Support Vector Machines

Teeradaj Racharak (เอ็กซ์)
r.teeradaj@gmail.com

Introduction

Thus far, we have studied machine learning models that are relatively easy to analyze and are optimal, when the assumptions they make are satisfied.

For example, GDA assumes the conditional distribution $p(\boldsymbol{x} \mid y)$ is a multivariate Gaussian.

What happens when the assumptions are violated?

Now, we will look at support vector machines (SVMs), which are more flexible and widely applicable than the methods we have looked so far.

Although deep neural networks have received the most attention recently, many still believe that SVM is the best 'off-the-shelf' supervised classifiers.

Introduction

SVMs are based on the idea of maximum margin classification.

Suppose the yellow points are training data from class 1 and the cyan points are training data from class 0.

$\theta^T \boldsymbol{x} = 0$ is a separating hyperplane or decision boundary between the two classes.

Point A is furthest from the decision boundary. We should be confident to predict class 1 for point A.

Point C is more ambiguous.

This observation leads to the principle of maximizing the margin !

SVM Decision Boundary: Linearly Separatable Case

SVM Decision Boundary: Linearly Separatable Case

called ‘margin’

Note: SVM tries to separate the data with margins as large as possible.

SVM as Large Margin Classifier

called ‘margin’

Note: SVM tries to separate the data with margins as large as possible.

Optimization Objective

Alternative View of Logistic Regression

Recall that in logistic regression, we model $p(y = 1 \,|\, \boldsymbol{x}; \boldsymbol{\theta})$ by

$$h_\theta(x) = g(\boldsymbol{\theta}^T \boldsymbol{x})$$

The logistic classification rule is:

$$y^{\textbf{pred}}(\boldsymbol{x}) = \begin{cases} 1 & \textbf{if } h_\theta(\boldsymbol{x}) \geq 0.5 \\ 0, & \textbf{otherwise} \end{cases}$$

Informally, our goal should be to find $\boldsymbol{\theta}$ such that $\boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \gg 0$ for all i with $y^{(i)} = 1$ and $\boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \ll 0$ for all i with $y^{(i)} = 0$.

Alternative View of Logistic Regression

Hypothesis function:

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

Let's think about what we'd like logistic regression to do:

- If $y = 1$, then we want $h_\theta(x) \approx 1$ *i.e.* $\theta^T x \gg 0$
- If $y = 0$, then we want $h_\theta(x) \approx 0$ *i.e.* $\theta^T x \ll 0$

Alternative View of Logistic Regression

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

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

Case 1 ($y = 1 \implies \theta^T x \gg 0$):

$z = \theta^T x$

in SVM

Alternative View of Logistic Regression

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

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

Case 1 ($y = 1 \Longrightarrow \theta^T x \gg 0$):

$z = \theta^T x$

in SVM

Alternative View of Logistic Regression

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

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

Case 2 ($y = 0 \Longrightarrow \theta^T x \ll 0$):

$z = \theta^T x$

in SVM

Support Vector Machine

Logistic regression:

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

Support Vector Machine

Logistic regression:

$$J(\theta) = \min_{\theta} \frac{1}{m}\left[\Sigma_{i=1}^{m} y^{(i)}\underbrace{\left(-\log h_\theta(x^{(i)})\right)}_{\textbf{cost}_1(\theta^T x^{(i)})} + (1-y^{(i)})\underbrace{\left(-\log(1-h_\theta(x^{(i)}))\right)}_{\textbf{cost}_0(\theta^T x^{(i)})}\right] + \frac{\lambda}{2m}\Sigma_{j=1}^{n}\theta_j^2$$

Support Vector Machine

Logistic regression:

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

Support vector machine:

$$J(\theta) = \min_{\theta} \frac{1}{m}\left[\Sigma_{i=1}^{m} y^{(i)}\mathbf{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\mathbf{cost}_0(\theta^T x^{(i)})\right] + \frac{\lambda}{2m}\Sigma_{j=1}^{n}\theta_j^2$$

Now, Let's rewrite the above equation to correspond with SVM's convention !

