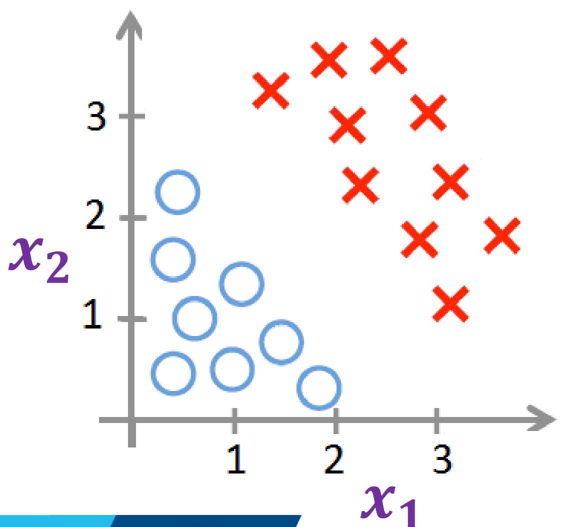
$$g(a) = \frac{1}{1 + e^{-a}}$$



if $h_\theta(x) \geq 0.5$ then we predict class $y = 1$
if $h_\theta(x) < 0.5$ then we predict class $y = 0$

same as

if $\theta^T x \geq 0$ then we predict class $y = 1$
if $\theta^T x < 0$ then we predict class $y = 0$



- Example of a linear decision boundary:

$$\begin{aligned} h_\theta(x) &= g(\theta^T x) \\ &= g(\theta_0 + \theta_1 x_1 + \theta_2 x_2) \\ &= g(-3 + x_1 + x_2) \end{aligned}$$

Predict $y = 1$ if $-3 + x_1 + x_2 \geq 0$

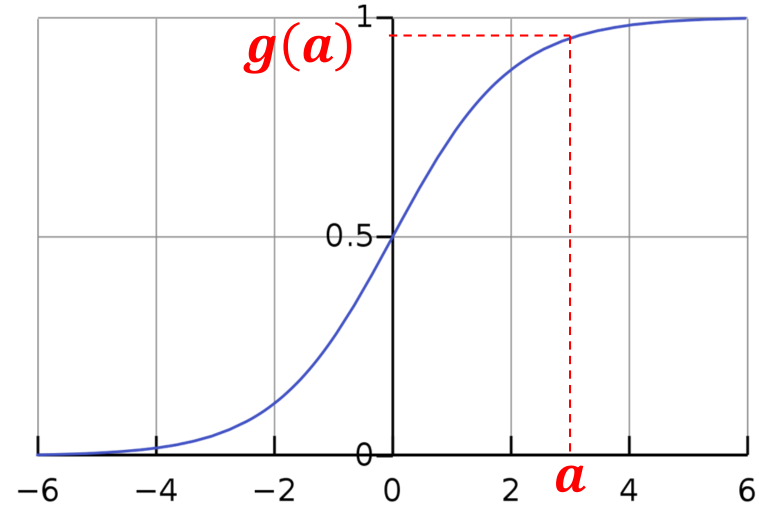
$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

Logistic Regression

Linear decision boundary

$$h_\theta(x) = g(\theta^T x) = P(y = 1 \mid x, \theta)$$

$$g(a) = \frac{1}{1 + e^{-a}}$$



if $h_\theta(x) \geq 0.5$ then we predict class $y = 1$
if $h_\theta(x) < 0.5$ then we predict class $y = 0$

same as

if $\theta^T x \geq 0$ then we predict class $y = 1$
if $\theta^T x < 0$ then we predict class $y = 0$

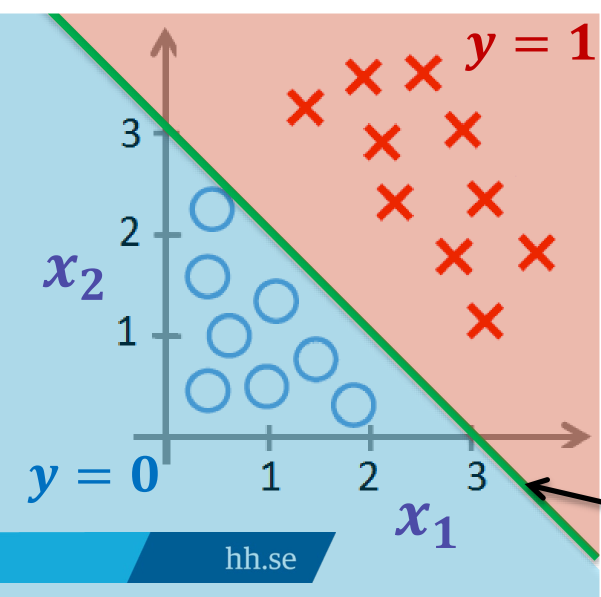
- Example of a linear decision boundary:

$$\begin{aligned} h_\theta(x) &= g(\theta^T x) \\ &= g(\theta_0 + \theta_1 x_1 + \theta_2 x_2) \\ &= g(-3 + x_1 + x_2) \end{aligned}$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

Predict $y = 1$ if $-3 + x_1 + x_2 \geq 0$

- for all regions where $x_1 + x_2 \geq 3$, this will predict $y = 1$
- for all regions where $x_1 + x_2 < 3$, this will predict $y = 0$
- The **decision boundary** is $x_1 + x_2 = 3$



decision boundary

Defining the Cost Function for Logistic Regression

Logistic Regression – Error function

- Training dataset $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(n)}, y^{(n)})\}$
- $x = \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_d \end{bmatrix} \in \mathbb{R}^{d+1}, \quad x_0 = 1, \quad y \in \{0,1\}$

- $h_\theta(x) = 1 \,/\, (1 + e^{-\theta^T x})$
- How de we choose the parameters θ ?
  - By minimizing some error (cost) function

$$E(\theta) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \left[h_\theta(x^{(i)}) - y^{(i)} \right]^2$$
$$= \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \left[\frac{1}{1 + e^{-\theta^T x^{(i)}}} - y^{(i)} \right]^2$$

If our cost function is defined this way, it will be **Non-Convex** !

Several local minimums. GD is not guaranteed to converge to the global minimum.

Logistic Regression – Error function

Instead, we use the following **convex** cost function:

$$E(\theta) = \frac{1}{n}\sum_{i=1}^{n} \text{cost}\left(h_\theta(x^{(i)}), y^{(i)}\right)$$

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

This will give us a **convex** optimization problem when we want to minimize $E(\theta)$

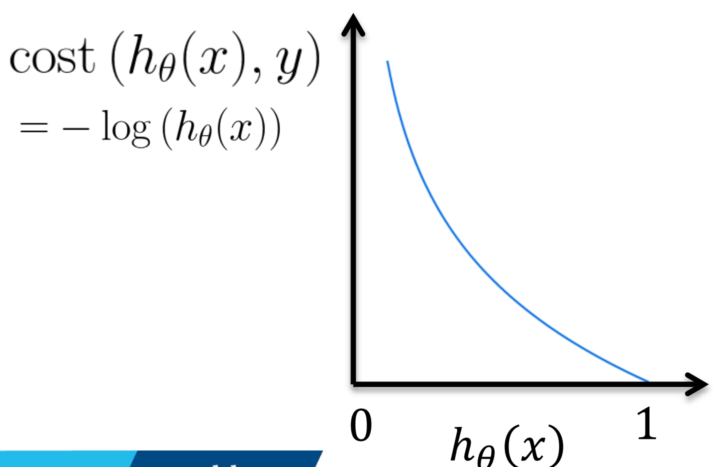
Logistic Regression – Error function

Instead, we use the following error function:

$$E(\theta) = \frac{1}{n}\sum_{i=1}^{n} \text{cost}\left(h_\theta(x^{(i)}), y^{(i)}\right)$$

$$\text{cost}\left(h_\theta(x), y\right) = \begin{cases} \boxed{-\log\left(h_\theta(x)\right) \quad \text{if } y = 1} \\ -\log\left(1 - h_\theta(x)\right) \quad \text{if } y = 0 \end{cases}$$

If $y = 1$



In the case where $y = 1$

- When $h_\theta(x)$ is closer to 1, the $cost(h_\theta(x), y)$ is closer to 0.
- The $cost(h_\theta(x), y) = 0$ if $h_\theta(x) = 1$
- As $h_\theta(x) \rightarrow 0$, the $cost \rightarrow \infty$
- Captures the intuition that if $h_\theta(x) = 0$ (i.e. $P(y = 1 \mid x, \theta) = 0$), but $y = 1$, then we will penalize the learning algorithm by a very large cost.

