Using an SVM

SVM in Practice

Support Vector Machines

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

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):

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 $\theta^T x \geq 0$

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 $\theta^T x \geq 0$

$\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \geq 0$

n large, m small

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 $\theta^T x \geq 0$

$\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \geq 0$

$x \in \mathbb{R}^{n+1}$

$\rightarrow$ n large, m small

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.

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 $\theta^T x \geq 0$

$\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \geq 0$

→ n large, m small

$x \in \mathbb{R}^{n+1}$

Gaussian kernel:

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

Need to choose σ^2.

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

and/or m large

Kernel (similarity) functions:

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

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

```
return
```

Kernel (similarity) functions:

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

f_i

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

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

```
return
```

$x \rightarrow$ $f_1, f_2, \vdots, f_m$

Kernel (similarity) functions:

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

f_i

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

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

```
return
```

$x \rightarrow$ f_1, f_2, $\vdots$, f_m

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

Kernel (similarity) functions:

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

f_i

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

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

```
return
```

$x \rightarrow \begin{matrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{matrix}$

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

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

$\boxed{\|x - l\|^2}$

$v = x - l$

$\|v\|^2 = v_1^2 + v_2^2 + \cdots + v_n^2$

$= \underbrace{(x_1 - l_1)^2}_{1000 \text{ feet}^2} + \underbrace{(x_2 - l_2)^2}_{1\text{-}5 \text{ bedrooms}} + \cdots + (x_n - l_n)^2 .$

Other choices of kernel

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

Other choices of kernel

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

$(x^T l + \text{constant})^{\text{degree}}$

Many off-the-shelf kernels available:

- Polynomial kernel: $k(x, l) = (x^T l)^2$ $+0$

  $(x^T l)^3$, $(x^T l + 1)^{3}$, $(x^T l + 5)^{4}$

Other choices of kernel

Note: Not all similarity functions $\text{similarity}(x, l)$ 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:

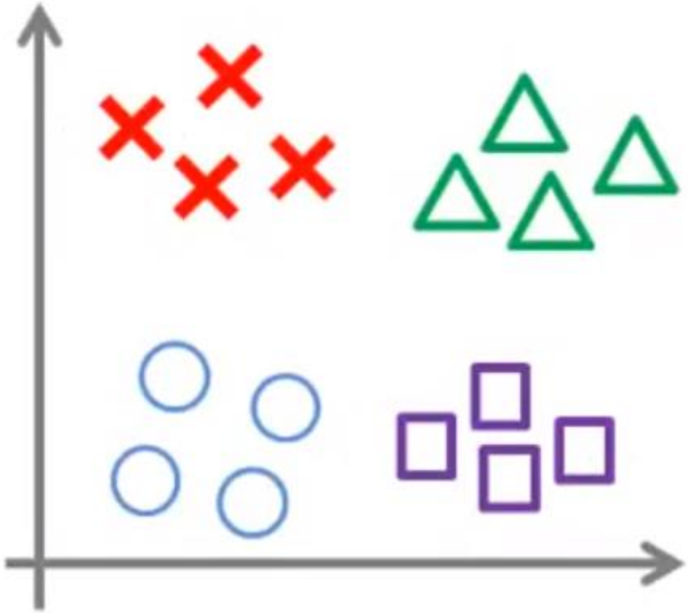
- Polynomial kernel: $k(x,l) = (x^T l)^2$ $+0$

  $(x^T l + \text{constant})^{\text{degree}}$

  $(x^T l)^3, \quad (x^T l + 1)^3, \quad (x^T l + 5)^4$

- More esoteric: String kernel, chi-square kernel, histogram intersection kernel, ...

Multi-class classification



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

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 $i = 1, 2, \ldots, K$), get $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(K)}$

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")

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. $n \geq m$, $n = 10,000$, $m = 10 \cdots 1000$)

Use logistic regression, or SVM without a kernel ("linear kernel")

Logistic regression vs. SVMs

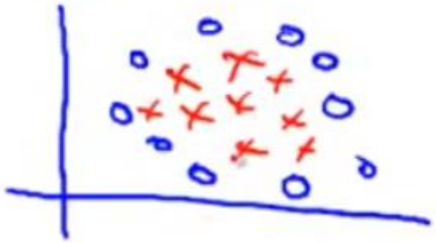
$n =$ number of features ($x \in \mathbb{R}^{n+1}$), $m =$ number of training examples

If n is large (relative to m): (e.g. $n \geq m$, $n = 10{,}000$, $m = 10 \cdots 1000$)

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



Andrew Ng

Logistic regression vs. SVMs

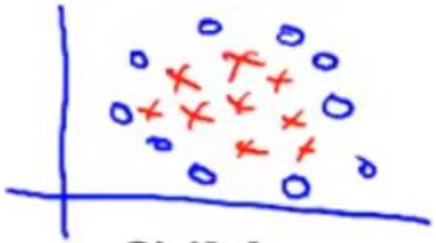
$n =$ number of features ($x \in \mathbb{R}^{n+1}$), $m =$ number of training examples

→ If n is large (relative to m): (e.g. $n \geq m$, $n = 10{,}000$, $m = 10 \cdots 1000$)

→ 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

Packages are good but they struggle if there are too many data.

Logistic regression vs. SVMs

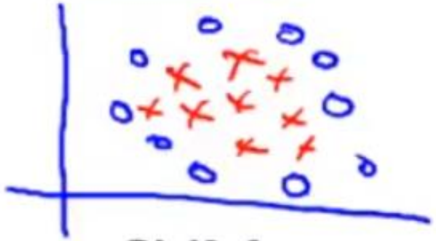
$n =$ number of features ($x \in \mathbb{R}^{n+1}$), $m =$ number of training examples

If n is large (relative to m): (e.g. $n \geq m$, $n = 10{,}000$, $m = 10 \cdots 1000$)

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.

Andrew Ng