Support Vector Machine

1. Let's remove 'm' from the equation !

Cost function:

$$J(\theta) = \min_{\theta} \cancel{\frac{1}{m}} \left[\Sigma_{i=1}^{m} y^{(i)} \mathbf{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \mathbf{cost}_0(\theta^T x^{(i)}) \right] + \frac{\lambda}{2\cancel{m}} \Sigma_{j=1}^{n} \theta_j^2$$

Why is this reasonable?

Let's think about:

- $\min_{u} \left((u-5)^2 + 1 \right) \Longrightarrow 2(u-5) \cdot 1 = 0 \Longleftrightarrow u = 5$
- $\min_{u} 10 \left((u-5)^2 + 1 \right) \Longrightarrow 2 \cdot 10 \cdot (u-5) \cdot 1 = 0 \Longrightarrow u = 5$

Support Vector Machine

2. Let's reorganize the equation a bit !

Cost function:

$$J(\theta) = \min_{\theta} \cancel{\frac{1}{m}} \underbrace{\left[\Sigma_{i=1}^{m} y^{(i)} \mathbf{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \mathbf{cost}_0(\theta^T x^{(i)})\right]}_{\mathbf{A}} + \underbrace{\frac{\lambda}{2\cancel{m}} \Sigma_{j=1}^{n} \theta_j^2}_{\mathbf{B}}$$

i.e. **in logistic regression, we write:** $A + \lambda B$

whereas in SVM, we write: $CA + B$

Noted that C is not necessary to be $1/\lambda$ **!**

Support Vector Machine

Logistic regression:

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

Support vector machine:

$$J(\theta) = \min_{\theta} C\Sigma_{i=1}^{m}\left[y^{(i)}\mathbf{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\mathbf{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$$

Question

- Consider the following minimization problems:

1. $J(\theta) = \min_{\theta} \frac{1}{m}\left[\Sigma_{i=1}^{m} y^{(i)}\mathbf{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)})\mathbf{cost}_0(\theta^T x^{(i)})\right] + \frac{\lambda}{2m}\Sigma_{j=1}^{n}\theta_j^2$
2. $J(\theta) = \min_{\theta} C\left[\Sigma_{i=1}^{m} y^{(i)}\mathbf{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)})\mathbf{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$

These two optimization problems will give the same value of θ (*i.e.* the same value of θ gives the optimal solution to both problems) if:

(i) $C = \lambda$

(ii) $C = -\lambda$

(iii) $C = 1/\lambda$

(iv) $C = 2/\lambda$

Support Vector Machine

Unlike logistic regression, SVM doesn't have probabilistic interpretation !

Cost function:

$$J(\theta) = \min_{\theta} C\Sigma_{i=1}^{m}\left[y^{(i)}\textbf{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)})\textbf{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$$

Hypothesis function:

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

Large Margin Classifier

Intuition

Cost function:

$$J(\theta) = \min_{\theta} C\Sigma_{i=1}^{m}\left[y^{(i)}\mathbf{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\mathbf{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$$

- 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)

Next, what if we set C to be a very large value *e.g.* 100,000 ?

SVM Decision Boundary

$$\min_{\theta} \underbrace{C}_{10^5} \underbrace{\Sigma_{i=1}^{m}\left[y^{(i)}\textbf{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\textbf{cost}_0(\theta^T x^{(i)})\right]}_{\approx 0} + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$$

Whenever $y^{(i)} = 1,\ \theta^T x^{(i)} \geq 1$

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

SVM Decision Boundary

$$\min_{\theta} \underbrace{C}_{10^5} \underbrace{\Sigma_{i=1}^{m}\left[y^{(i)}\mathbf{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\mathbf{cost}_0(\theta^T x^{(i)})\right]}_{\approx 0} + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$$

Whenever $y^{(i)} = 1,\ \theta^T x^{(i)} \geq 1$

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

$$\min_{\theta} C \cdot 0 + \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$$

subject to - $\theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$

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

When we solve this optimization problem,
we'll get an interesting decision boundary !

i.e. when C is very big, SVM is sensitive to the outlier;
decreasing C makes it becomes LESS sensitive.

SVM Decision Boundary: Linearly Separatable Case

SVM Decision Boundary: Linearly Separatable Case

called ‘margin’

Note: SVM tries to separate the data with margins as large as possible.

SVM as Large Margin Classifier

called ‘margin’

Note: SVM tries to separate the data with margins as large as possible.

