UCK358E – INTR. TO ARTIFICIAL INTELLIGENCE

SPRING ‘23

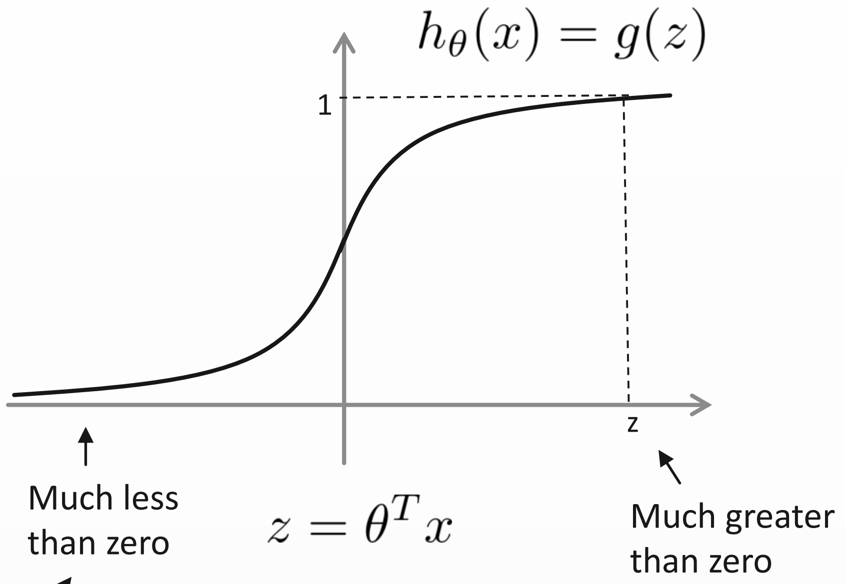
LECTURE 7

SUPPORT VECTOR MACHINES

Instructor: Asst. Prof. Barış Başpınar

Alternative View of Logistic Regression

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



If $y = 0$, we want $h_\theta(x) \approx 0$, $\theta^T x \ll 0$

If $y = 1$, we want $h_\theta(x) \approx 1$, $\theta^T x \gg 0$

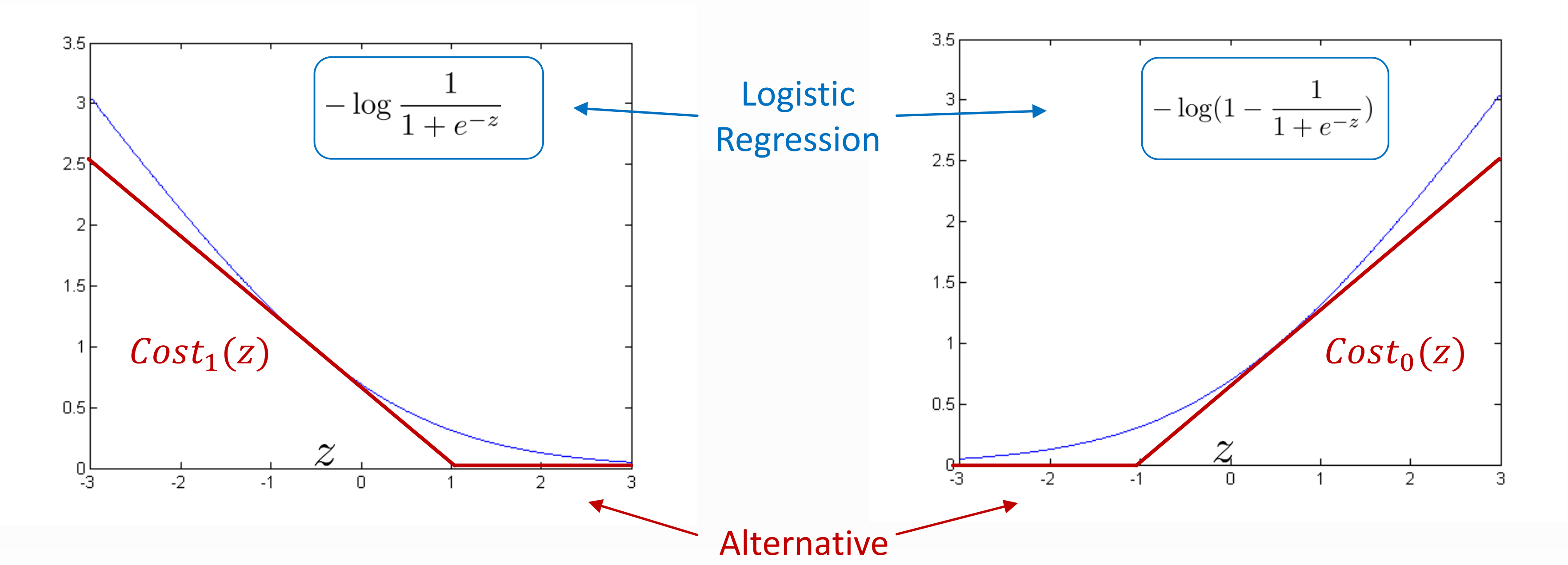
Alternative View of Logistic Regression

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 $\theta^T x \ll 0$):



