Visuelle Perzeption für Mensch-Maschine Schnittstellen

Vorlesung, WS 2013 / 2014

Prof. Dr. Rainer Stiefelhagen
Dr. Saquib Sarfraz
Institut für Anthropomatik
Karlsruher Institut für Technologie - KIT

http://cvhci.anthropomatik.kit.edu/
rainer.stiefelhagen@kit.edu

Some Machine Learning (fast track)

WS 2013/14

saquib.sarfraz@kit.edu

Based on Andrew Ng lectures (Stanford Uni.)

Institute for Anthropomatics
Computer Vision for Human-Computer Interaction

Quick Machine Learning

- Supervised- (labeled data)
- Unsupervised (no labels, not seen)- Clustering, e.g. by GMM (covered before)

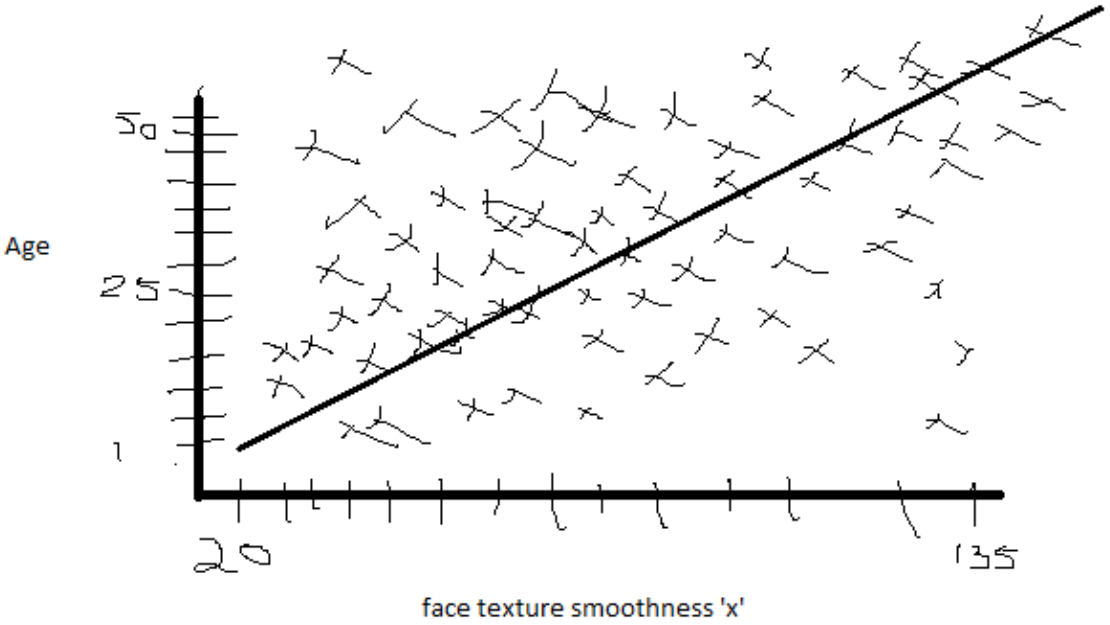
In this lecture

- Supervised learning algorithms
  (Regression, Logistic regression, SVM‘s)

- Try to build the intuitive understanding on how these are developed.

Institute for Anthropomatics
Computer Vision for Human-Computer Interaction

Example: Age estimation from facial images



Regression Problem
Predict real-valued output

Supervised Learning

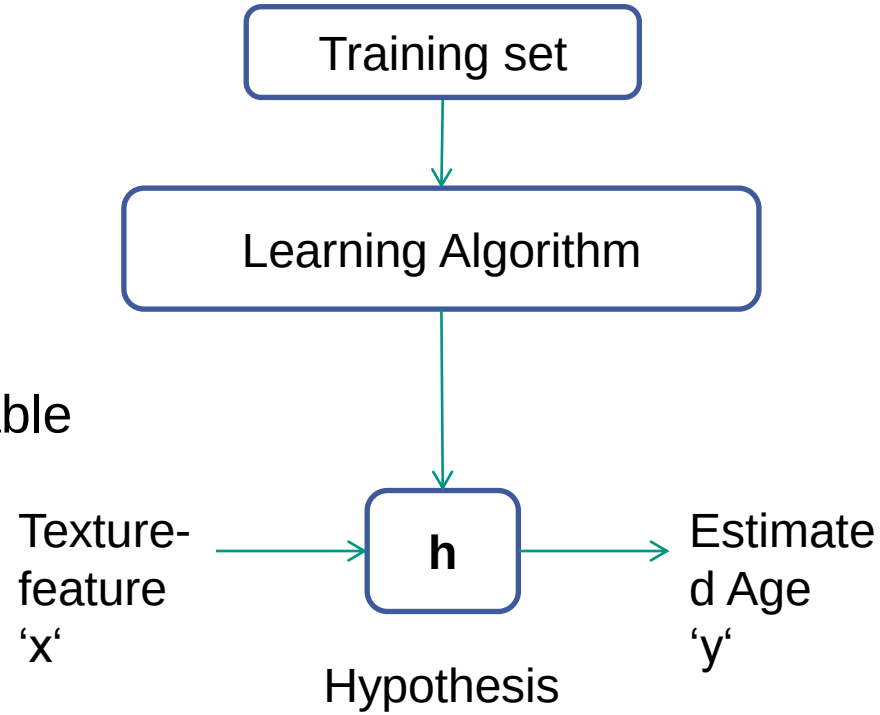
Given the label for each example in the training set