Large Margin Classifier in Presence of Outliers

SVM tries to find a decision boundary that has the widest distance (*margin* from the samples)

Large Margin Classifier in Presence of Outliers

Let's add an outlier !

When C is very big, SVM is sensitive to the outlier; decreasing C makes it becomes LESS sensitive.

If the data is not 'linearly separable', then SVM can still do the right thing.

Question

- Consider the training set, where 'x' denotes examples (y = 1) and 'o' denotes negative examples (y = 0). Suppose you train an SVM (which will predict 1 when $\theta_0 + \theta_1 x_1 + \theta_2 x_2 \geq 0$). What values might the SVM give for θ_0, θ_1, and θ_2?

(i) $\theta_0 = 3,\ \theta_1 = 1,\ \theta_2 = 0$

(ii) $\theta_0 = -3,\ \theta_1 = 1,\ \theta_2 = 0$

(iii) $\theta_0 = 3,\ \theta_1 = 0,\ \theta_2 = 1$

(iv) $\theta_0 = -3,\ \theta_1 = 0,\ \theta_2 = 1$

Mathematics Behind Large Margin Classification

Preliminary: Vector Inner Product

Given $\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, compute $\boldsymbol{u}^T\boldsymbol{v} = ?$

($\boldsymbol{u}^T\boldsymbol{v}$ is called 'vector inner product')

Preliminary: Vector Inner Product

Given $\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, compute $\boldsymbol{u}^T\boldsymbol{v} = ?$

($\boldsymbol{u}^T\boldsymbol{v}$ is called 'vector inner product')

In Vector, we can quantify a certain vector by identifying its euclidean length (or its norm)

$$\underset{\substack{\cap \\ \mathbb{R}}}{\|\boldsymbol{u}\|} = \sqrt{u_1^2 + u_2^2} \quad \text{(by Pythagoras theorem)}$$

Preliminary: Vector Inner Product

Given $\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, compute $\boldsymbol{u}^T\boldsymbol{v} = ?$

($\boldsymbol{u}^T\boldsymbol{v}$ is called 'vector inner product')

From the figure,

p = the length of projection of $\boldsymbol{v}$ onto $\boldsymbol{u}$

It's possible to show that

$$\boldsymbol{u}^T\boldsymbol{v} = p \cdot \|\boldsymbol{u}\| = u_1v_1 + u_2v_2$$

$$= \boldsymbol{v}^T\boldsymbol{u}$$

Preliminary: Vector Inner Product

Given $\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, compute $\boldsymbol{u}^T\boldsymbol{v} = ?$

($\boldsymbol{u}^T\boldsymbol{v}$ is called 'vector inner product')

From the figure,

p = the length of projection of $\boldsymbol{v}$ onto $\boldsymbol{u}$

It's possible to show that

$$\boldsymbol{u}^T\boldsymbol{v} = p \cdot \|\boldsymbol{u}\|$$

Note: p is negative if the angle between $\boldsymbol{u}$ and $\boldsymbol{v} > 90$

SVM Decision Boundary

Let's simplify the equation so that it becomes easier to analyze !

- $J(\theta) = \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2$ s.t. - $\theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$
  - $\theta^T x^{(i)} \leq -1$ if $y^{(i)} = 0$
- $\theta_0 = 0$
- $n = 2$

SVM Decision Boundary

Let's simplify the equation so that it becomes easier to analyze !

- $J(\theta) = \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2 = \frac{1}{2}(\theta_1^2 + \theta_2^2) = \frac{1}{2}(\underbrace{\sqrt{\theta_1^2 + \theta_2^2}}_{= \|\boldsymbol{\theta}\|})^2$ where $\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$

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

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

- $\theta_0 = 0$

- $n = 2$

SVM Decision Boundary

Let's simplify the equation so that it becomes easier to analyze !