Support Vector Machine

Logistic regression:

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

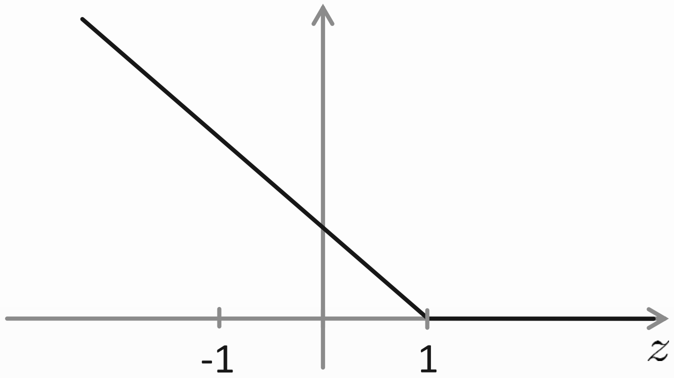
Support vector machine:

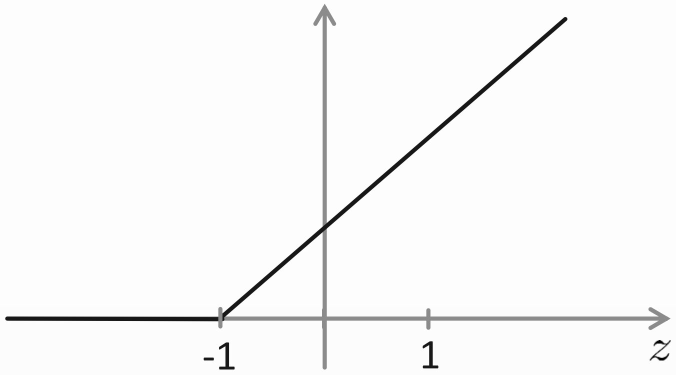
$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{i=1}^{n}\theta_j^2$$

C → controlling the cost trade-off

SVM Hypothesis: Large Margin Intuition

$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{i=1}^{n}\theta_j^2$$





If $y = 1$, we want $\theta^T x \geq 1$ (not just ≥ 0)

If $y = 0$, we want $\theta^T x \leq -1$ (not just < 0)

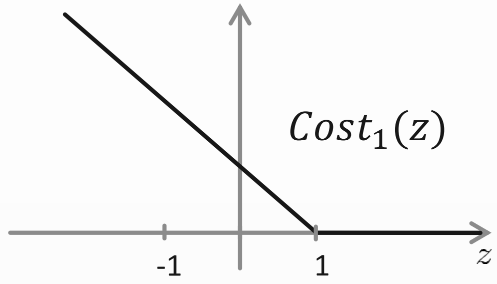
SVM Decision Boundary

$$\min_\theta C \underbrace{\sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)}) \right]}_{=0} + \frac{1}{2}\sum_{i=1}^{n} \theta_j^2$$

Assume
a very large value is chosen

Whenever $y^{(i)} = 1$:

$$\theta^T x \geq 1$$

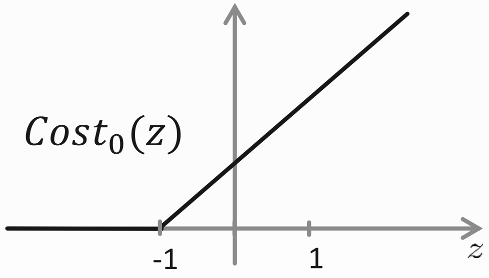


$$\min_\theta \frac{1}{2}\sum_{j=1}^{n} \theta_j^2$$

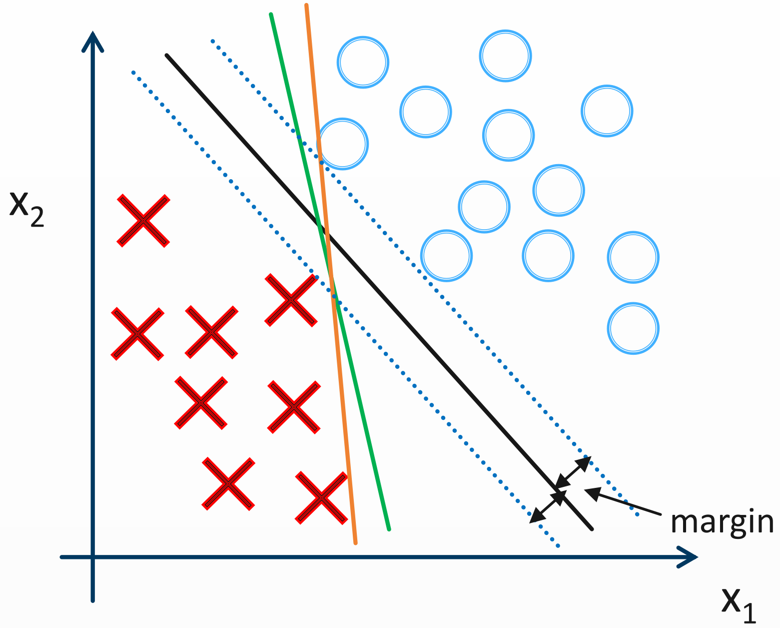
$$\text{s.t.} \quad \theta^T x^{(i)} \geq 1 \quad \text{if } y^{(i)} = 1$$
$$\theta^T x^{(i)} \leq -1 \quad \text{if } y^{(i)} = 0$$

Whenever $y^{(i)} = 0$:

$$\theta^T x \leq -1$$



SVM Decision Boundary: Linearly separable case

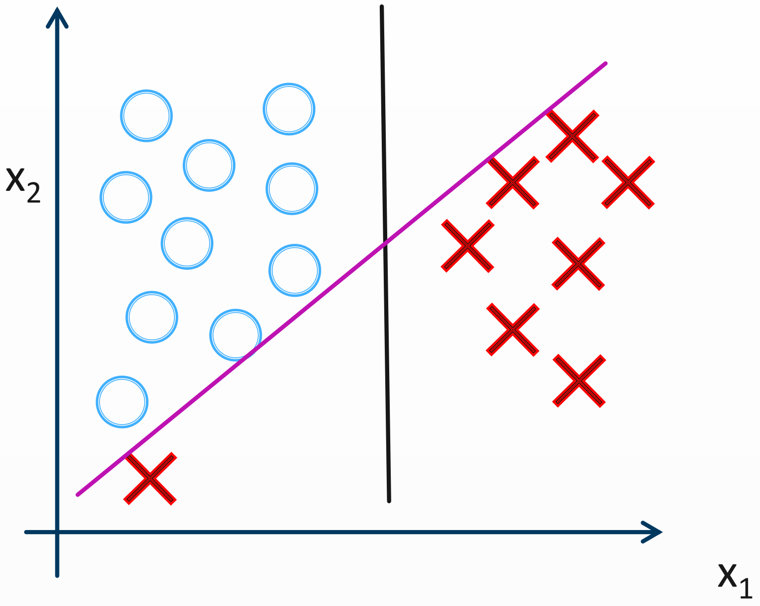


Large margin classifier

- There are many linear decision boundaries that separate the classes
  - But, many of them are not particularly good choices

- The black one (SVM decision boundary) is a more robust separator
  - Mathematically, it has larger margins (or larger minimum distance from any of my training examples)

Large margin classifier in presence of outliers



C is very large

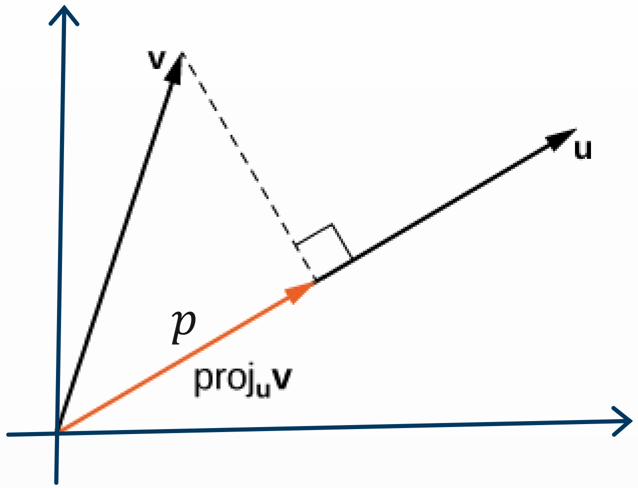
C is not too large

It is a regularization parameter

- If C is very large, SVM will be too sensitive to outliers
- Using not too large values, it will manage to generate a robust decision boundary such as black one