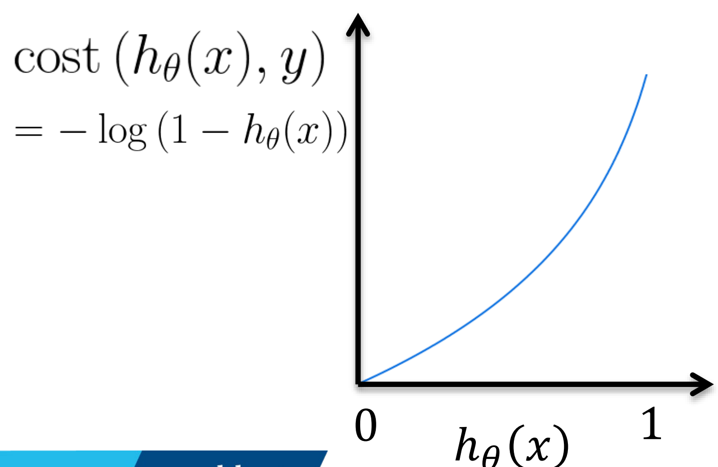
Logistic Regression – Error function

Instead, we use the following error function:

$$E(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{cost}\left(h_\theta(x^{(i)}), y^{(i)}\right)$$

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

If $y = 0$



In the case where $y = 0$

- When $h_\theta(x)$ is closer to 0, the $cost(h_\theta(x), y)$ is closer to 0.
- The $cost(h_\theta(x), y) = 0$ if $h_\theta(x) = 0$
- As $\mathrm{h}_\theta(x) \rightarrow 1$, the $cost \rightarrow \infty$
- Captures the intuition that if $h_\theta(x) = 1$ (i.e. $P(y = 0 \mid x, \theta) = 0$), but $y = 0$, then we will penalize the learning algorithm by a very large cost.

Logistic Regression – Error function

$$E(\theta) = \frac{1}{n}\sum_{i=1}^{n} \text{cost}\left(h_\theta(x^{(i)}), y^{(i)}\right)$$

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

$x \in \mathbb{R}^d$

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

Simpler way to write the error function:

$$\text{cost}\left(h_\theta(x), y\right) = -y \ \log\left(h_\theta(x)\right) - (1 - y) \ \log\left(1 - h_\theta(x)\right)$$

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

- To find the best parameters θ:

$$\min_\theta E(\theta)$$

- To make a prediction given new x:

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

- $h_\theta(x)$ is interpreted as $P(y = 1 \mid x; \theta)$

Gradient of the Cost Function

Derivative of the cost function

$$E(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y \underbrace{\log\left(h_\theta(x)\right)}_{F} + (1-y)\underbrace{\log\left(1-h_\theta(x)\right)}_{Q}\right]$$

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

To use gradient descent, we need to know $\frac{\partial E}{\partial \theta_j}$ for $j = 1, \dots, p$

$$\frac{\partial E}{\partial \theta_j} = -\frac{1}{n}\sum_{i=1}^{n}\left[y\,\frac{\partial F}{\partial \theta_j} + (1-y)\,\frac{\partial Q}{\partial \theta_j}\right]$$

$$\frac{\partial F}{\partial \theta_j} = \frac{\partial F}{\partial h_\theta(x)} \cdot \frac{\partial h_\theta(x)}{\partial \theta_j}$$

$$\frac{\partial Q}{\partial \theta_j} = \frac{\partial Q}{\partial (1-h_\theta(x))} \cdot \frac{\partial (1-h_\theta(x))}{\partial \theta_j}$$

Derivative of the cost function

$$E(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y \underbrace{\log\left(h_\theta(x)\right)}_{F} + (1-y) \underbrace{\log\left(1-h_\theta(x)\right)}_{Q}\right]$$

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

To use gradient descent, we need to know $\frac{\partial E}{\partial \theta_j}$ for $j = 1, \dots, p$

$$\frac{\partial E}{\partial \theta_j} = -\frac{1}{n}\sum_{i=1}^{n}\left[y \frac{\partial F}{\partial \theta_j} + (1-y) \frac{\partial Q}{\partial \theta_j}\right]$$

$$\frac{\partial F}{\partial \theta_j} = \frac{\partial F}{\partial h_\theta(x)} \cdot \frac{\partial h_\theta(x)}{\partial \theta_j}$$

$$\frac{\partial Q}{\partial \theta_j} = \frac{\partial Q}{\partial (1-h_\theta(x))} \cdot \frac{\partial (1-h_\theta(x))}{\partial \theta_j}$$

Derivative of the cost function

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

$$\frac{\partial h_\theta(x)}{\partial \theta_j} = \frac{-1}{(1 + e^{-\theta^T x})^2} \quad [e^{-\theta^T x} \quad (-x_j)]$$

$$= \frac{1}{1 + e^{-\theta^T x}} \quad \frac{e^{-\theta^T x}}{1 + e^{-\theta^T x}} \quad x_j$$

$$= h_\theta(x) \quad (1 - h_\theta(x)) \quad x_j$$

$$\boxed{\frac{\partial h_\theta(x)}{\partial \theta_j} = h_\theta(x) \quad (1 - h_\theta(x)) \quad x_j}$$

Derivative of the cost function

$$E(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y \underbrace{\log\left(h_\theta(x)\right)}_{F} + (1-y)\underbrace{\log\left(1-h_\theta(x)\right)}_{Q}\right]$$

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

To use gradient descent, we need to know $\frac{\partial E}{\partial \theta_j}$ for $j = 1, \dots, p$

$$\frac{\partial E}{\partial \theta_j} = -\frac{1}{n}\sum_{i=1}^{n}\left[y\,\frac{\partial F}{\partial \theta_j} + (1-y)\,\frac{\partial Q}{\partial \theta_j}\right]$$

$$\frac{\partial F}{\partial \theta_j} = \frac{\partial F}{\partial h_\theta(x)} \cdot \frac{\partial h_\theta(x)}{\partial \theta_j}$$

$$\frac{\partial Q}{\partial \theta_j} = \frac{\partial Q}{\partial(1-h_\theta(x))} \cdot \frac{\partial(1-h_\theta(x))}{\partial \theta_j}$$

Derivative of the cost function

$$E(\theta) = -\frac{1}{n}\sum_{i=1}^{n}[y \underbrace{\log(h_\theta(x))}_{F} + (1-y)\underbrace{\log(1-h_\theta(x))}_{Q}]$$

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

$$\frac{\partial F}{\partial \theta_j} = \frac{\partial F}{\partial h_\theta(x)} \frac{\partial h_\theta(x)}{\partial \theta_j}$$

$$= \frac{1}{h_\theta(x)} \boxed{h_\theta(x) \ (1-h_\theta(x)) \ x_j}$$

$$= \boxed{(1-h_\theta(x)) \ x_j}$$

$$\frac{\partial Q}{\partial \theta_j} = \frac{\partial Q}{\partial (1-h_\theta(x))} \cdot \frac{\partial (1-h_\theta(x))}{\partial \theta_j}$$

$$= \frac{1}{(1-h_\theta(x))} [-\boxed{h_\theta(x) \ (1-h_\theta(x)) \ x_j}]$$

$$= \boxed{-h_\theta(x) \ x_j}$$

$$\frac{\partial E}{\partial \theta_j} = -\frac{1}{n}\sum_{i=1}^{n}\left[y \frac{\partial F}{\partial \theta_j} + (1-y)\frac{\partial Q}{\partial \theta_j}\right] = -\frac{1}{n}\sum_{i=1}^{n}[y \boxed{(1-h_\theta(x)) \ x_j} + (1-y)\boxed{(-h_\theta(x) \ x_j)}]$$

$$= \boxed{\frac{1}{n}\sum_{i=1}^{n}[(h_\theta(x)-y) \ x_j]}$$

Looks identical to linear regression

Gradient Descent for the Logistic Regression Classifier

Gradient descent algorithm

Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} \left[h_\theta(x^{(i)}) - y^{(i)} \right] x_j^{(i)}$$

}

Simultaneously update all θ_j for $j = 0, \dots, d$

- Looks identical to linear regression!
- But here in logistic regression $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ instead of $h_\theta(x) = \theta^T x$ which was used in linear regression.

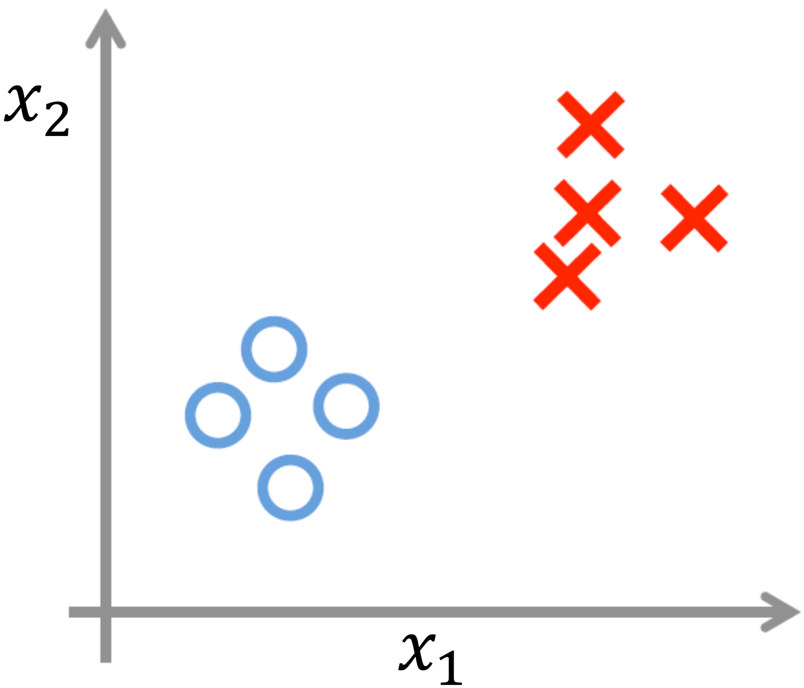
Can We Use Logistic Regression for Multi-class Classification?