- $J(\theta) = \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2 = \frac{1}{2}(\theta_1^2 + \theta_2^2) = \frac{1}{2}(\sqrt{\theta_1^2 + \theta_2^2})^2 = \frac{1}{2}\|\theta\|^2$

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

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

- $\theta_0 = 0$
- $n = 2$

Let's compute $\boldsymbol{\theta}^T\boldsymbol{x}^{(i)} = ?$

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

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

SVM Decision Boundary

Now, its objective function becomes:

$$\min_{\theta} \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2 \Longleftrightarrow \min_{\theta} \Sigma_{j=1}^{n}\frac{1}{2}\|\boldsymbol{\theta}\|^2 \quad \text{s.t.} \quad \begin{aligned} &- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \geq 1 \quad \text{if } y^{(i)} = 1 \\ &- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \leq -1 \quad \text{if } y^{(i)} = 0 \end{aligned}$$

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

Simplification: $\theta_0 = 0$

$\because \boldsymbol{p}^{(1)}$ is very small

$\therefore \|\boldsymbol{\theta}\|$ must be very large

(why is this reasonable?)

$\because \boldsymbol{p}^{(2)}$ is very small

$\therefore \|\boldsymbol{\theta}\|$ must be very large

(Again, why is this reasonable?)

SVM Decision Boundary

Now, its objective function becomes:

$$\min_{\theta} \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2 \Longleftrightarrow \min_{\theta} \Sigma_{j=1}^{n}\frac{1}{2}\|\boldsymbol{\theta}\|^2 \quad \text{s.t.} \quad \begin{aligned} &- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \geq 1 \quad \text{if} \quad y^{(i)} = 1 \\ &- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \leq -1 \quad \text{if} \quad y^{(i)} = 0 \end{aligned}$$

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

Simplification: $\theta_0 = 0$

This explains how SVM gives rise to the large margin classification !

SVM Decision Boundary

Now, its objective function becomes:

$$\min_{\theta} \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2 \Longleftrightarrow \min_{\theta} \Sigma_{j=1}^{n}\frac{1}{2}\|\boldsymbol{\theta}\|^2 \quad \text{s.t.} \quad \begin{aligned} &- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \geq 1 \quad \text{if } y^{(i)} = 1 \\ &- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \leq -1 \quad \text{if } y^{(i)} = 0 \end{aligned}$$

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

Simplification: $\theta_0 = 0$

This explains how SVM gives rise to the large margin classification !

Question

- The SVM optimization problem we used is:

$$\min_{\theta} \frac{1}{2}\Sigma_{j=1}^{n}\theta_j^2 \text{ s.t. } - \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \geq 1 \quad \text{if } y^{(i)} = 1$$
$$- \boldsymbol{p}^{(i)} \cdot \|\boldsymbol{\theta}\| \leq -1 \quad \text{if } y^{(i)} = 0$$

where $p^{(i)}$ is the (signed) projection of $x^{(i)}$ onto θ. Consider the training set on the right, at the optimal value of θ, what is $\|\theta\|$?

(i) 1 / 4

(ii) 1 / 2

(iii) 1

(iv) 2

Kernels (Part 1)

Intuition

Non-linear Decision Boundary

Predict $y = 1$ if

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

i.e.

$$h_\theta(x) = \begin{cases} 1, & \textbf{if } \theta_0 + \theta_1 x_1 + \ldots \geq 0 \\ 0, & \textbf{otherwise} \end{cases}$$

Intuition

Non-linear Decision Boundary

Predict $y = 1$ if

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

i.e.

$$h_\theta(x) = \begin{cases} 1, & \textbf{if } \theta_0 + \theta_1 x_1 + \ldots \geq 0 \\ 0, & \textbf{otherwise} \end{cases}$$

New notation: $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \ldots$ where f_j denotes a feature

e.g. $f_1 = x_1,\ f_2 = x_2,\ f_3 = x_1 x_2,\ f_4 = x_1^2,\ f_5 = x_2^2, \ldots$

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

Kernels (Intuition)

Here is a new idea of defining new features $f_1, f_2, f_3, \ldots$

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

Kernels (Intuition)

Here is a new idea of defining new features $f_1, f_2, f_3, \ldots$

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

Given an example x, $f_1 :=$ **similarity**$(x, l^{(1)}) = \exp(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2})$

$f_2 :=$ **similarity**$(x, l^{(2)}) = \exp(-\frac{\|x - l^{(2)}\|^2}{2\sigma^2})$

$\vdots$

kernels
(Gaussian kernels)

Kernels (Intuition)

Here is a new idea of defining new features $f_1, f_2, f_3, \ldots$

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

Given an example x, $f_1 := \textbf{similarity}(x, l^{(1)}) = \exp(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2})$

