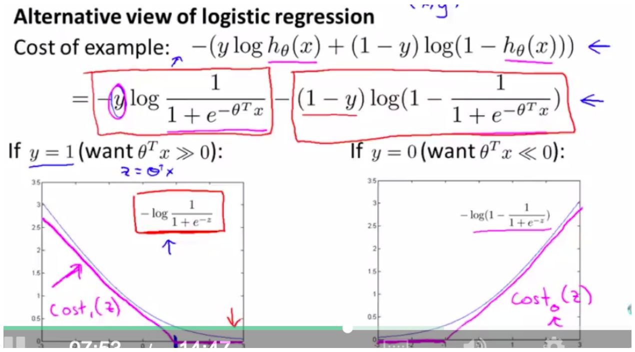
- Optimization objective
- Alternative view of logistic regression
    - sigmoid activation
    - if y = 1
        - we want h_theta(x) close to 1
        - Theta’x >> 0
    - if y = 0
        - we want h_theta(x) close to 0
        - Theta’x << 0
- Alternative view of logistic regression
    - new cost function gives support vector machine advantages and easier optimization problem
    - two new cost functions one when y = 1 and one when y = 0



Alternative view of logistic regression

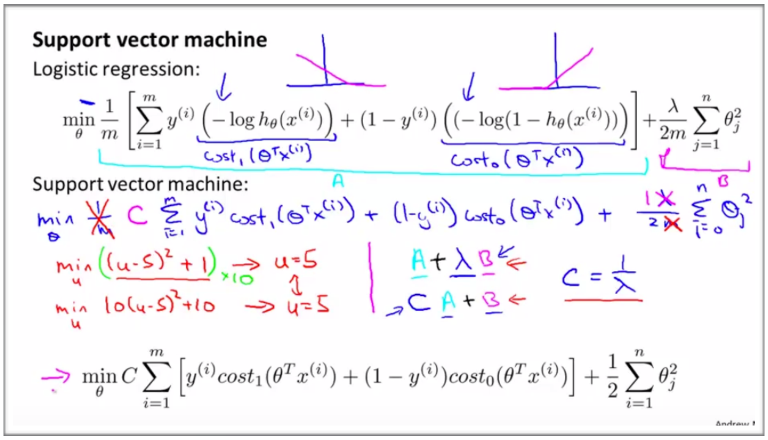
Cost of example: $-(y\log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$

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

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

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

- Support vector machine
    - Logistic regression:



Support vector machine

Logistic regression:

$$\min_\theta \frac{1}{m}\underbrace{\left[\sum_{i=1}^m y^{(i)}\underbrace{\left(-\log h_\theta(x^{(i)})\right)}_{cost_1(\theta^T x^{(i)})} + (1-y^{(i)})\underbrace{\left((-\log(1-h_\theta(x^{(i)}))\right)}_{cost_0(\theta^T x^{(i)})}\right]}_{A} + \underbrace{\frac{\lambda}{2m}\sum_{j=1}^n \theta_j^2}_{B}$$

Support vector machine:

$$\min_\theta \cancel{\frac{1}{m}}\, C \sum_{i=1}^m y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)}) + \frac{1\cancel{\lambda}}{2\cancel{m}}\sum_{i=0}^n \theta_j^2$$