Multi-class classification

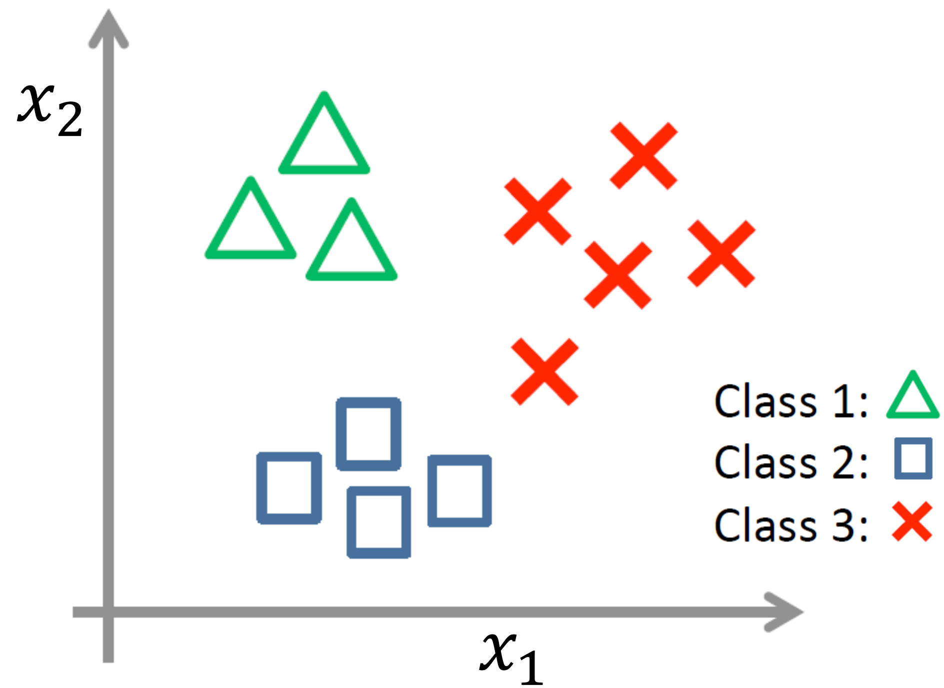
- Examples of multi-class classification applications
  - Activity recognition in smart homes:
    - Sleeping, Cooking, Taking Lunch, Watching TV …
  - Medical diagrams:
    - Cold, Flu, Not ill, …
  - Email classification/folding/tagging
    - Work, Friends, Family, Hobby, …
  - Weather:
    - Sunny, Cloudy, Rainy, Snow

Multi-class classification

Binary classification:

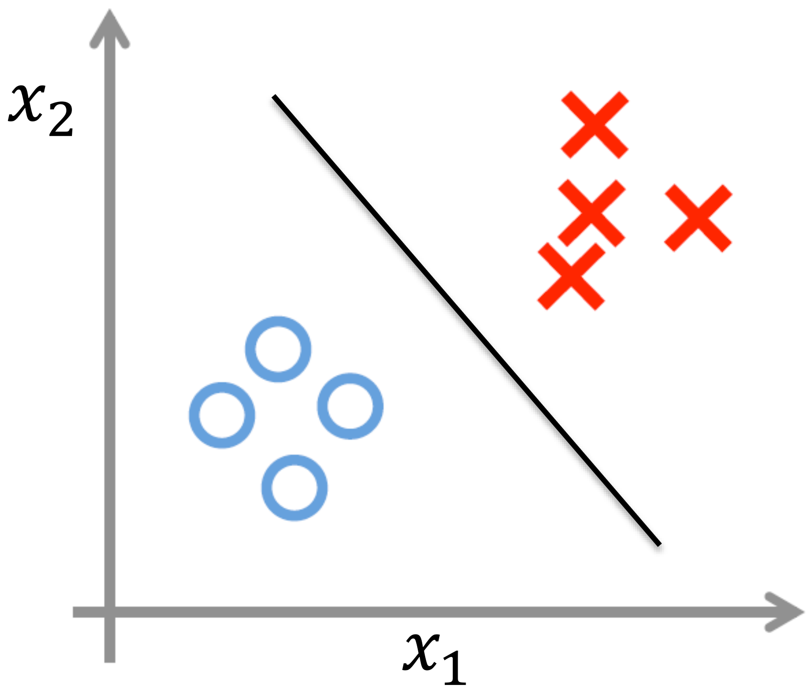


Multi-class classification:



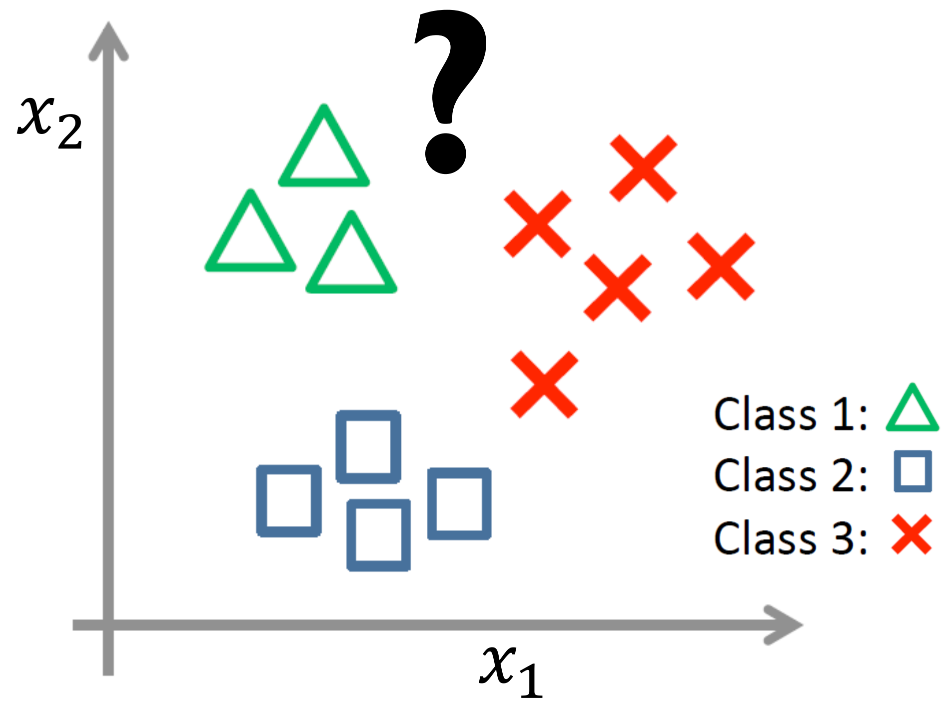
Multi-class classification

Binary classification:

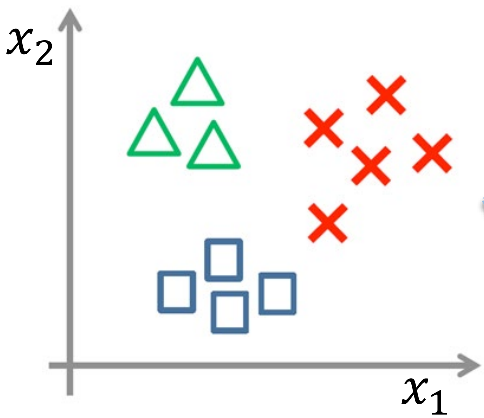


Multi-class classification:

How do we do in multi-class classification ?