$k(x, l^{(i)})$

$f_2 := \textbf{similarity}(x, l^{(2)}) = \exp(-\frac{\|x - l^{(2)}\|^2}{2\sigma^2})$

$\vdots$

kernels
(Gaussian kernels)

Kernels and Similarity

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

$$\text{where } \|x - l\|^2 = \Sigma_{j=1}^{n}(x_j - l_j)^2$$

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

If x is far from $l^{(1)}$, then

Kernels and Similarity

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

$$= \exp\left(-\frac{\Sigma_{j=1}^{n}(x_i - l_j^{(1)})^2}{2\sigma^2} \right)$$

If $x \approx l^{(1)}$, then $f_1 \approx \exp\left(-\frac{0^2}{2\sigma^2} \right) \approx 1$

If x is far from $l^{(1)}$, then $f_1 \approx \exp\left(-\frac{\textbf{(large number)}^2}{2\sigma^2} \right) \approx 0$

Noted that each landmark defines a new feature.

Kernels and Similarity

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

$x = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$

$\sigma^2 = 1$

(Contour plot)

Next, let's see the effect of σ^2 (which is a parameter of Gaussian kernel)

Kernels and Similarity

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

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

Question

- Consider a 1-D example with one feature x_1. Suppose $l^{(1)} = 5$. Above is a plot of $f_1 = \exp\left(-\frac{\|x_1 - l^{(1)}\|^2}{2\sigma^2}\right)$ when $\sigma^2 = 1$. Suppose we now change $\sigma^2 = 4$. Which of the following is a plot of f_1 with the new value of σ^2?

Kernels and Similarity

Given these features, let's see what the hypothesis function can learn

Hypothesis function:

Predict '1' when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Given an example x_1,

we'll compute: f_1, f_2, f_3

Let's say we predict :

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

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

i.e. $\theta_0 + \theta_1 \cdot 1 + \theta_2 \cdot 0 + \theta_3 \cdot 0 = -0.5 + 1 = 0.5 \geq 0$

$\therefore$ **Predict $y = 1$ for x_1**

Kernels and Similarity

Given these features, let's see what the hypothesis function can learn

Hypothesis function:

Predict '1' when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Given an example x_2,

we'll compute: f_1, f_2, f_3

Let's say we predict :

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

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

i.e. $\theta_0 + \theta_1 \cdot 1 + \theta_2 \cdot 0 + \theta_3 \cdot 0 = -0.5 < 0$

$\therefore$ **Predict $y = 0$ for x_2**

Kernels and Similarity

Given these features, let's see what the hypothesis function can learn

Hypothesis function:

Predict '1' when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Given an example x_2,

we'll compute: f_1, f_2, f_3

Let's say we predict :

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

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

i.e. $\theta_0 + \theta_1 \cdot 1 + \theta_2 \cdot 0 + \theta_3 \cdot 0 = -0.5 < 0$

$\therefore$ **Predict $y = 0$ for x_2**

Kernels and Similarity

Given these features, let's see what the hypothesis function can learn

Hypothesis function:

Predict '1' when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Given an example x_2,

we'll compute: f_1, f_2, f_3

Let's say we predict :

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

This gives an intuition of how definition of landmarks and kernels function can learn pretty complex non-linear decision boundary !

Kernels and Similarity

Given these features, let's see what the hypothesis function can learn

Hypothesis function:

Predict '1' when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Given an example x_2,

we'll compute: f_1, f_2, f_3

Let's say we predict :

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

Question: How to get these landmarks?, How to choose these landmarks?, what else are other similarity functions?, *etc.*

Kernels (Part 2)

Choosing the Landmarks

For every x, if x is in the dataset, we mark x as a landmark.
Each color on the right is not significant for now.