Classification Problem
Perdict the discrete-valued output e.g. 0 -1

Given Training set

Notation:
m – No. Of traing examples
x‘s - input variables or features
y‘s´- output variable / target variable

(x^i, y^i) – ith training example



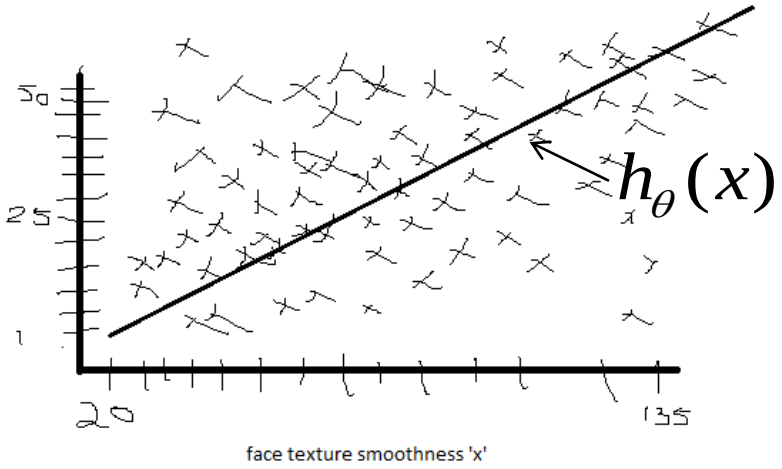
How do we represent h

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Linear Regression- Cost function

Hypothesis $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters θ_i - how to choose θ_i



One way:- choose θ_0, θ_1 so $h_\theta(x)$

is close to y for our training examples (x,y)

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

Squared error function

→ Call it $J(\theta_0, \theta_1)$ - Cost Function

Minimize by Gradient descent

- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$
- Untill we reach minimum

$$repeat\ untill\ convergence \quad \{$$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

$$\}$$

Simultaneously update θ_j for **j = 0 and j = 1**

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

$$= \frac{\partial}{\partial\theta_j}\left(\frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0+\theta_1 x^i-y^i\right)^2\right)$$

$$J=0: \frac{\partial}{\partial\theta_0}\left(\frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0+\theta_1 x^i-y^i\right)^2\right) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^i)-y^i\right)$$

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

Gradient Descent Algorithm

repeat untill convergence {

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^i)-y^i\right)$$

$$\theta_1 := \theta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^i)-y^i\right)\cdot x^i$$

}

Multivariate Linear Regression

Let the input x contain multiple features such that $x^i = [x_1, x_2, x_3, ..., x_n]$

Where x_j^i is the jth variable of the ith training example.

Hypothesis $h_\theta(x)$ can now be written as: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ...\theta_n x_n$

For convenience of notation, lets define $x_0 = 1$, so we can write

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + ...\theta_n x_n$$

$$= \theta^T x$$

$$\theta^T x \in \square^{n+1}$$

Multivariate Linear Regression …

Cost fucntion

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

Gradient Descent Algorithm

$$repeat\{$$

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta_0,\theta_1,...,\theta_n)$$

$$\}$$

Simultaneously update for j=0,1,2..,n

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^i)-y^i\right)$$

$$\theta_1 := \theta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^i)-y^i\right)\cdot x_1^i$$

$$\theta_2 := \theta_2 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^i)-y^i\right)\cdot x_2^i$$

Applying Gradient descent—in practice

- Feature scaling: all the features should be in approximately the same range

  Better if every feature is in e.g. $-1 \leq x_i \leq 1$

- We can do that by Zero Mean Normalization

Regression Vs Classification

e.g instead of age lets determine gender from facial images.

Here output y=0 (female) OR y=1 (Male)

- We can use linear regression hypothesis for this purpose.

e.g. by thresholding $h_\theta(x)$ such that $y = 1$ *when* $h_\theta(x) \geq 0.5$

- Classification using linear regression isn‘t a good idea !

Logistic Regression
a useful classifier

Classification output : y = 0 OR 1

$h_\theta(x)$ can be > 1 or < 0

Logistic regression Model : we want $0 \le h_\theta(x) \le 1$

$$h_\theta(x) = g(\theta^T x)$$

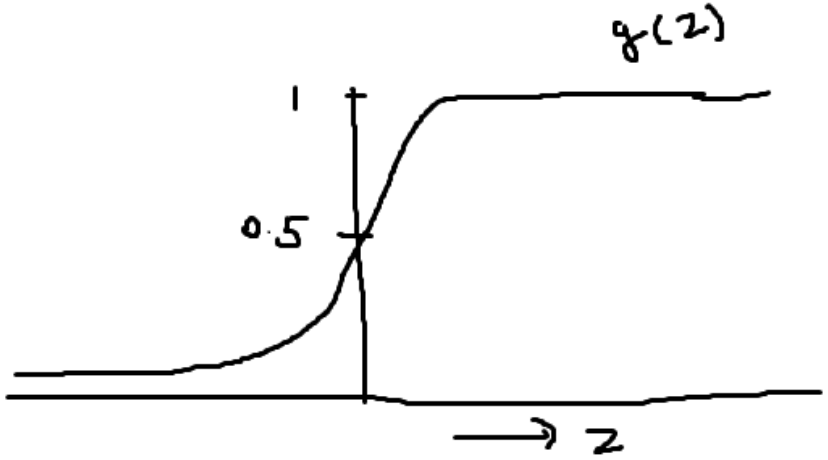
Where is g(z) is a sigmoid function or Logistic function. $g(z) = \frac{1}{1+e^{-z}}$