The mathematics behind large margin classification

Vector Inner Product



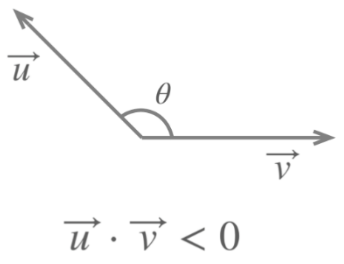
$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$u^T v = ?$$

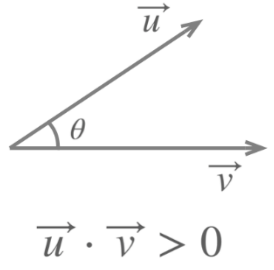
p = (signed) length of projection of vector v onto vector u

$\|u\|$ = length of vector u $\rightarrow$ $\|u\| = \sqrt{u_1^2 + u_2^2}$

$$u^T v = p\ \|u\| \qquad u_1 v_1 + u_2 v_2 = p\ \|u\|$$



$p < 0$



$p > 0$

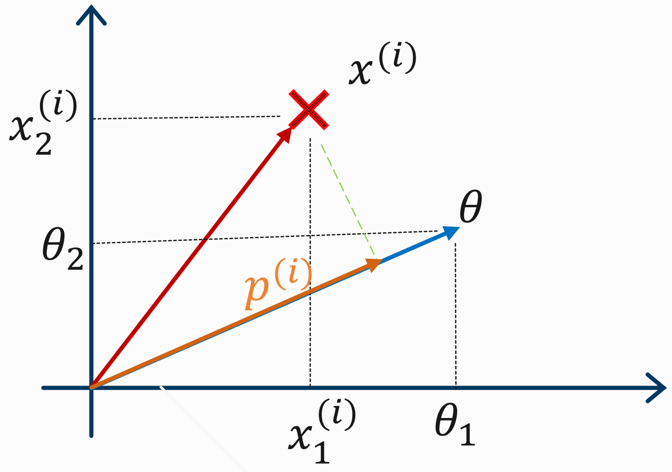
The mathematics behind large margin classification

SVM Decision Boundary

$$\min_{\theta} \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \frac{1}{2}(\theta_1^2+\theta_2^2) = \frac{1}{2}\left(\sqrt{\theta_1^2+\theta_2^2}\right)^2 = \frac{1}{2}\|\theta\|^2$$

$$\begin{aligned}\text{s.t.}\quad & \theta^T x^{(i)} \geq 1 && \text{if } y^{(i)} = 1\\ & \theta^T x^{(i)} \leq -1 && \text{if } y^{(i)} = 0\end{aligned}$$

Simplification $\theta_0 = 0,\ n = 2$



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

$$= \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)}$$

Constraints can be presented in terms of p

The mathematics behind large margin classification

SVM Decision Boundary

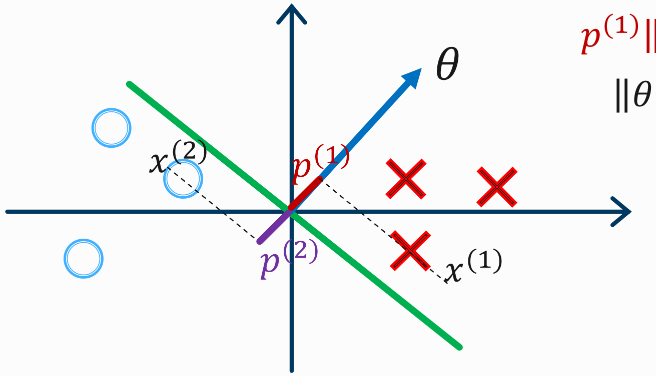
$$\min_{\theta} \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \frac{1}{2}\|\theta\|^2$$

$$\text{s.t.} \quad p^{(i)} \cdot \|\theta\| \geq 1 \qquad \text{if } y^{(i)} = 1$$

$$p^{(i)} \cdot \|\theta\| \leq -1 \qquad \text{if } y^{(i)} = 0$$

where $p^{(i)}$ is the projection of $x^{(i)}$ onto the vector θ.