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

SVM with Kernels

Given (training / cross validation / testing) example x :

$$\left.\begin{array}{r} f_1 = \textbf{similarity}(x, l^{(1)}) \\ f_2 = \textbf{similarity}(x, l^{(2)}) \\ \vdots \end{array}\right\} \quad \boldsymbol{f} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} \quad \text{where } f_0 = 1 \text{ (an interceptor)}$$

SVM with Kernels

Given (training / cross validation / testing) example x :

$$\left.\begin{aligned} f_1 &= \textbf{similarity}(x, l^{(1)}) \\ f_2 &= \textbf{similarity}(x, l^{(2)}) \\ &\vdots \end{aligned}\right\} \quad \boldsymbol{f} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} \quad \text{where } f_0 = 1 \text{ (an interceptor)}$$

Example: For a training example $(x^{(i)}, y^{(i)})$, we can create a vector

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

$$f_i^{(i)} = \textbf{similarity}(x^{(i)}, \overset{x^{(i)}}{l^{(i)}})$$

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

SVM with Kernels

Given (training / cross validation / testing) example x :

$$\left.\begin{aligned} f_1 &= \textbf{similarity}(x, l^{(1)}) \\ f_2 &= \textbf{similarity}(x, l^{(2)}) \\ &\vdots \end{aligned}\right\} \quad \boldsymbol{f} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} \quad \text{where } f_0 = 1 \text{ (an interceptor)}$$

Example: For a training example $(x^{(i)}, y^{(i)})$, we can create a vector

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

Instead of using $x^{(i)} \in \mathbb{R}^{n+1}$, we'll represent each $x^{(i)}$ with feature vector

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

SVM with Kernels

Suppose that we have already learnt parameters $\boldsymbol{\theta}$ ($\boldsymbol{\theta} \in \mathbb{R}^{m+1}$)

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

Then, we predict '$y = 1$' if $\boldsymbol{\theta}^T \boldsymbol{f} \geq 0 \iff \theta_0 f_0 + \theta_1 f_1 + \ldots + \theta_m f_m \geq 0$

How do we get the parameters $\boldsymbol{\theta}$?

SVM with Kernels

Suppose that we have already learnt parameters $\boldsymbol{\theta}$ ($\boldsymbol{\theta} \in \mathbb{R}^{m+1}$)

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

Then, we predict '$y = 1$' if $\boldsymbol{\theta}^T \boldsymbol{f} \geq 0 \iff \theta_0 f_0 + \theta_1 f_1 + \ldots + \theta_m f_m \geq 0$

Objective function:

$$\min_{\boldsymbol{\theta}} C \Sigma_{i=1}^{m} y^{(i)} \mathbf{cost}_1(\boldsymbol{\theta}^T \boldsymbol{f}^{(i)}) + (1 - \boldsymbol{y}^{(i)}) \mathbf{cost}_0(\boldsymbol{\theta}^T \boldsymbol{f}^{(i)}) + \frac{1}{2} \Sigma_{j=1}^{n} \theta_j^2$$

Noted that, for this optimization problem, we have $n = m$

SVM with Kernels

Suppose that we have already learnt parameters $\boldsymbol{\theta}$ ($\boldsymbol{\theta} \in \mathbb{R}^{m+1}$)

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

Then, we predict '$y = 1$' if $\boldsymbol{\theta}^T \boldsymbol{f} \geq 0 \iff \theta_0 f_0 + \theta_1 f_1 + \ldots + \theta_m f_m \geq 0$

Objective function:

$$\min_{\boldsymbol{\theta}} C \Sigma_{i=1}^{m} y^{(i)} \mathbf{cost}_1(\boldsymbol{\theta}^T \boldsymbol{f}^{(i)}) + (1 - y^{(i)}) \mathbf{cost}_0(\boldsymbol{\theta}^T \boldsymbol{f}^{(i)}) + \boxed{\frac{1}{2} \Sigma_{j=1}^{n} \theta_j^2}$$

$$\because \frac{1}{2} \Sigma_{j=1}^{n} \theta_j^2 = \frac{1}{2}(\theta_1^2 + \theta_2^2) = \frac{1}{2}(\sqrt{\theta_1^2 + \theta_2^2})^2 = \frac{1}{2}\|\boldsymbol{\theta}\|^2$$

$$\because \Sigma_j \theta_j^2 = \boldsymbol{\theta}^T \boldsymbol{\theta}$$

introduced for rescaling !

In some implementations, they compute $\boldsymbol{\theta}^T \boldsymbol{M} \boldsymbol{\theta}$ for computational efficiency.