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

Interpretation of Hypothesis output

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

$$h_\theta(x) = P(y=1 \mid x;\theta)$$

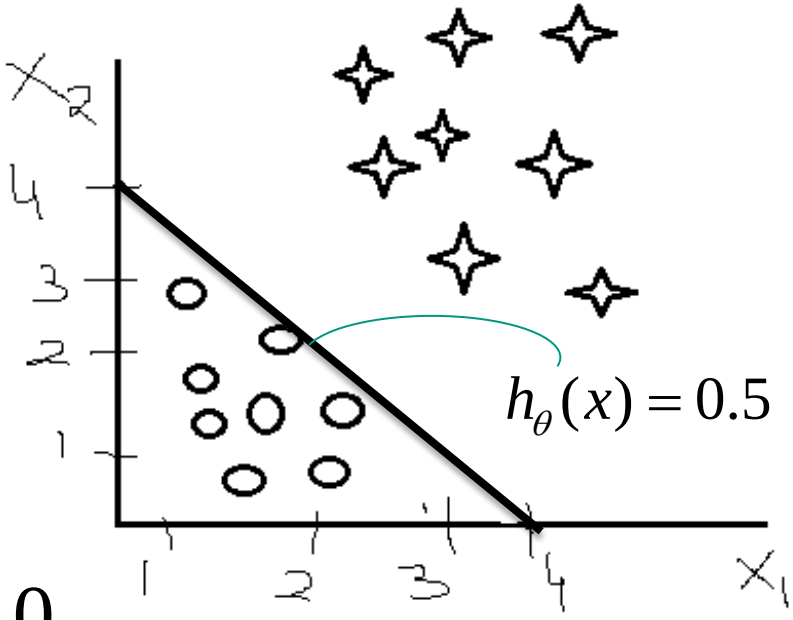


Predict y = 1 when $h_\theta(x) > 0.5$ or vice versa

Which means $\theta^T x \geq 0$ *for* $y = 1$

Descision Boundary

$$h_\theta(x) = g(\theta_0 + \theta_1 x + \theta_2 x)$$



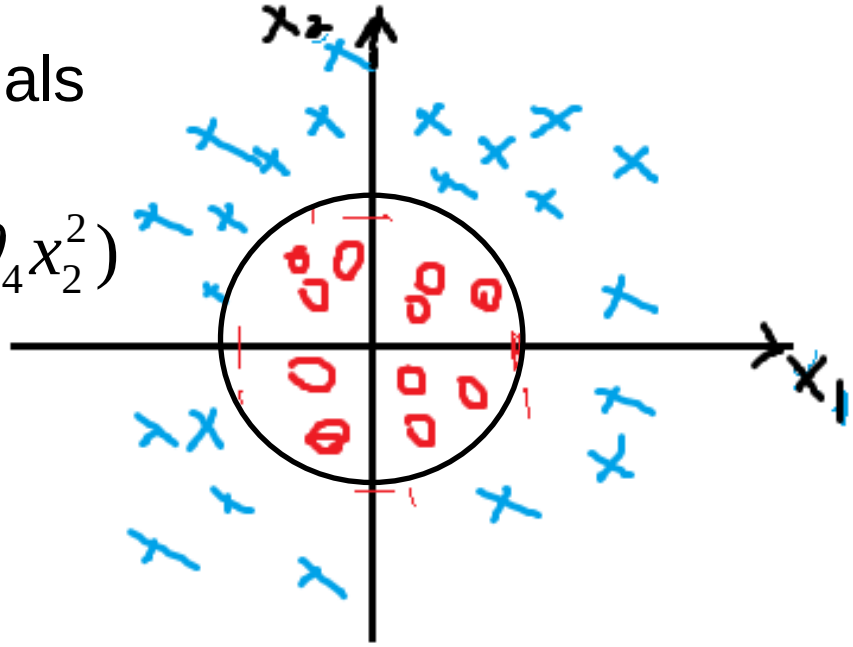
Predict y=1 , if $-4 + x_1 + x_2 \geq 0$

$x_1 + x_1 = 4$ Defines the boundary

Nonlinear descision boundaries

We can use higher order polynomials

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



Suppose $\theta = [-1, 0, 0, 1, 1]$

Predict "y=1", if $-1 + x_1^2 + x_2^2 \geq 0$

We can define even complex boundaries by using even higher order polynials

How to fit Parameters for Logistic regression

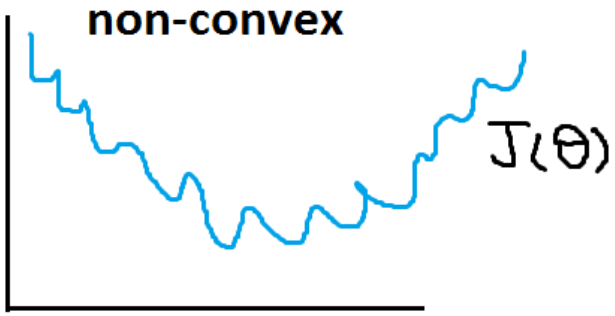
Linear Regression:

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

For Logistic regression

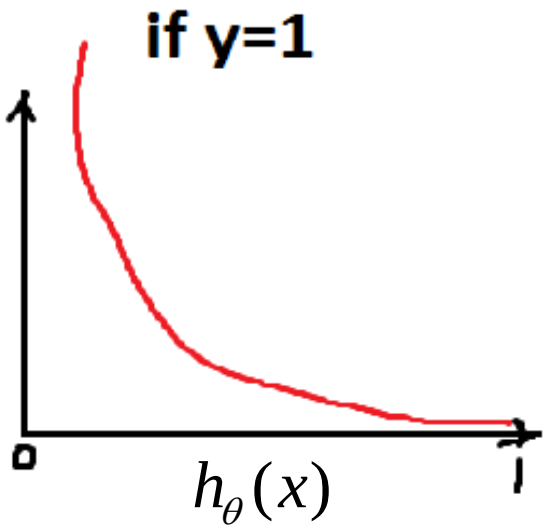
Cost Function

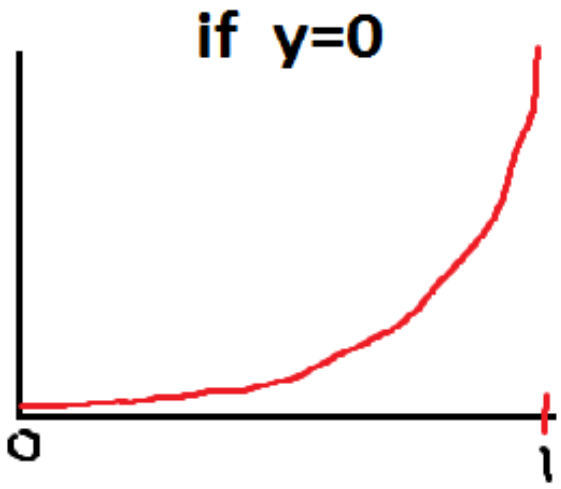
$$\cos t(h_\theta(x)^i, y^i) = \frac{1}{2}\left(h_\theta(x^i)-y^i\right)^2$$



Logistic Regression Cost Function

$$\cos t(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & if \quad y = 1 \\ -\log(1 - h_\theta(x)) & if \quad y = 0 \end{cases}$$





Simple way to write it
:

$$\cos t(h_\theta(x^i), y^i) = -y^i \log(h_\theta(x^i)) - (1 - y^i) \log(1 - h_\theta(x^i))$$

Logistic Regression Cost Function

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \cos t(h_\theta(x^i), y^i)$$

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

To fit parameters θ **:**

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

To classify a new x, use

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

Gradient Descent for logistic regression

$$repeat\ untill\ convergence \quad \{$$

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

$$\}$$

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

$$repeat \quad \{$$

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

$$\}$$

This update rule looks exactly the same as Linear regression. But this does not mean that the linear regresison and Logistic regresison are same

Multiclass classification

e.g face recogniton as opposed to gender classification

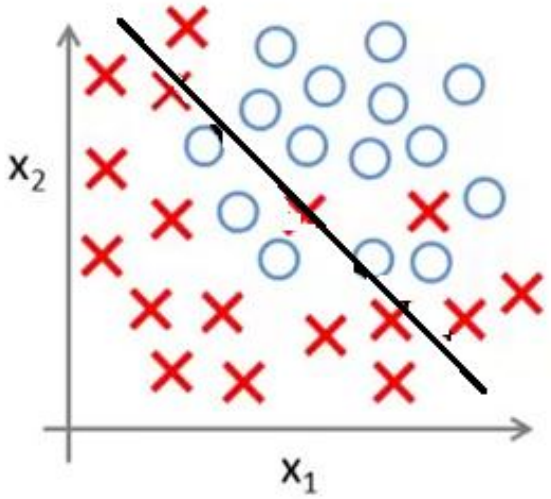
We can use logistic regression for multiclass probelms. Using a **One-vs-all** strategy