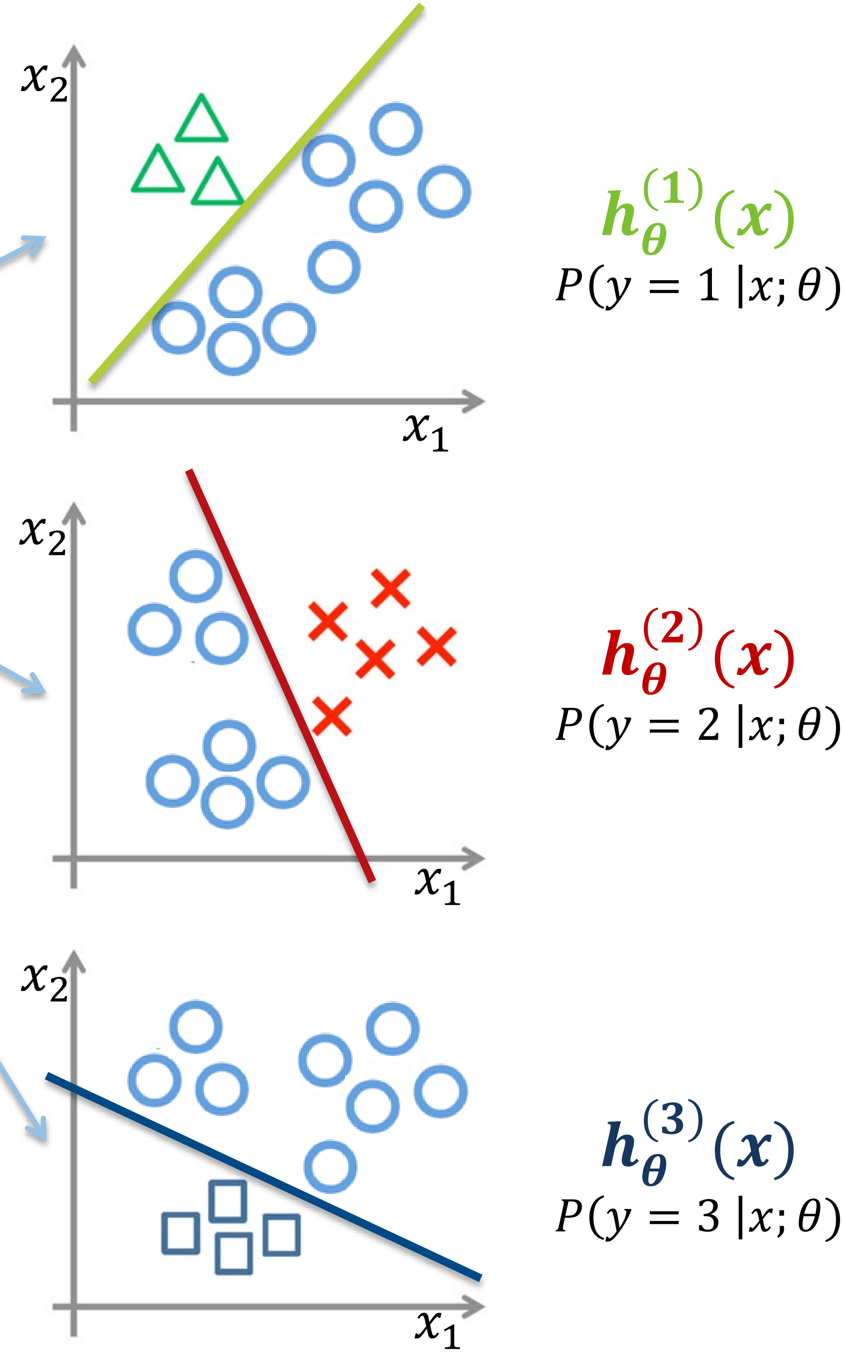
Multi-class classification
one-vs-all (one-vs-rest)



- Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$

- $h_\theta^{(i)}(x) = P(y = i \mid x; \theta),\ \ i = 1,2,3$

- To make a prediction on a new input x, pick the class that maximizes the probability: $\max_i h_\theta^{(i)}(x)$



Multi-class classification

- One-vs-all (one-vs-rest)
  - Train one binary classification model for each class (vs all the other classes).
    - Number of models is equal to the number of classes (c)

- One-vs-one
  - You can also train one binary classification model for each pair of classes.
    - Number of models is in the order of 2^c

Nonlinear Classification

Non-linear classification with Logistic Regression.

Logistic Regression

Non-linear decision boundary

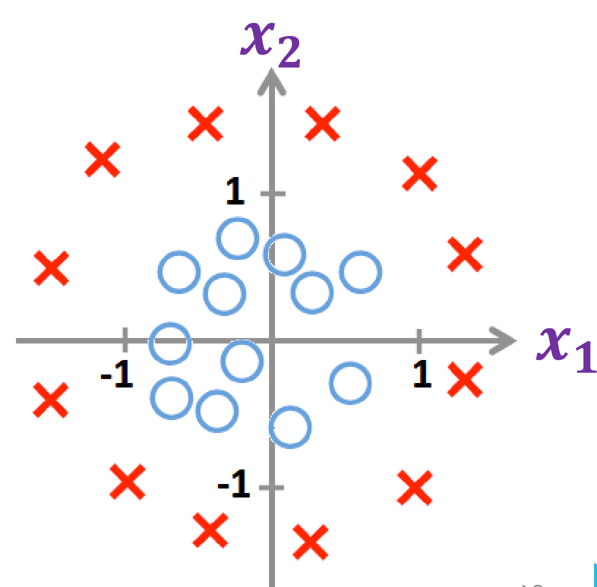
Example of a non-linear decision boundary

- Let's add extra higher order polynomial terms to the features:

$$\begin{aligned} h_\theta(x) &= g(\theta^T x) \\ &= g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2) \\ &= g(-1 + x_1^2 + x_2^2) \end{aligned}$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Predict $y = 1$ if $x_1^2 + x_2^2 \geq 1$



Logistic Regression

Non-linear decision boundary

Example of a non-linear decision boundary

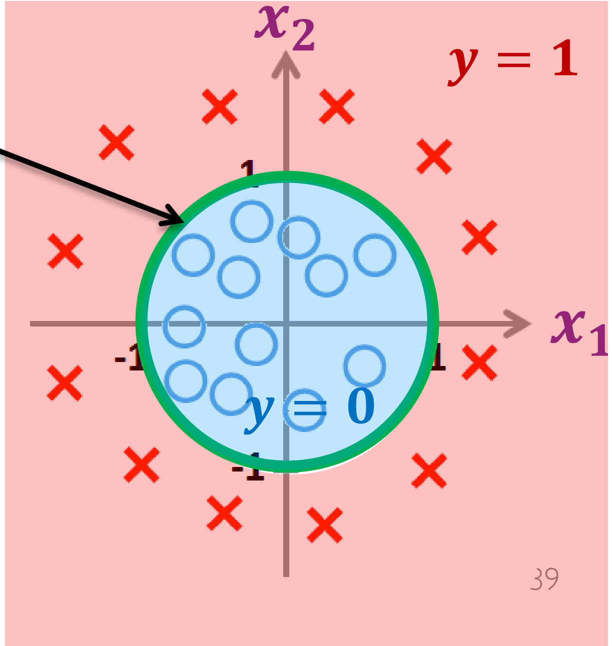
- Let's add extra higher order polynomial terms to the features:

$$h_\theta(x) = g(\theta^T x)$$
$$= g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$
$$= g(-1 + x_1^2 + x_2^2)$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Predict $y = 1$ if $x_1^2 + x_2^2 \geq 1$

decision boundary
$x_1^2 + x_2^2 = 1$



NOTE:

- The decision boundary is a property of the hypothesis and the parameters θ, not a property of the training dataset.. Choosing a different θ leads to a different decision boundary (regardless of the training dataset).
- The training dataset is used to fit the parameters θ (i.e. find optimal θ). We will see how later.

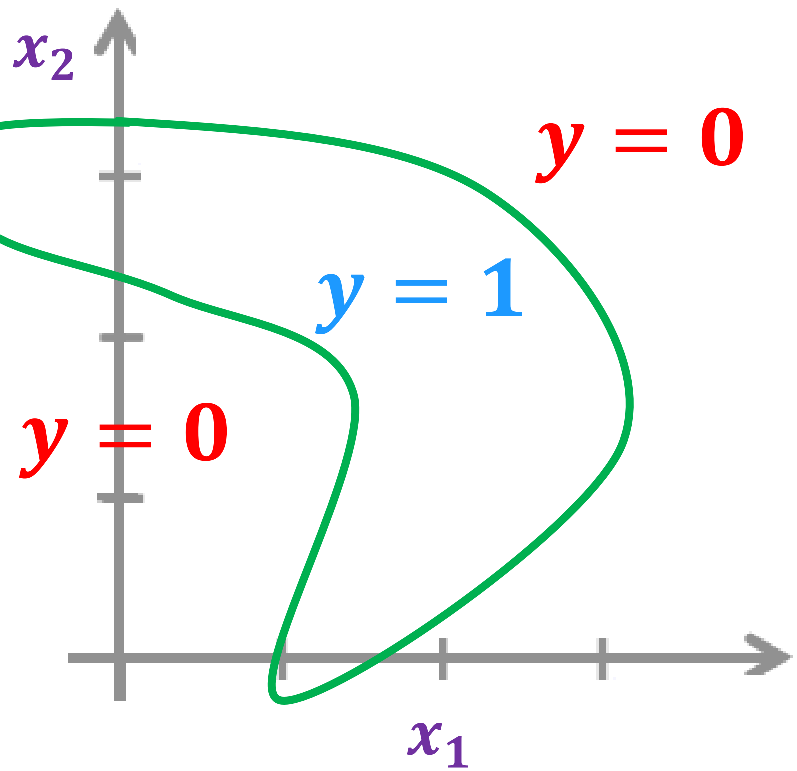
Logistic Regression

More complex non-linear decision boundary

Example of a **more complex** non-linear decision boundary

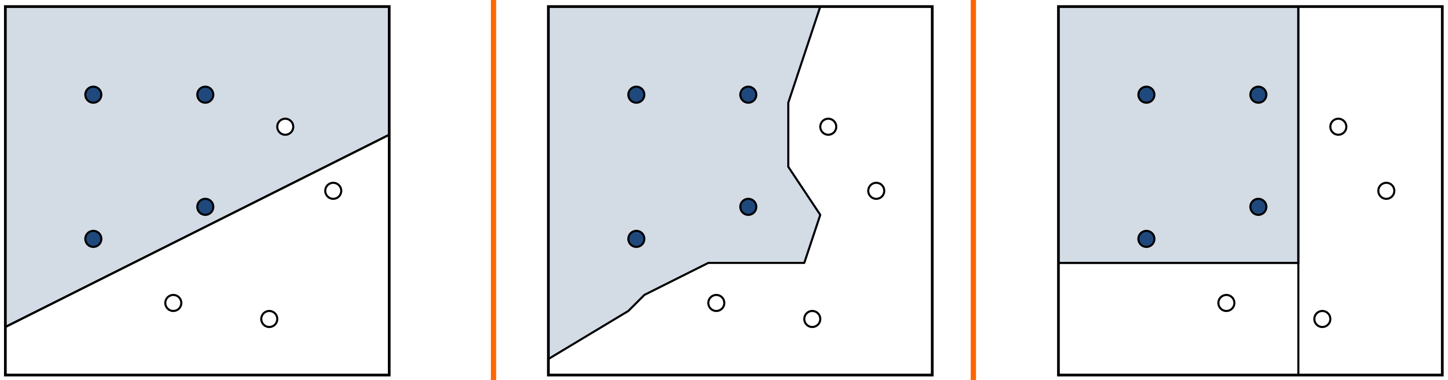
- Let's add even more extra higher order polynomial terms to the features:

$$h_\theta(x) = (\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \ \dots)$$