Choosing SVM Parameters

C (*e.g.* $\frac{1}{\lambda}$) Large C : Lower bias, high variance. ($\approx$ small λ) prone to overfitting
Small C : Higher bias, low variance. ($\approx$ large λ) prone to underfitting

---

σ^2 Large σ^2 : Features f_i vary smoothly. prone to underfitting $\sigma^2 = 3$
Higher bias, lower variance.

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

prone to overfitting $\sigma^2 = 0.5$

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

Question

- Suppose you train an SVM and find it overfits your training data. Which of these would be a reasonable next step? Circle all that apply.

(i) Increase C

(ii) Decrease C

(iii) Increase σ^2

(iv) Decrease σ^2

Using an SVM

Practical Use

1. Use SVM software package (*e.g.* scikit-learn, libsvm, …) to solve for parameters
2. Need to specify:
   i. Choice of parameter C
   ii. Choice of kernel (*i.e.* similarity function) *e.g.*
      - No kernel (*aka.* 'linear kernel'):
        predict $y = 1$ if $\boldsymbol{\theta}^T \boldsymbol{x} \geq 0$ $\quad (\theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n \geq 0)$

Linear kernel implies that we are using 'standard linear classifier'.

Question: when do we need to use it ?

when n is large and m is small *i.e.* $x_i \in \mathbb{R}^{n+1}$

in this case, we don't have enough data and want to avoid overfitting !

Practical Use

1. Use SVM software package (*e.g.* scikit-learn, libsvm, …) to solve for parameters
2. Need to specify:
   i. Choice of parameter C
   ii. Choice of kernel (*i.e.* similarity function) *e.g.*
   - No kernel (*aka.* 'linear kernel'): predict $y = 1$ if $\boldsymbol{\theta}^T \boldsymbol{x} \geq 0$ $\quad (\theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n \geq 0)$
   - Gaussian kernel (need to choose σ^2):

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

Question: when do we use Gaussian kernel ?
when $x \in \mathbb{R}^n$, n is small, and m is large

Practical Use

Just an implementation reminder !

Do perform feature scaling before using Gaussian kernel

Why ? $\because \|x - l\|^2 = (x_1 - l_1)^2 + (x_2 - l_2)^2 + \ldots + (x_n - l_n)^2 \quad (x \in \mathbb{R}^{n+1})$

Example (Housing domain):

$x_1 \in [0,\ 1000]$ feet^2

$x_2 \in \{1,\ 2,\ 3,\ 4,\ 5\}$

If this is the case,
these distances will be dominated by the size of the house !

Other Choices of Kernel

- Not all similarity functions **similarity**(x, l) are valid kernels
- To recognize as valid kernels, need to satisfy 'Mercer's theorem'
  - to make sure that SVM packages' optimizations run correctly, and
  - do not diverge !

- Other choices of off-the-shelf kernels are:
  - Polynomial kernel: **similarity**$(x, l) := (x^T l + \epsilon)^d$ *e.g.*

    $(x^T l)^2 \qquad (x^T l)^3 \qquad (x^T l + 1)^3 \qquad (x^T l + 5)^4$

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

Question

- Suppose you are trying to decide among a few different choices of kernel and are also choosing parameters such as C, σ^2, *etc.* How should you make the choice?

(i) Choose whatever performs best on the training data

(ii) Choose whatever performs best on the cross-validation data

(iii) Choose whatever performs best on the test data

(iv) Choose whatever gives the largest SVM margin

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 *i.e.*
  - 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 the largest $(\theta^{(i)})^T x$

Logistic Regression *vs.* SVMs

When should we use one algorithm versus another?

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

1. If n is large (relative to m), (*e.g.* $n \geq m$, $n = 10{,}000$, $10 \leq m \leq 10{,}000$) use logistic regression, or SVM without a kernel (*i.e.* 'linear kernel')

2. If n is small, m is intermediate (*e.g.* $1 \leq n \leq 10{,}000$, $10 \leq m \leq 50{,}000$) use SVM with Gaussian kernel

3. If n is small, m is large (*e.g.* $1 \leq n \leq 1{,}000$, $m > 50{,}000$) Create/add more features, then use logistic regression or SVM without a kernel

4. **Neural network (NN) likely to work well for most of these settings, but may be slower to train**

5. **SVM is a convex optimization problem.** In practice, local optima are not huge problem in NN; but, we do not need to worry about it if using SVM