$\min_u ((u-5)^2+1) \times 10 \rightarrow u=5$

$\min_u 10(u-5)^2+10 \rightarrow u=5$

$A + \lambda B \leftarrow$

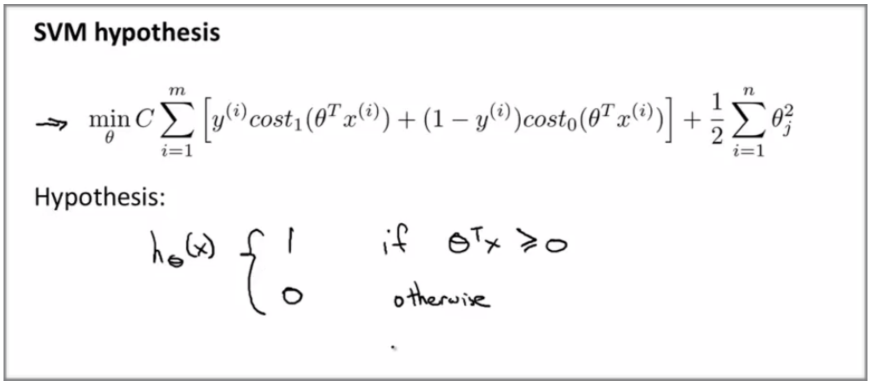
$C A + B \leftarrow$

$C = \frac{1}{\lambda}$

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

        - replace the cost function with the new cost functions for support vector machines
        - Logistic regression uses lambda to control the relative weighting between the first and second terms
            - large value of lambda gives B a very large weight
        - Support vector machine uses c as an alternative parameter
            - we then minimize CA + B
                - giving C a large value gives A a very large weight
        - Different way of controlling or parameterizing the weights
            - c = 1/lambda

- SVM hypothesis
  - does not output a probability
  - just makes a direct prediction if y = 1 or y = 0



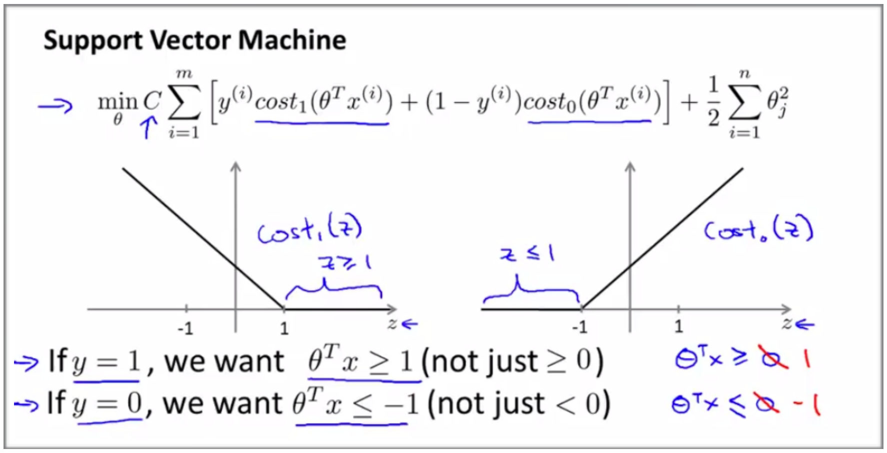
SVM hypothesis

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

Hypothesis:

$$h_\theta(x) \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

- Large Margin Intuition
- Support vector machine



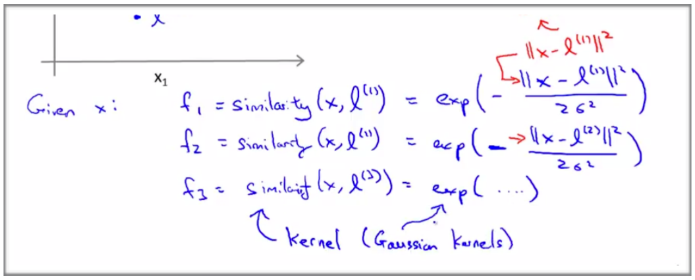
Support Vector Machine

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

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

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

- SVM Decision Boundary
  - set the constant C to be a very large value
  - if C is large highly motivated to choose a term to make sure the first term is equal to zero in the objective function
- SVM Decision Boundary: Linearly separable case
  - margin minimum distance to any of the training examples
  - large margin classifier
- Large Margin classifier in presence of outliers
  - if C is very large with change the decision the boundary for an outlier
  - if C is not very large the SVM will not change the decision boundary for an outlier
- Kernels
  - Given x compute new feature depending on proximity to landmarks l(1), l(2), and l(3)



Given x:

$\|x - l^{(1)}\|^2$

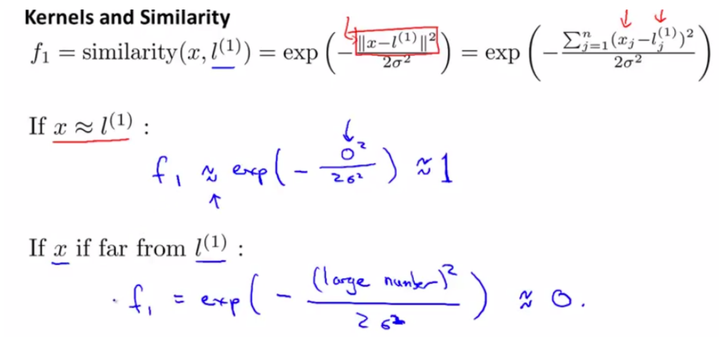
$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$