Simplification: $\theta_0 = 0$



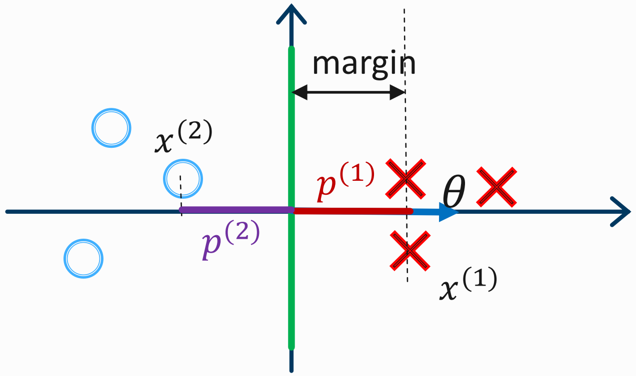
$p^{(1)}\|\theta\| > 1$

$\|\theta\|$ large

$p^{(2)} < 0$

$p^{(2)}\|\theta\| < -1$

$\|\theta\|$ large

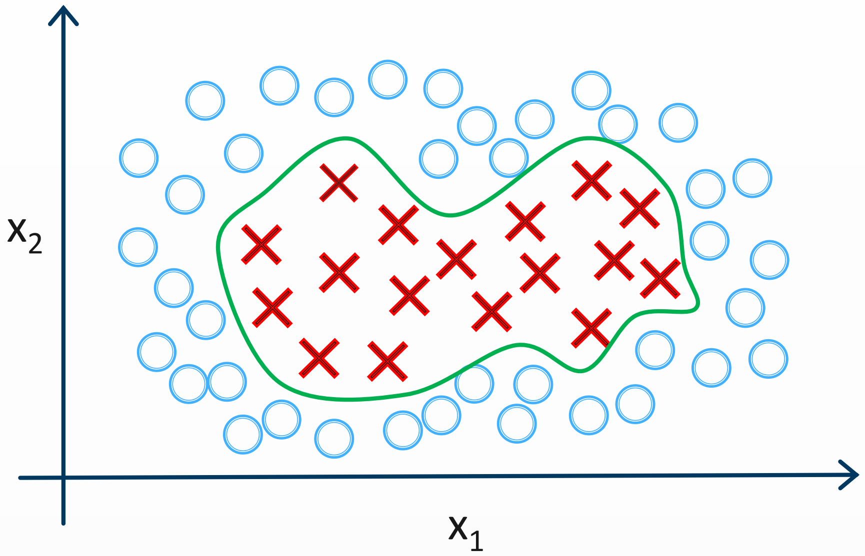


$p^{(1)}\|\theta\| > 1$

$\|\theta\|$ can be smaller

$p^{(2)}\|\theta\| < -1$

Non-linear Decision Boundary

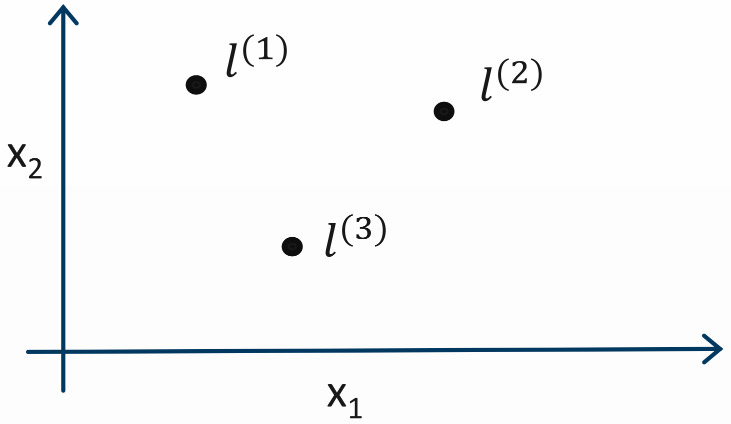


Predict $y = 1$ if

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

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

Kernels



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

Given x:

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

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

$$f_3 = \text{similarity}(x, l^{(3)}) = \exp\left(-\frac{\|x-l^{(3)}\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{j=1}^{n}(x_j - l_j^{(3)})^2}{2\sigma^2}\right)$$

Kernel

Gaussian Kernel

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

$$f_1 \approx exp\left(-\frac{0^2}{2\sigma^2}\right) \approx 1$$

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

$$f_1 \approx exp\left(-\frac{(\text{large number})^2}{2\sigma^2}\right) \approx 0$$

For each landmark,
a new feature:

$l^{(1)} \rightarrow f_1$

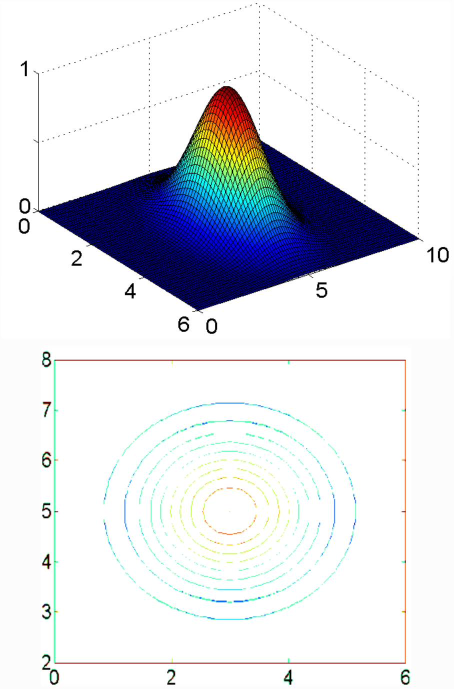
$l^{(2)} \rightarrow f_2$

$l^{(3)} \rightarrow f_3$

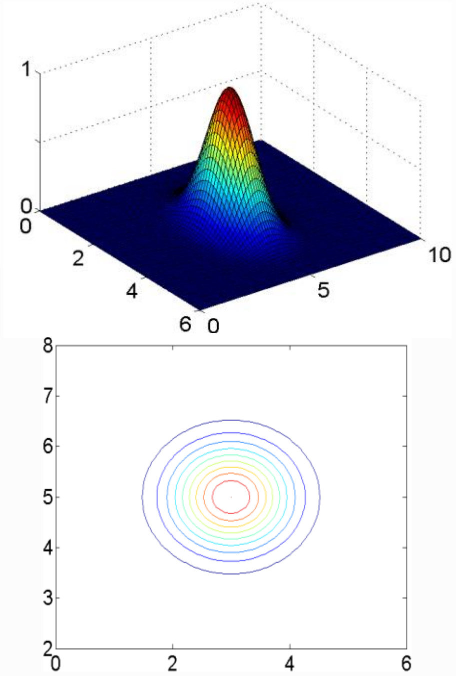
Kernels and Similarity

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

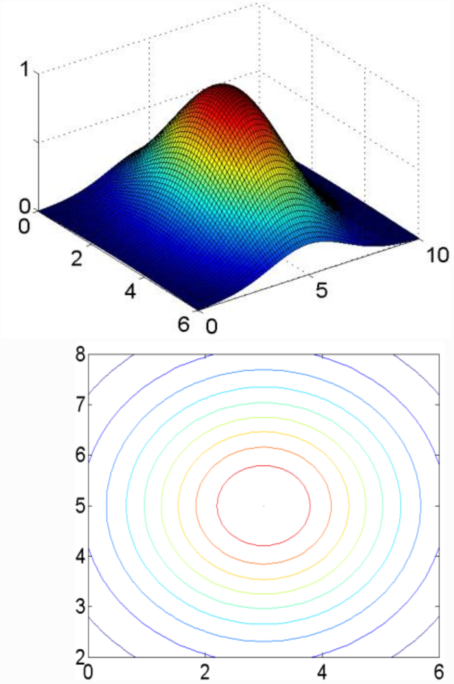
$\sigma^2 = 1$



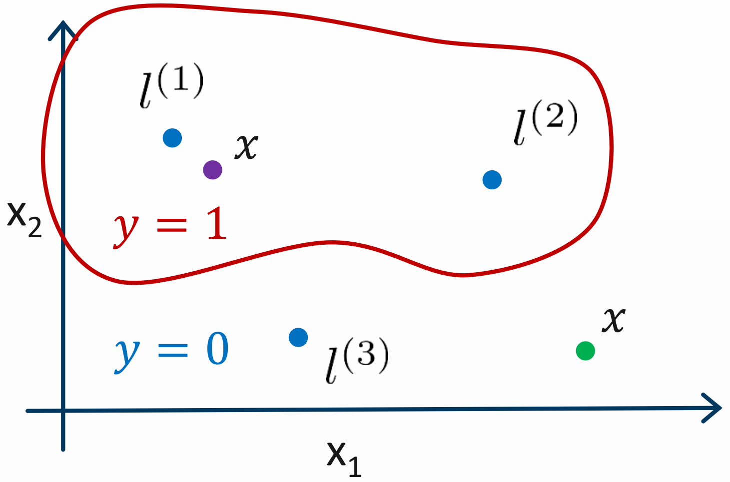
$\sigma^2 = 0.5$



$\sigma^2 = 3$



Kernels and Similarity



Predict "1" when

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

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

$f_1 \approx 1, \quad f_2 \approx 0, \quad f_3 \approx 0$

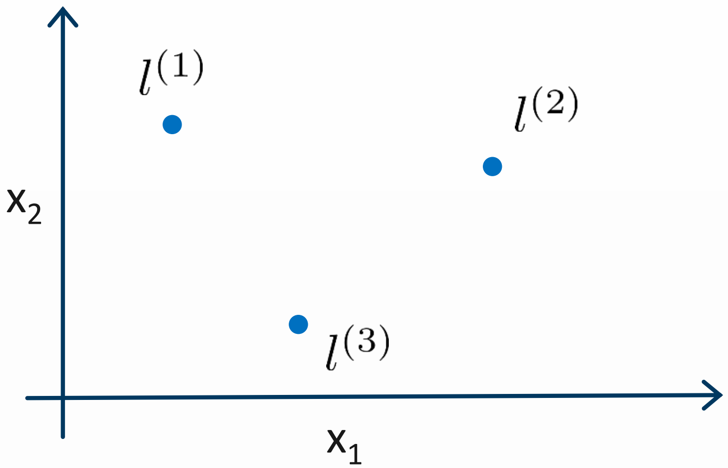
$\theta_0 + \theta_1 \times 1 + \theta_2 \times 0 + \theta_3 \times 0 = 0.5 \geq 0 \quad \rightarrow y = 1$

$f_1 \approx 0, \quad f_2 \approx 0, \quad f_3 \approx 0$

$\theta_0 + \theta_1 \times 0 + \theta_2 \times 0 + \theta_3 \times 0 = -0.5 < 0 \quad \rightarrow y = 0$

We can learn pretty complex non-linear decision boundaries

Choosing the Landmarks

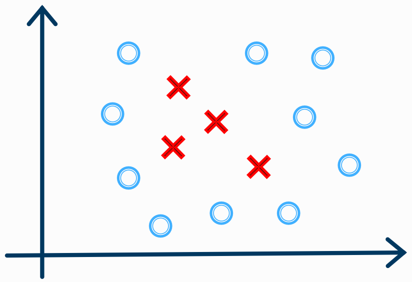


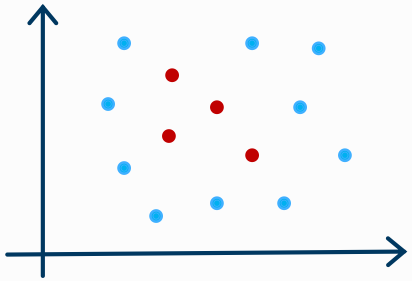
Given x:

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

Predict $y = 1$ if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Where to get $l^{(1)}, l^{(2)}, l^{(3)}, \ldots$?





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:

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

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

$$x^i \rightarrow \quad \begin{aligned} f_1^{(i)} &= sim(x^i, l^1) \\ f_2^{(i)} &= sim(x^i, l^2) \\ &\vdots \\ f_m^{(i)} &= sim(x^i, l^m) \end{aligned}$$

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

SVM with Kernels

Hypothesis: Given x, compute features $f \in \mathbb{R}^{m+1}$
Predict "y=1" if $\theta^T f \geq 0$

Training:

$n = m$

$$\min_\theta C \sum_{i=1}^{m} y^{(i)} cost_1(\theta^T f^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T f^{(i)}) + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

$\theta^T \cancel{x^i}$

Used in this form
to improve scalability

$\theta^T M \theta$

- The optimization problem that the SVM has is a convex opt. problem.
  - You don't need to worry about local optima.

- We can apply kernel idea trick for other algorithms like logistic regression,
  - But the computational tricks that apply for SVM don't generalize well to other algorithms
  - Using kernels with logistic regression will be very slow

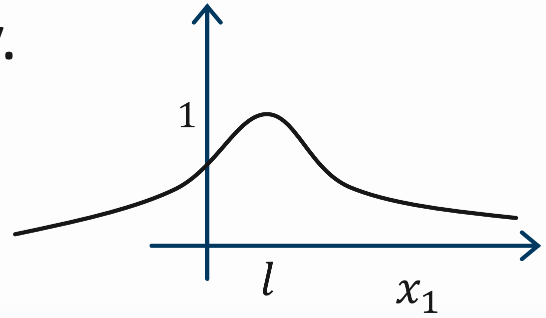
SVM Parameters

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

Small C: Higher bias, low variance. (Large λ)

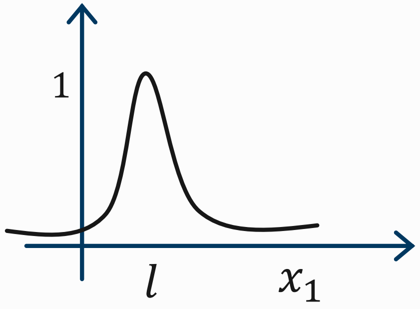
σ^2 Large σ^2: Features f_i vary more smoothly.