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

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

The Problem of Overfitting

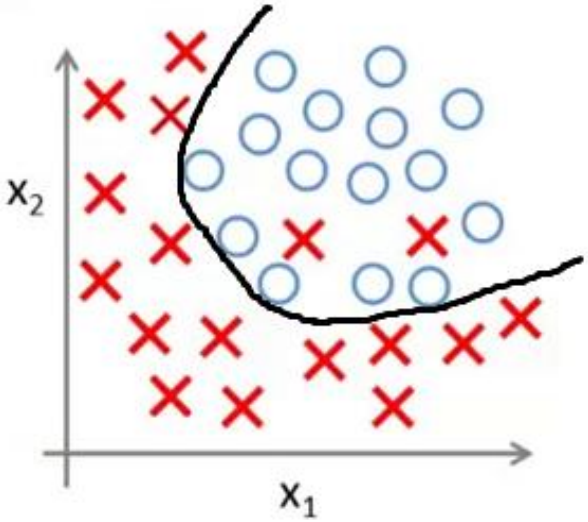
Example: Logistic regression



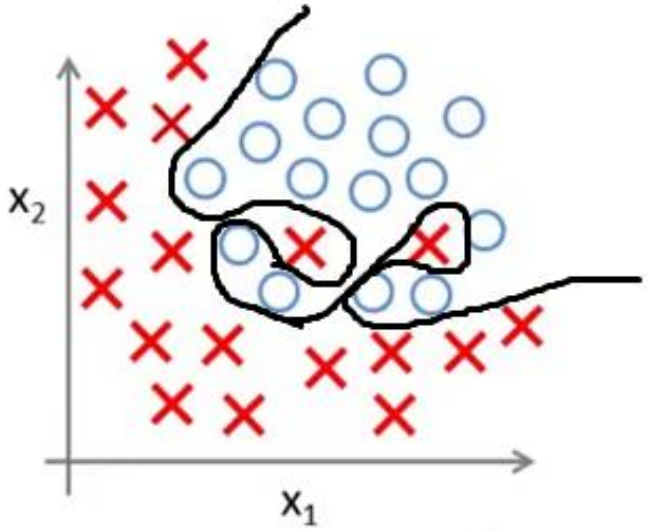
$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

Underfitting



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \ldots)$

Overfitting

Solution to avoid overfitting

1. Reduce no. Of features

2. Regularization

- keep all the features but reduce magnitude/values of parameters θ_j

Small values of parameters $\theta_0, \theta_1, \theta_2, \theta_3, \ldots\ldots, \theta_n,$

- simpler hypothesis (less prone to overfitting)

We can do that by penalizing the parameters in the cost function

$$\min_{\theta_0, \theta_1, .., \theta_n} imize, J(\theta) \quad = \quad \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^i) - y^i \right)^2 + \lambda \sum_{i=1}^{n} \theta_i^2$$

Regularized Logistic regression

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

Minimizing the cost function with regularization term

$$repeat \quad \{$$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^i) - y^i \right) \cdot x_0^i$$

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

$$for \qquad j = 1, 2, 3...., n$$

$$\}$$

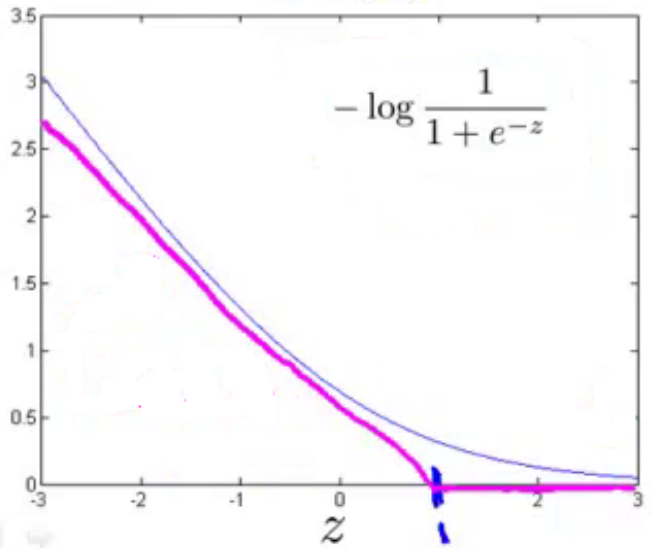
Support Vector Machines (SVM)

Alternative view of logistic regression

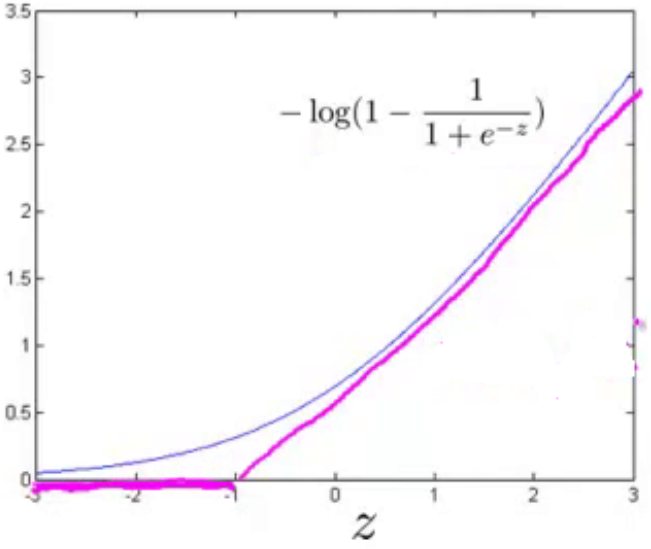
Cost of example: $-(y \log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$

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

If $y = 1$ (want $\theta^T x \gg 0$):



If $y = 0$ (want $\theta^T x \ll 0$):



SVM Cost function/ Objective function

Logistic regression

$$\min_{\theta} \frac{1}{m}\left[\sum_{i=1}^{m} y^i \underbrace{(-\log(h_\theta(x^i))}_{\cos t_1 \theta^T x^{(i)}} + (1-y^i)\underbrace{(-\log(1-h_\theta(x^i))}_{\cos t_0 \theta^T x^{(i)}}\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Support Vector Machine

$$\min_{\theta} \frac{1}{\cancel{m}}\left[\sum_{i=1}^{m} y^i \cos t_1 \theta^T x^{(i)} + (1-y^i)\cos t_0 \theta^T x^{(i)}\right] + \frac{\lambda}{2\cancel{m}}\sum_{j=1}^{n}\theta_j^2$$

$$\min_{\theta} C\left[\sum_{i=1}^{m} y^i \cos t_1 \theta^T x^{(i)} + (1-y^i)\cos t_0 \theta^T x^{(i)}\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