The K Nearest Neighbors Classifier
KNN

Nearest Neighbors (KNN)



Logistic Regression

K Nearest
Neighbors

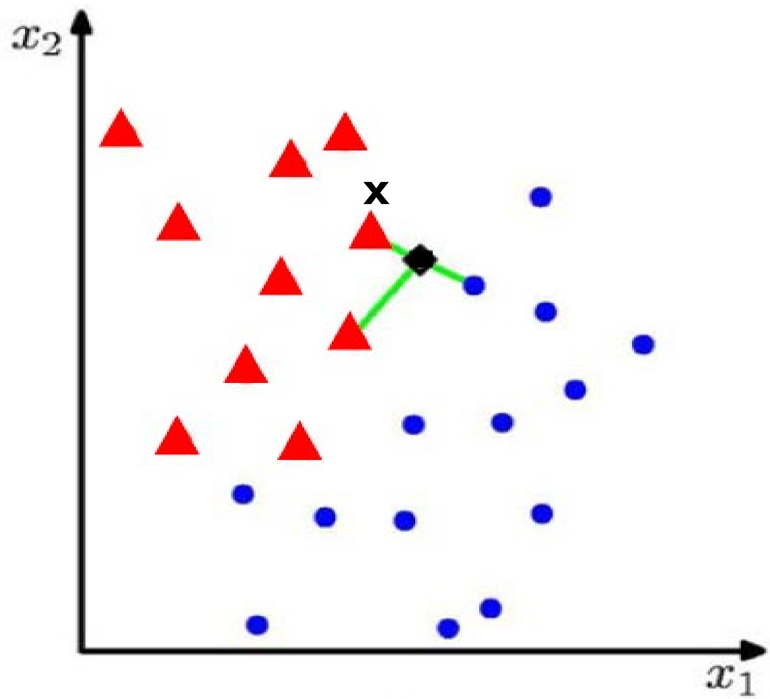
Decision
Tree

K Nearest Neighbors (KNN) - Classification

- Simple method that does not require learning (the model is just the labeled training dataset itself).
- For each test data-point **x**, to be classified, find the K nearest points in the training data.
- Classify the point **x**, according to the majority vote of their class labels

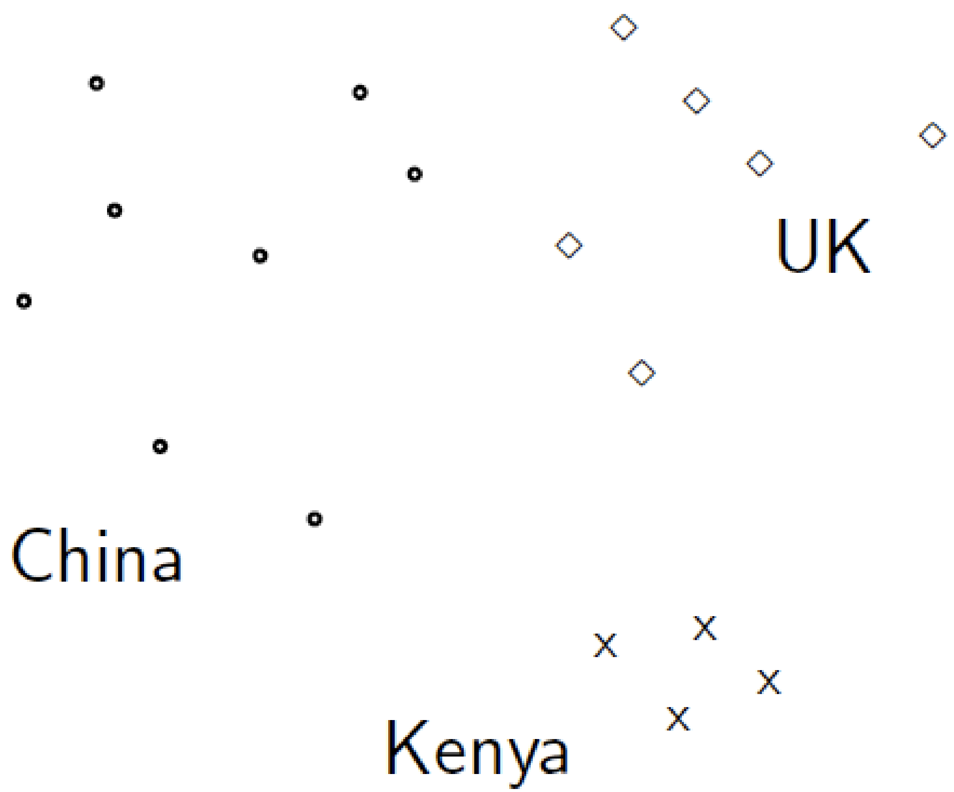
Example:

- K = 3
- 2 classes (red / blue)



Classification by Nearest Neighbor

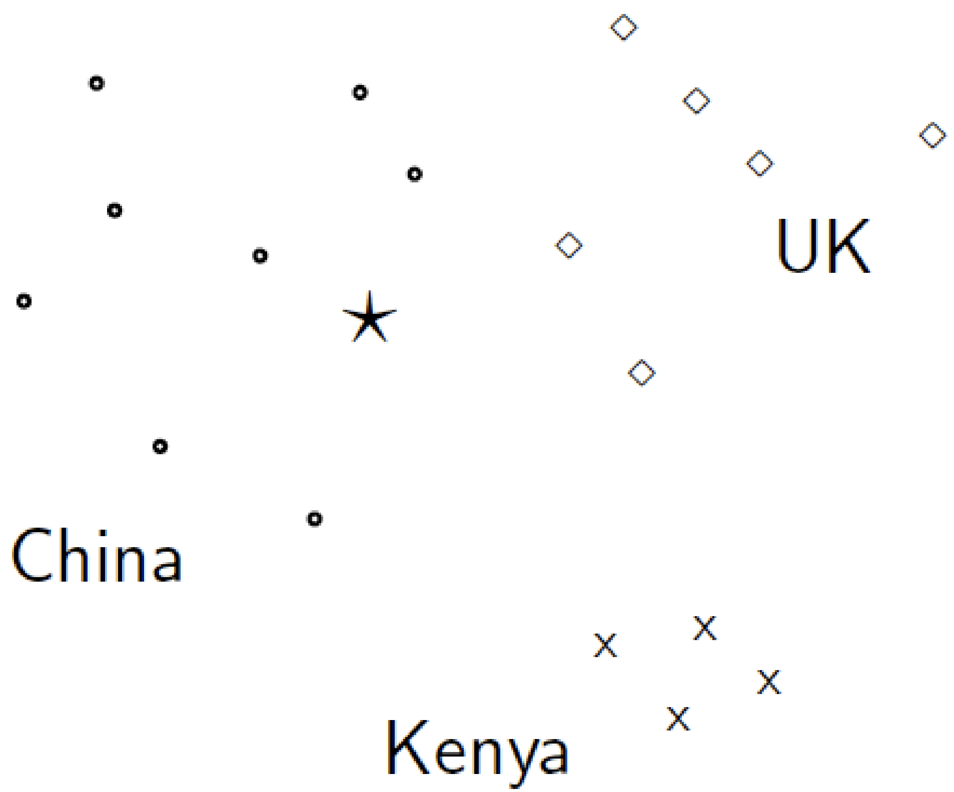
Classes in the vector space



Word vector document classification – here the vector space is illustrated as having 2 dimensions. But for real text document data, what would be our features? How many?