Higher bias, lower variance.



Small σ^2: Features f_i vary less smoothly.

Lower bias, higher variance.



Using an SVM

Use SVM software package (e.g. liblinear, libsvm, …) to solve for parameters θ.

Need to specify:

Choice of parameter C.

Choice of kernel (similarity function):

E.g. No kernel (“linear kernel”)

Predict “y = 1” if $\theta^T x \geq 0$

Gaussian kernel:

$f_i = \exp\left(-\frac{||x - l^{(i)}||^2}{2\sigma^2}\right)$, where $l^{(i)} = x^{(i)}$.

Need to choose σ^2.

Using an SVM

Kernel (similarity) functions:

$x^{(i)}$ $l^{(j)}$

```
function f = kernel(x1,x2)
```

$$f = \exp\left(-\frac{||\mathbf{x1} - \mathbf{x2}||^2}{2\sigma^2}\right)$$

```
return
```

Note: Do perform feature scaling before using the Gaussian kernel.

$$\|x - l\|^2 = (x_1 - l_1)^2 + (x_2 - l_2)^2 + \cdots + (x_n - l_n)^2$$

House pricing example, 1000 ft^2 1-5 bedrooms

This can dominate, perform feature scaling

Other Choices of Kernel

Note: Not all similarity functions $\mathrm{similarity}(x, l)$ make valid kernels. (Need to satisfy technical condition called "Mercer's Theorem" to make sure SVM packages' optimizations run correctly, and do not diverge).

Many off-the-shelf kernels available:

- Polynomial kernel: $(x^T l + constant)^{degree}$

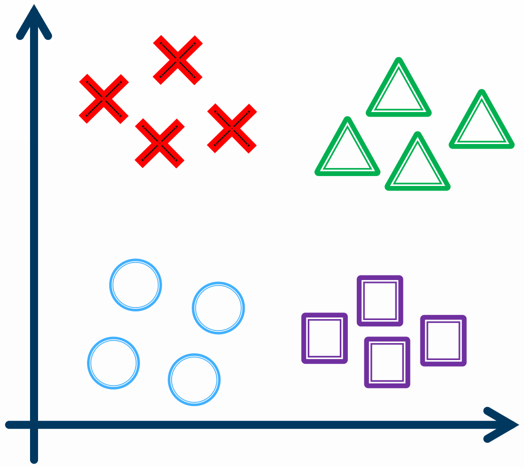
$$(x^T l + 1)^2, \quad (x^T l + 5)^3$$

  - Usually perform worse than Gaussian Kernel
  - It is not used that often

- Esoteric kernels: String kernel, chi-square kernel, histogram intersection kernel, …

Multi-class Classification

Multi-class classification



$$y \in \{1, 2, 3, \dots, K\}$$

Many SVM packages already have built-in multi-class classification functionality.

Otherwise, use one-vs.-all method. (Train K SVMs, one to distinguish $y = i$ from the rest, for $i = 1, 2, \dots, K$), get $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(K)}$
Pick class i with largest $(\theta^{(i)})^T x$

Logistic Regression vs. SVMs

$n =$number of features ($x \in \mathbb{R}^{n+1}$), $m =$number of training examples
If n is large (relative to m):
Use logistic regression, or SVM without a kernel (“linear kernel”)

If n is small, m is intermediate:
Use SVM with Gaussian kernel

If n is small, $m$is large:
Create/add more features, then use logistic regression or SVM without a kernel

Neural network likely to work well for most of these settings, but may be slower to train.

Strengths and Weaknesses

- SVMs allow for complex decision boundaries, even if the data has only a few features.
- They work well on low-dimensional and high-dimensional data (i.e., few and many features), but don't scale very well with the number of samples.
  - Running an SVM on data with up to 10,000 samples might work well, but working with datasets of size 100,000 or more can become challenging in terms of runtime and memory usage.
- Another downside of SVMs is that they require careful preprocessing of the data and tuning of the parameters.
- It might be worth trying SVMs, particularly if all of your features represent measurements in similar units (e.g., all are pixel intensities) and they are on similar scales. Or, you should normalize the features.
- It can be difficult to understand why a particular prediction was made by an SVM model, and it might be tricky to explain the model to a non-expert.

References

- A. Ng. Machine Learning, Lecture Notes.
- I. Goodfellow, Y. Bengio and A. Courville, “Deep Learning”, 2016.