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

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

Kernel (Gaussian kernels)

- Kernels and Similarity
    - measure how similar x is to one of the landmarks



Kernels and Similarity

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

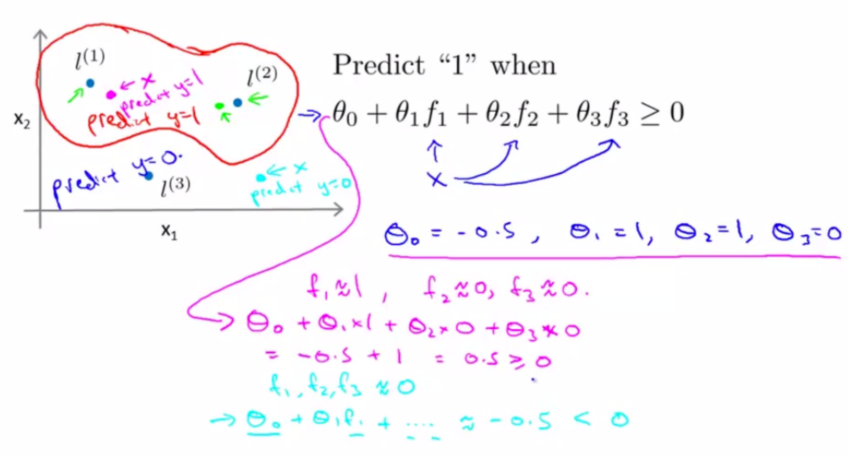
If $\underline{x \approx l^{(1)}}$:

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

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

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

- each landmark can now compute new features given the corresponding landmark
- varying sigma
    - .5 the feature falls to zero more rapidly
    - 3 the value of the feature falls away much more slowly



Predict "1" when

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

x

$\underline{\theta_0 = -0.5, \quad \theta_1 = 1, \quad \theta_2 = 1, \quad \theta_3 = 0}$

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

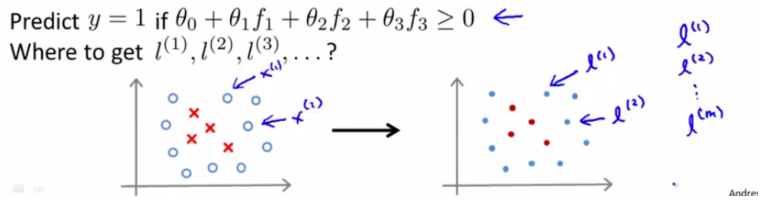
$\rightarrow \theta_0 + \theta_1 \times 1 + \theta_2 \times 0 + \theta_3 \times 0$

$= -0.5 + 1 \quad = 0.5 \geq 0$

$f_1, f_2, f_3 \approx 0$

$\rightarrow \underline{\theta_0} + \theta_1 \underline{f_1} + \underline{\cdots} \approx -0.5 < 0$

- Kernels 2
    - landmarks allowed us to define the similarity function or the kernel
    - take the examples and for every training example we are just going to call it put landmarks as the same locations as the training examples
    - features are going to measure how close something is as seen in training set

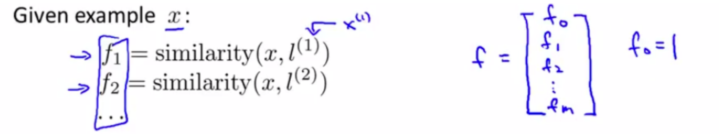


Predict $y = 1$ if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$ ←

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

- SVM with Kernels
    - Given m training examples
    - choose m landmarks

- Given example x:



Given example $\underline{x}$:

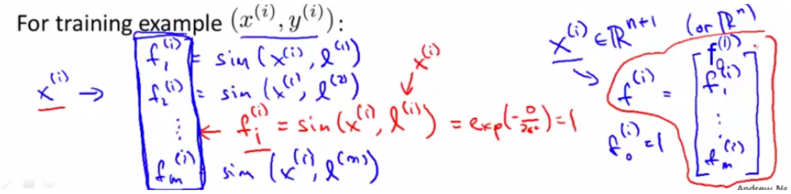
$\rightarrow \boxed{f_1} = \text{similarity}(x, l^{(1)})$ $\leftarrow x^{(i)}$

$\rightarrow \boxed{f_2} = \text{similarity}(x, l^{(2)})$

$\ldots$

$f = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$ $\quad f_0 = 1$

- For training example



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

$\underline{x^{(i)}} \rightarrow$

$f_1^{(i)} = sim(x^{(i)}, l^{(1)})$

$f_2^{(i)} = sim(x^{(i)}, l^{(2)})$

$\vdots \leftarrow \underline{f_i^{(i)}} = sim(\overset{x^{(i)}}{x^{(i)}}, l^{(i)}) = exp(-\frac{0}{2\sigma^2}) = 1$

$f_m^{(i)} \quad sim(x^{(i)}, l^{(m)})$

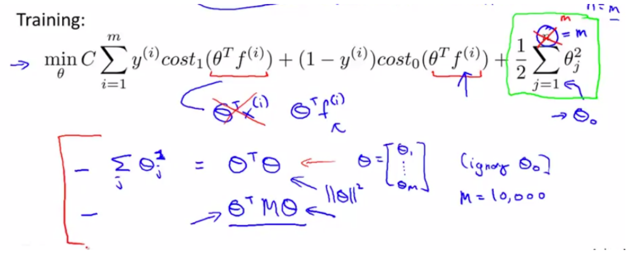
$\underline{x^{(i)}} \in \mathbb{R}^{n+1}$ (or $\mathbb{R}^n$)

$\rightarrow f^{(i)} = \begin{bmatrix} f_0^{(i)} \\ f_1^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix}$

$f_0^{(i)} = 1$

Andrew Ng

- SVM with Kernels
  - Hypothesis: Given x, compute features f element of R m+1
  - Training:
    - to obtain the parameters for the support vector machine solve the minimization problem



Training:

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

$n = m$

$\cancel{\theta^T x^{(i)}} \quad \theta^T f^{(i)}$

$\rightarrow \theta_0$

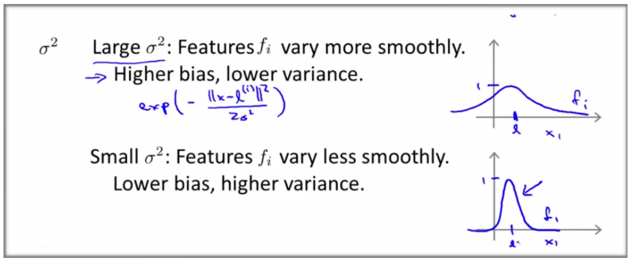
$- \quad \sum_j \theta_j^2 = \theta^T \theta \leftarrow \theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_m \end{bmatrix}$ (ignoring θ_0)

$\|\theta\|^2$

$M = 10,000$

$- \quad \rightarrow \underline{\theta^T M \theta} \leftarrow$

- SVM parameters
  - C = 1 lambda
  - Large C: Lower bias, high variance (small lambda)
  - Small C: Higher bias, low variance (large lambda)



σ^2

Large σ^2: Features f_i vary more smoothly.

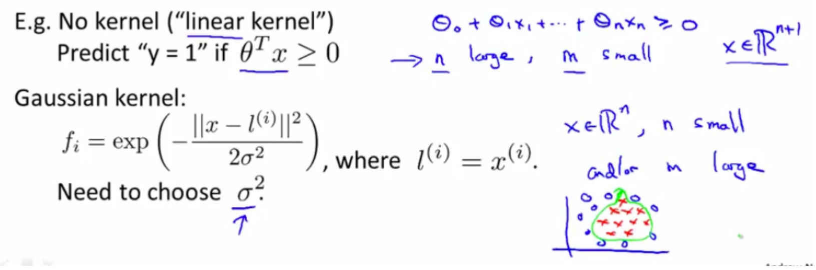
$\rightarrow$ Higher bias, lower variance.