SVM Hypothesis

$$\min_{\theta} C\left[\sum_{i=1}^{m} y^i \cos t_1 \theta^T x^{(i)} + (1-y^i)\cos t_0 \theta^T x^{(i)}\right] + \frac{1}{2}\sum_{j=1}^{n} \theta_j^2$$

Hypothesis:

$$h_\theta(x) = \begin{cases} 1 & if \quad \theta^T x \geq 0 \\ 0 & Otherwise \end{cases}$$

SVM Decision Boundary

Some intuition behind optimization of the SVM objective fucntion leading to the large margin decision boundaries.

SVM Objective function

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2 = \frac{1}{2} \|\theta\|^2$$

$$s.t \quad \theta^T x^{(i)} \geq 1 \quad if \quad y^{(i)} = 1$$

$$\theta^T x^{(i)} \leq -1 \quad if \quad y^{(i)} = 0$$

We can write

$$\theta^T x^{(i)} = P^{(i)} \|\theta\|$$

Where $P^{(i)}$ is the projection of vector x onto the vector Θ

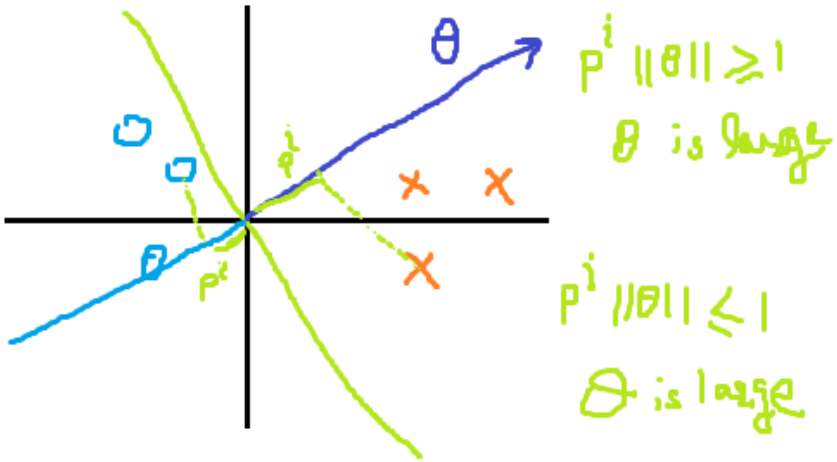
SVM Decision boundary- Large Margin

$$\min_{\theta} \frac{1}{2}\|\theta\|^2$$

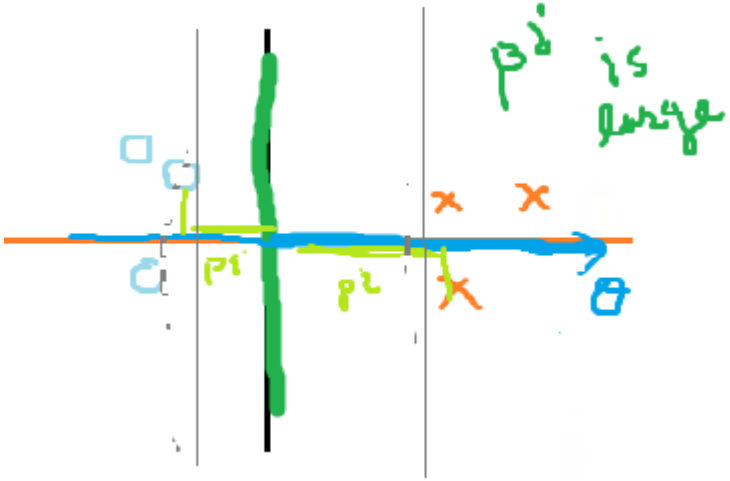
$$s.t \quad P^{(i)}\|\theta\| \geq 1 \quad if \quad y^{(i)} = 1$$