Classification by Nearest Neighbor

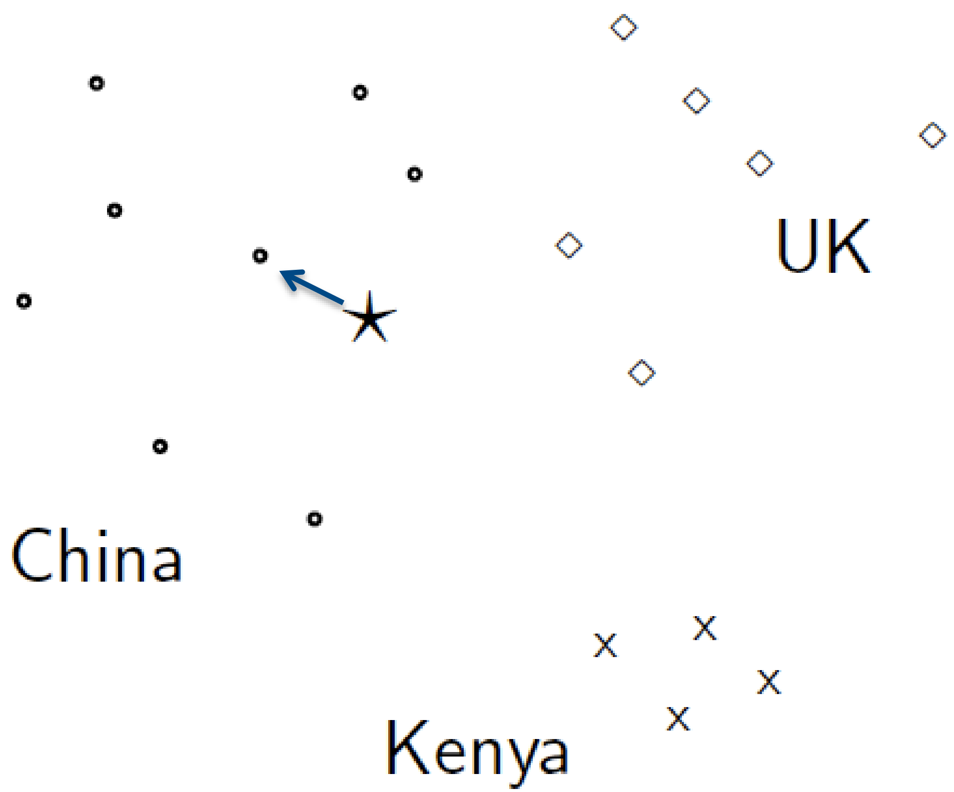
Classes in the vector space



Should the document $\star$ be assigned to *China*, *UK* or *Kenya*?

Classification by Nearest Neighbor

Classes in the vector space

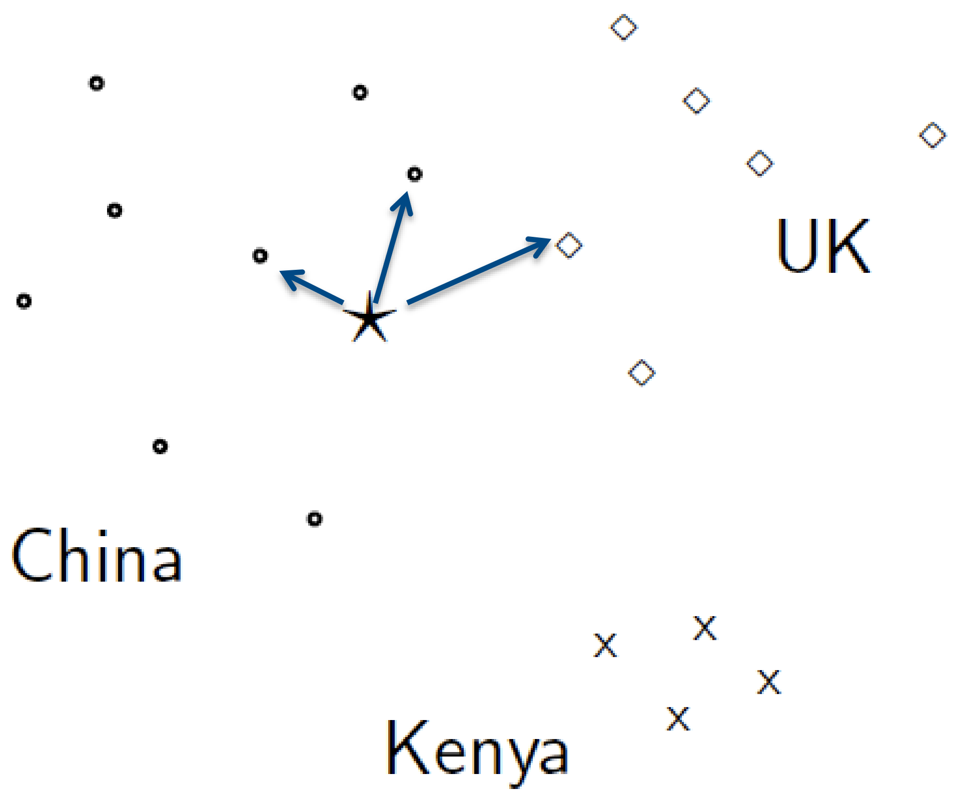


Should the document ⋆ be assigned to *China*, *UK* or *Kenya*?

Classify the test document as the class of the document "nearest" to the query document (use vector similarity to find most similar doc)

Classification by KNN

Classes in the vector space

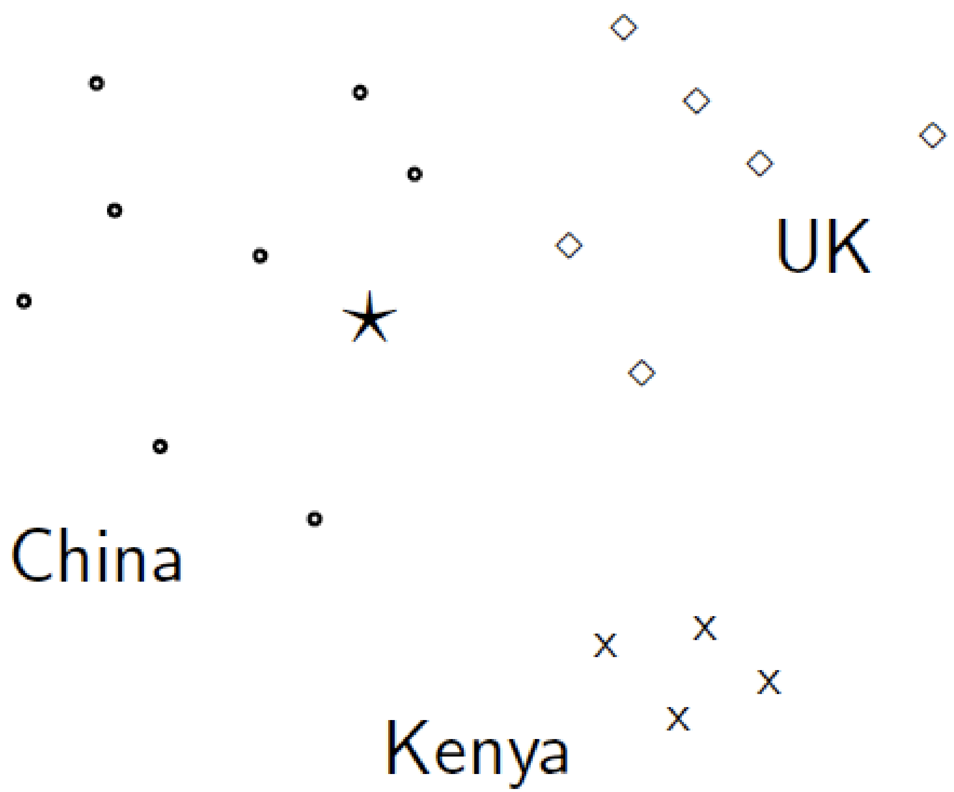


Should the document ⋆ be assigned to *China*, *UK* or *Kenya*?

Classify the test document as the majority class of the k documents “nearest” to the query document

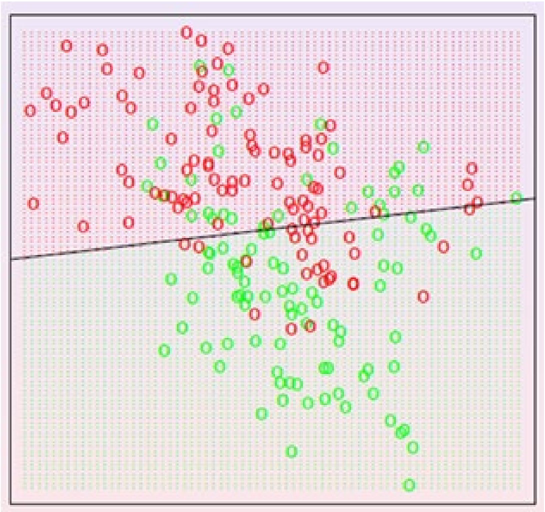
Classification by KNN

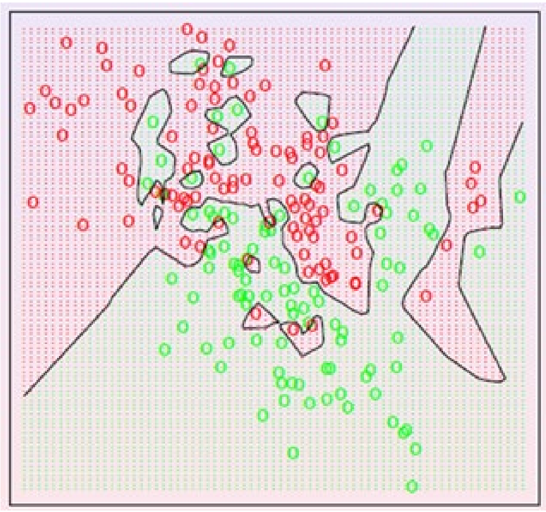
Classes in the vector space

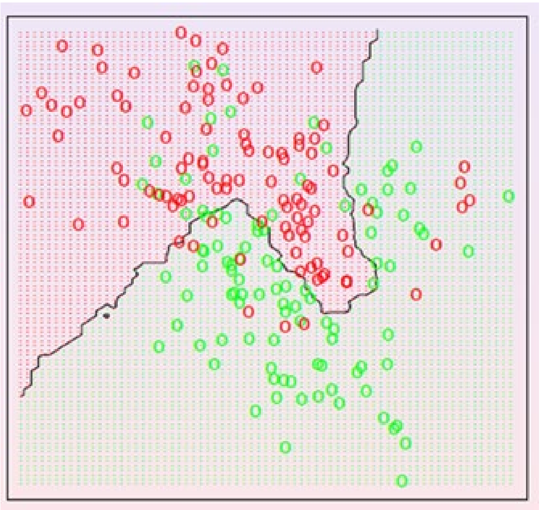


Should the document $\star$ be assigned to *China*, *UK* or *Kenya*?