$exp(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2})$

Small σ^2: Features f_i vary less smoothly.

Lower bias, higher variance.

- Using an SVM
- Support vector machine
  - choice of parameter C
  - choice of kernel
  - e.g. No kernel (“linear kernel”)
    - predict “y = 1” if theta_t dot x >= 0



E.g. No kernel (“linear kernel”)

Predict “y = 1” if $\underline{\theta^T x} \geq 0$ $\quad \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \geq 0$

→ $\underline{n}$ large, $\underline{m}$ small $\quad \underline{x \in \mathbb{R}^{n+1}}$

Gaussian kernel:

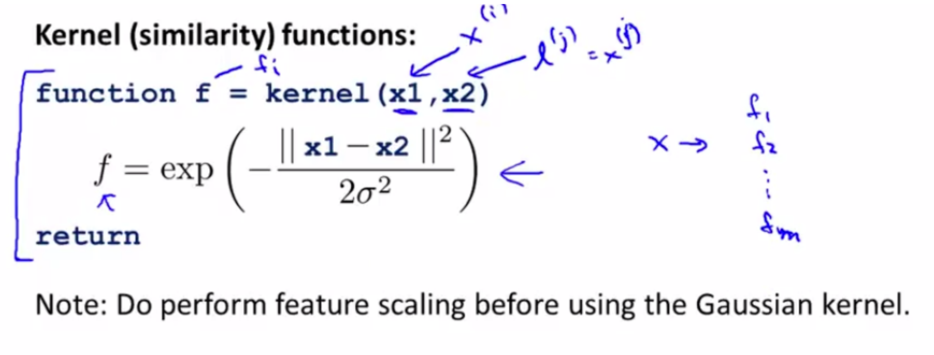
$$f_i = \exp\left(-\frac{||x - l^{(i)}||^2}{2\sigma^2}\right), \text{ where } l^{(i)} = x^{(i)}.$$

$x \in \mathbb{R}^n$, n small

and/or m large

Need to choose σ^2.

- Kernel (similarity) functions:



Kernel (similarity) functions:

$x^{(i)}$ $\quad l^{(j)} = x^{(j)}$

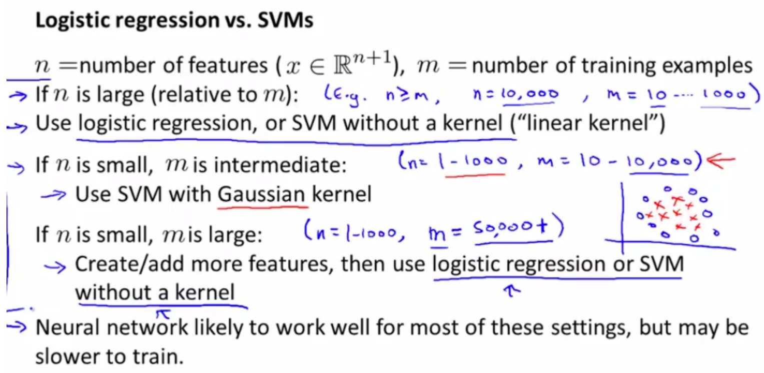
f_i

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

    f = exp(-||x1 - x2||^2 / (2σ^2))

return
```

$x \rightarrow \quad f_1, f_2, \ldots, f_m$

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

  - feature scaling is important
- Other choices of kernel
  - Not all similarity functions 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).
  - polynomial kernel: k(x,l) = (x’l)^2, (x’l)^3, (x’l + 1)^3, (x’l + 5)^4
  - more esoteric: String kernel, chi-square kernel, histogram intersection kernel
- Logistic regression VS SVMS



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): (e.g. $n \geq m$, $n = 10{,}000$, $m = 10 \ldots 1000$)

→ Use logistic regression, or SVM without a kernel (“linear kernel”)

→ If n is small, m is intermediate: ($n = 1 - 1000$, $m = 10 - 10{,}000$) ←

→ Use SVM with Gaussian kernel

If n is small, m is large: ($n = 1 - 1000$, $m = 50{,}000+$)

→ 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.