$$P^{(i)}\|\theta\| \leq -1 \quad if \quad y^{(i)} = 0$$

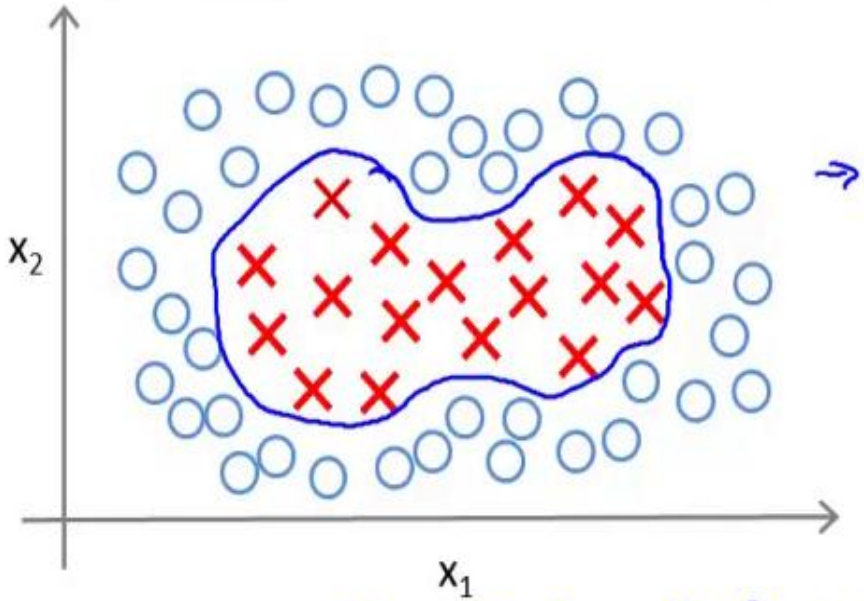
$$simplification: \quad \theta_0 = 0$$



SVM will choose this



Non-Linear decision boundaries



Predict $y = 1$ if

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$$

$h_\theta(x) = \begin{cases} 1 & \text{if } \theta_0 + \theta_1 x_1 + \cdots \geq 0 \\ 0 & \text{otherwise.} \end{cases}$

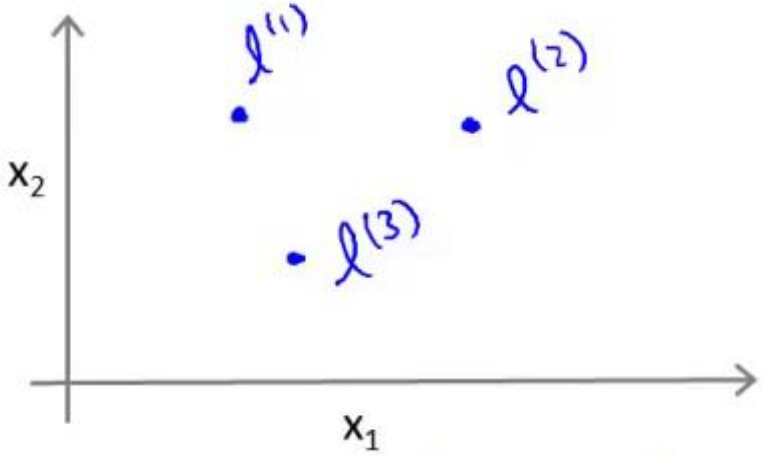
$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \cdots$

$f_1 = x_1, \quad f_2 = x_2, \quad f_3 = x_1 x_2, \quad f_4 = x_1^2, \quad f_5 = x_2^2, \ldots$

Is there a different / better choice of the features $f_1, f_2, f_3, \ldots$?

Idea of Kernels

Kernel



Given x, compute new feature depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$

Given x:

$$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$$f_2 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(2)}\|^2}{2\sigma^2}\right)$$

$$f_3 = \text{similarity}(x, l^{(3)}) = \exp(\dots)$$

Kernel (Gaussian kernels)

Kernels and Similarity

$$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{j=1}^{n}(x_j - l_j^{(1)})^2}{2\sigma^2}\right)$$

If $x \approx l^{(1)}$:

$f_1 \approx 1$

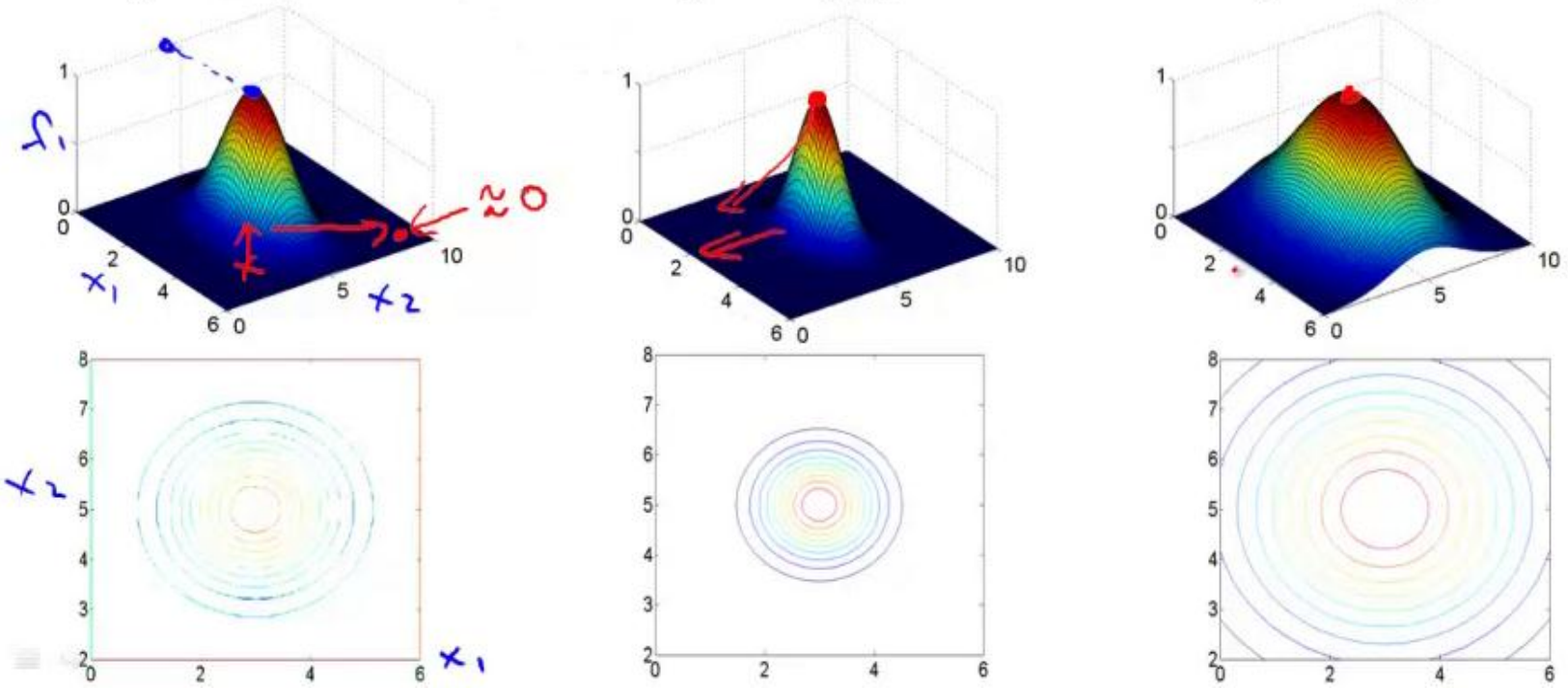
If x if far from $l^{(1)}$:

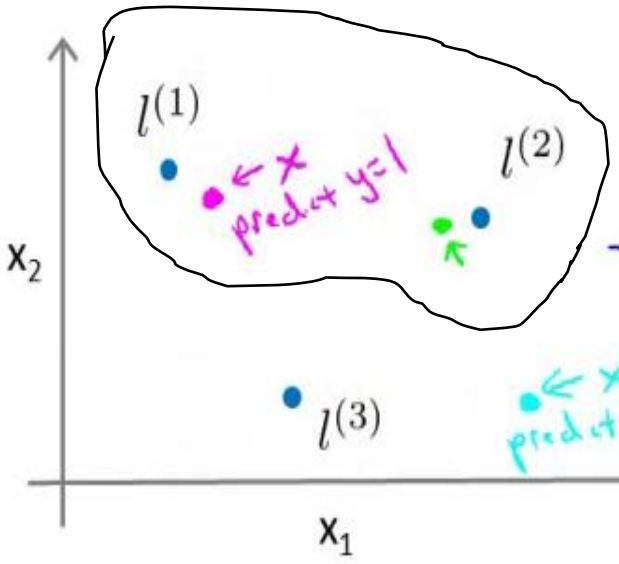
$f_1 \approx 0$

Example:

$$l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$\sigma^2 = 1$ $\sigma^2 = 0.5$ $\sigma^2 = 3$





Predict "1" when

$$\Rightarrow \theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$$

$\theta_0 = -0.5$, $\theta_1 = 1$, $\theta_2 = 1$, $\theta_3 = 0$

- How to choose landmarks?
- Can we use other similarity(kernels) functions?

SVM with Kernels

Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$,
choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \ldots, l^{(m)} = x^{(m)}$.

Given example x:

$$f_1 = \text{similarity}(x, l^{(1)})$$
$$f_2 = \text{similarity}(x, l^{(2)})$$
$$..$$

For training example $(x^{(i)}, y^{(i)})$:

$$f^{i} = \begin{bmatrix} f_0^{i} \\ f_1^{i} \\ f_2^{i} \\ \vdots \\ f_m^{i} \end{bmatrix} \qquad f_0^{i} = 1$$

SVM with Kernels

Hypothesis: Given x, compute features $f \in \mathbb{R}^{m+1}$

Predict "y=1" if $\theta^T f \geq 0$

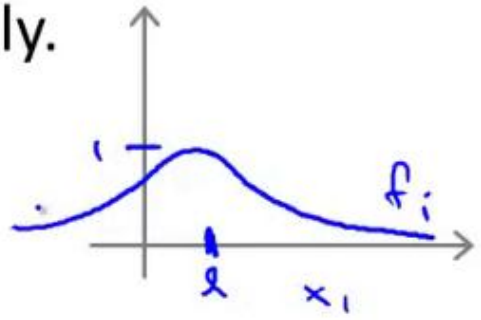
Training

$$\min_{\theta} C\left[\sum_{i=1}^{m} y^i \cos t_1 \theta^T f^{(i)} + (1-y^i)\cos t_0 \theta^T f^{(i)}\right] + \frac{1}{2}\sum_{j=1}^{m} \theta_j^2$$

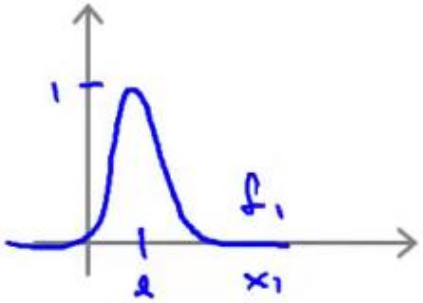
SVM parameters:

C ($= \frac{1}{\lambda}$). Large C: Lower bias, high variance.
Small C: Higher bias, low variance.

σ^2 Large σ^2: Features f_i vary more smoothly.
Higher bias, lower variance.



Small σ^2: Features f_i vary less smoothly.
Lower bias, higher variance.



Final words on SVM

Use SVM package (e.g. libsvm) to solve for parameters

Specify parameter C
Specify kernel

Tip:

Use linear SVM i.e. “No kernel“ Linear Kernel…when n is very large than the no. of training examples m.

Use Kernel (e.g Gausian) when n is small compared to m.

Thanks