KNN – Model Complexity







- Linear model (e.g. Logistic Regression)
- Very simple decision boundary.

- KNN with K=1
- It produces a complex decision boundary on this dataset.

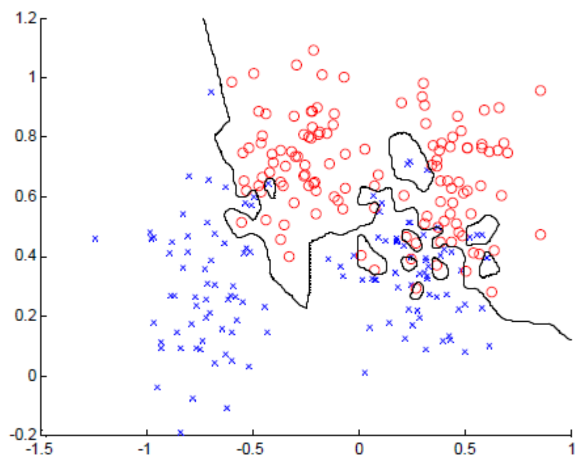
- KNN with K=15
- It produces a simpler decision boundary than K=1.

Smaller K produces a more complex decision boundary.

KNN – Model Complexity

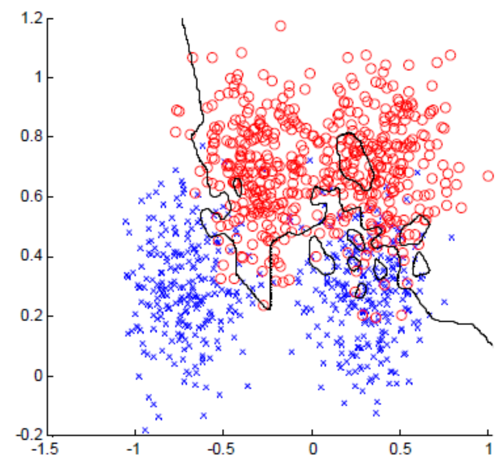
K = 1

Training data



error = 0.0

Testing data

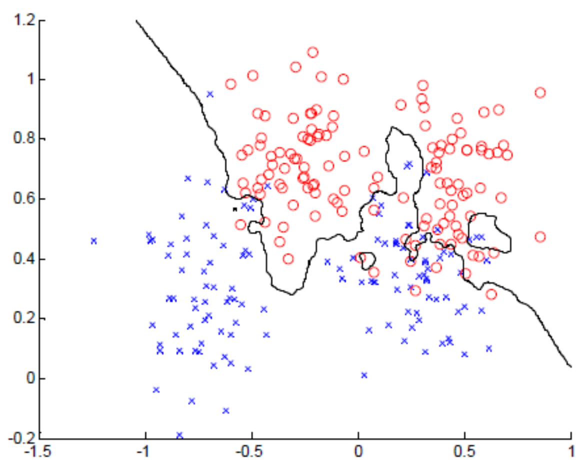


error = 0.15

KNN – Model Complexity

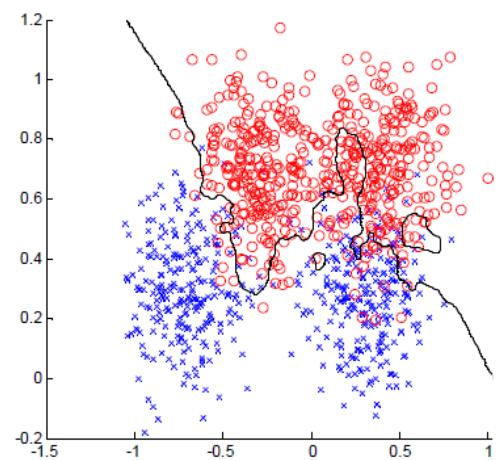
K = 3

Training data



error = 0.0760

Testing data

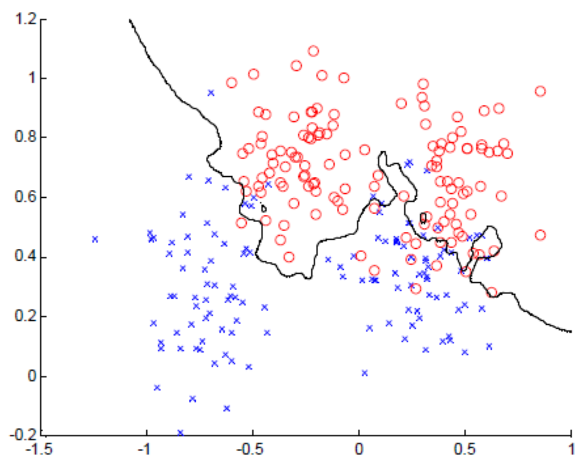


error = 0.1340

KNN – Model Complexity

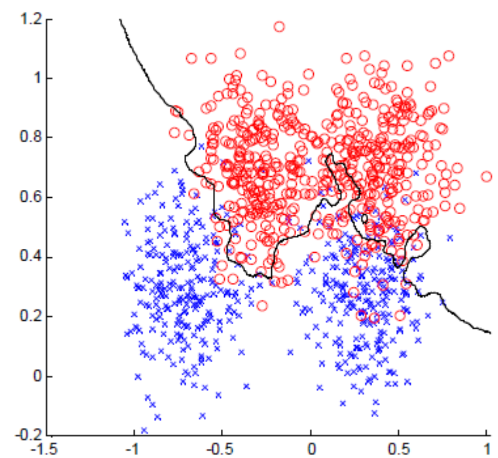
K = 7

Training data



error = 0.1320

Testing data

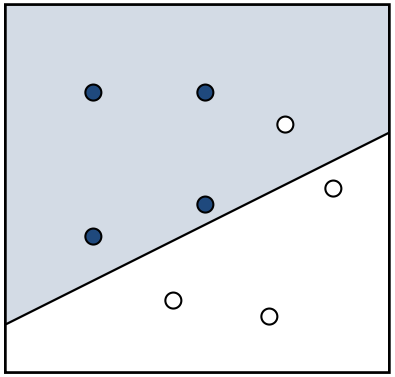


error = 0.1110

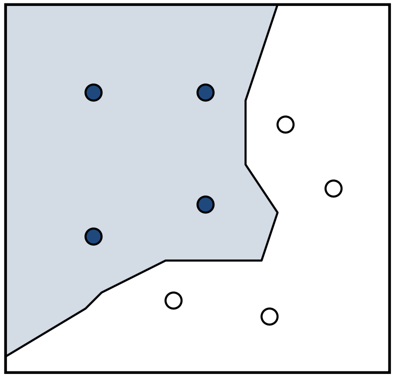
HALMSTAD UNIVERSITY

Decision Tree Classifier

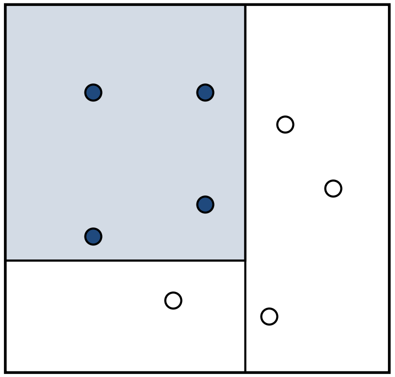
Classification with Decision Trees



Logistic Regression



K Nearest Neighbors

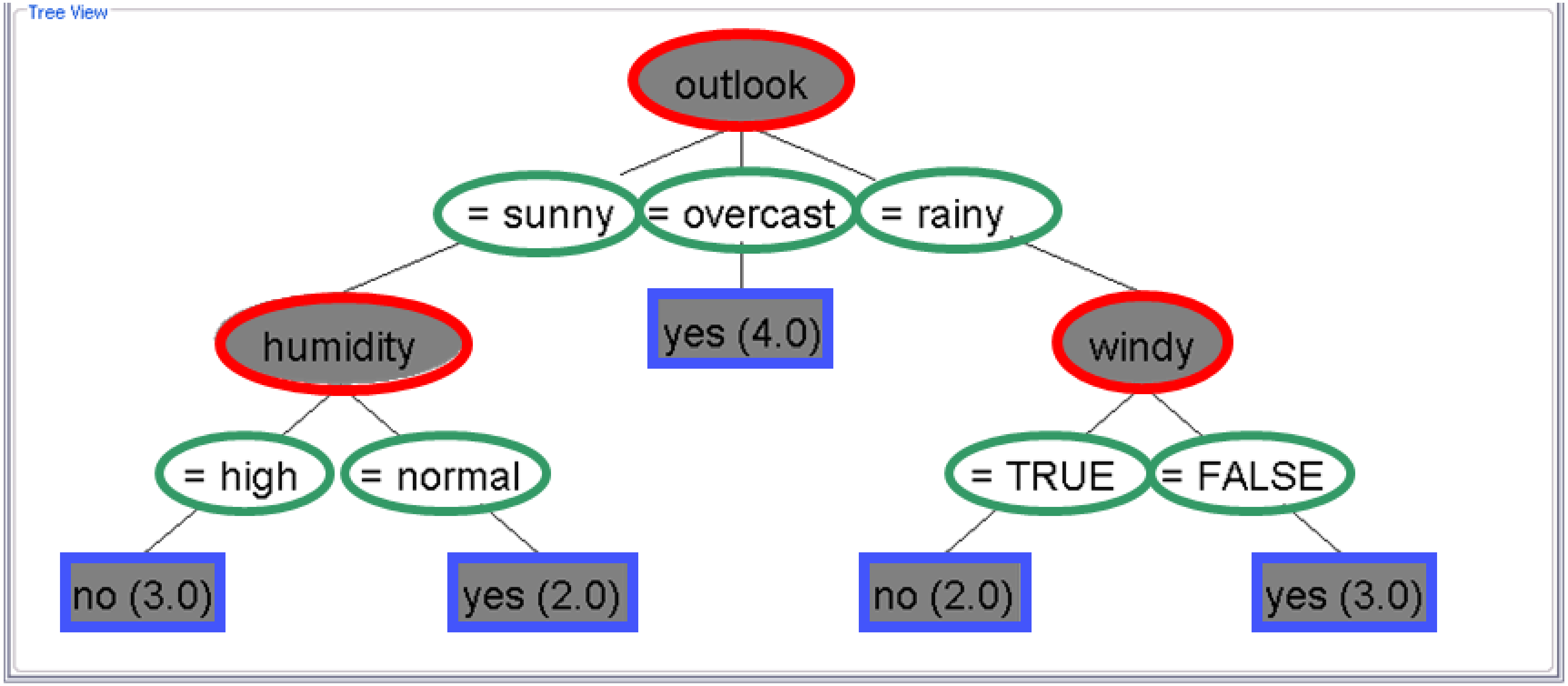


Decision Tree

Decision Tree

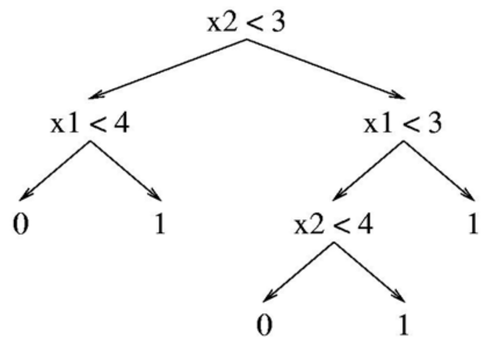
Attributes / Features
Attribute Values
Classes

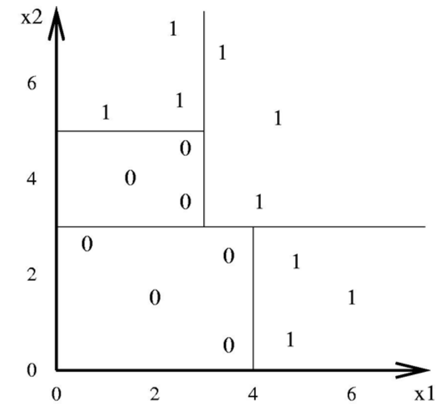
- Example: Shall we play golf today ?



Decision Tree - Classification

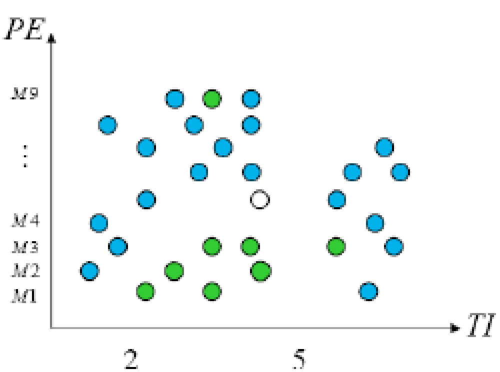
- At each nodes
  - A question is asked about data
  - One child node per possible answer
- Leaf nodes
  - Class label (i.e. decision to take)

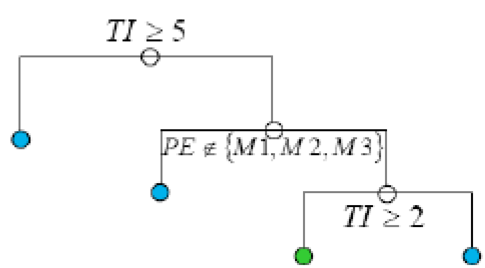


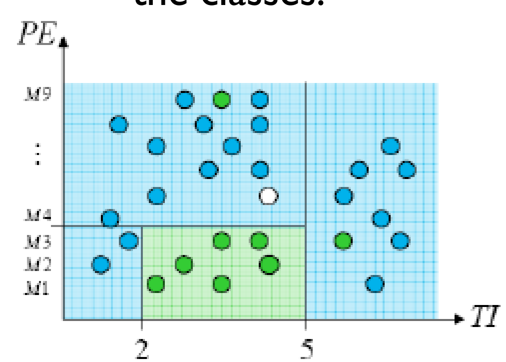


- Building the Tree:
  - For each node, find the feature F + threshold value T
  - ... that split the samples assigned to the node into 2 subsets
  - ... so as to maximize the label purity within these subsets.

Simple, practical and easy to interpret.
Given a set of instances (with a set of features), a tree is constructed with internal nodes as the features and the leaves as the classes.







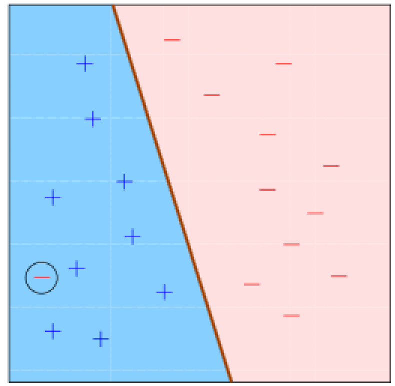
HALMSTAD UNIVERSITY

In the next lecture:

Overfitting, Generalization, Regularization

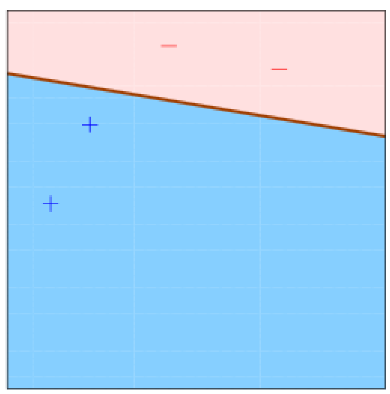
Good and Bad Classifiers

Good:

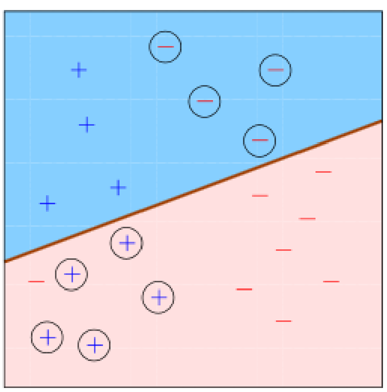


sufficient data
low training error
simple classifier

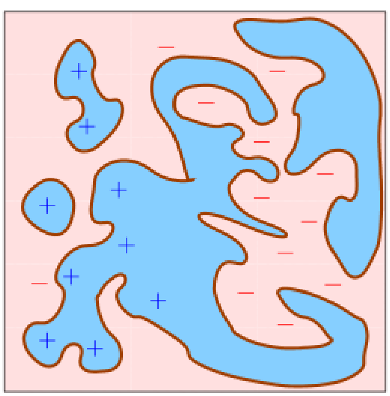
Bad:



insufficient data



training error
too high



classifier
too complex

Overfitting

- Overfitting:
  - A model that performs well on the training examples, but poorly on new examples.
  - Training and testing on the same data will generally lead to overfitting and produce a model which looks good only for this particular training dataset.

- To avoid overfitting:
  - Use separate training and testing data
  - Use cross-validation
  - Try using simple models first
  - Use regularization or ensemble models …

  - We will talk more about this next week.

Performance evaluation

Cross-Validation
(10 fold)