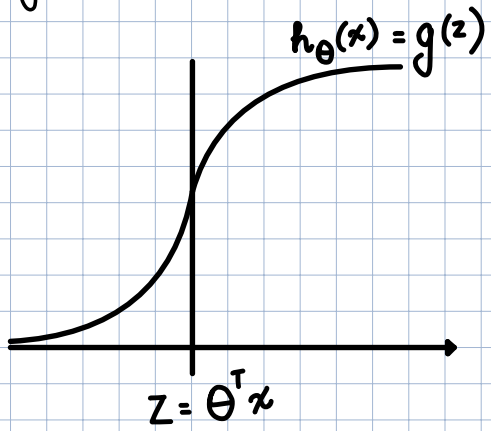
We are going to start from Logistic Regression and modify it to SVM.

From Logistic Regression, we have the hypothesis,

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

and the sigmoid graph as :-



What do we like logistic Regression to do ?

- Take an ex. of $y = 1$ (either in training / test set). Then we hope to get $h_\theta(x) \approx 1 \Rightarrow \theta^T x \gg 0$. (far right on the graph)

- Similarly for $y = 0$, we hope to get $h_\theta(x) \approx 0 \Rightarrow \theta^T x \ll 0$. (far left on the graph)

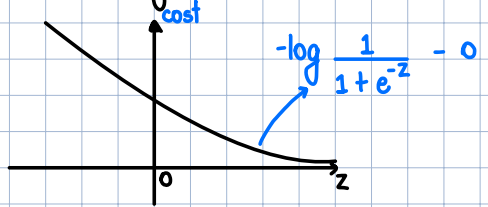
Cost fn. for logistics regression:-

$$\text{Cost}(h_\theta(x), y) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$$] of a single training ex. i.e. (x_i, y_i)

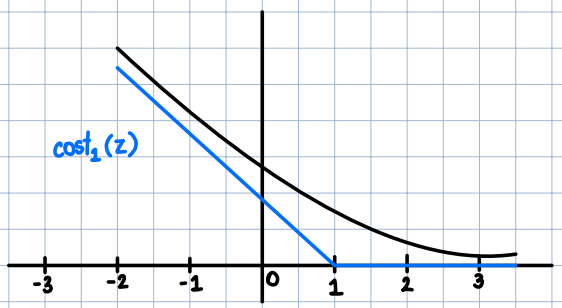
$$= -y\log\left(\frac{1}{1+e^{-\theta^T x}}\right) - (1-y)\log\left(1-\frac{1}{1+e^{-\theta^T x}}\right)$$

Case 1 from above :

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



Consider the cost fn from above (Case 1) and add a new curve to the plot. We can observe that it is quiet closely approximate the curve in black.

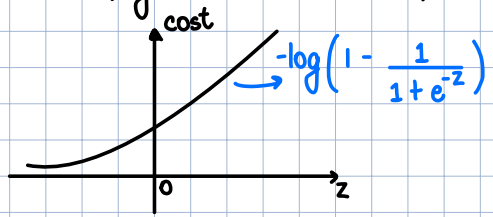


Don't worry about the slope of the blue line, it doesn't matter.

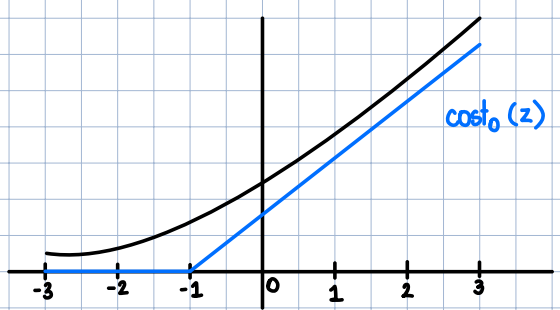
This blue line is the new cost fn. that we are going to use (when $y=1$)

Case 2 from above :

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



Adding a similar blue curve to this plot, we get,



Minimization Problem for Logistic Regression is :-

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} \left[-y^{(i)} \underbrace{\log(h_\theta(x^{(i)}))}_{Cost_1(\theta^T x^{(i)})} - (1-y^{(i)}) \underbrace{\log(1-h_\theta(x^{(i)}))}_{Cost_0(\theta^T x^{(i)})} \right] + \frac{\lambda}{2m} \sum_{i=1}^{n} \theta_i^2$$

Mimization problem for SVM becomes:-

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} \left[-y^{(i)} \, \text{Cost}_1(\theta^T x^{(i)}) - (1-y^{(i)}) \, \text{Cost}_0(\theta^T x^{(i)}) \right] + \frac{\lambda}{2m} \sum_{i=1}^{n} \theta_i^2$$

We parametrize the minimization problem slightly differently for SVM.

1.) Since $\frac{1}{m}$ is constant term, it does not contribute anything to the minimization equation and we can get rid of it.

2.) For logistic regression, we have 2 terms to the objective fn. as follows:-

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

$\text{Cost}_1(\theta^T x^{(i)})$ $\quad$ $\text{Cost}_0(\theta^T x^{(i)})$

1st term $\quad$ 2nd term

cost that comes from the training set $\quad$ regularization term

Rewritting

$$\Rightarrow \quad A \quad + \quad \lambda B$$

Here, we have a trade off between these 2 terms. What we did was by setting different values for λ (regularization parameter) we could trade-off the relative weight btw how much we want to fit the training set well (i.e minimizing A) and next is how much we care about keeping the values of the λ small.

For SVM, just for convention, we are going to use different parameters. We will use,

$$\Rightarrow \quad CA + B$$, where C is the new convention (or parameter)

Thus, we are going to minimize the above.

The way this would work is :-

- if λ is large, then B would have large weights.
- if C is small, then B would have large weights than A.

Thus, this is just a different way of parameterizing.

$$\min_{\theta} \; C \sum_{i=1}^{m} \left[-y^{(i)} \, \text{Cost}_1(\theta^T x^{(i)}) - (1 - y^{(i)}) \, \text{Cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_i^2$$

Finally, unlike logistic regression, SVM doesn't output a probability. Instead, we have the above cost fn. which we minimize to get the parameter θ.

Hypothesis :-

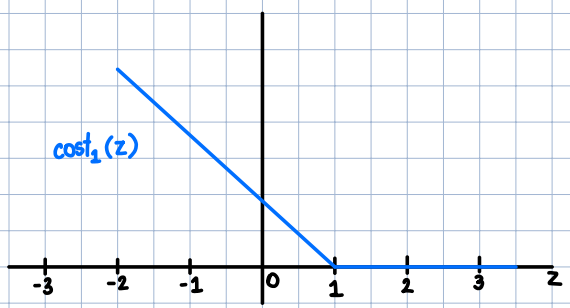
$$h_\theta(x) = \begin{cases} 1 & , \text{ if } \theta^T x \geq 0 \\ 0 & , \text{ otherwise} \end{cases}$$

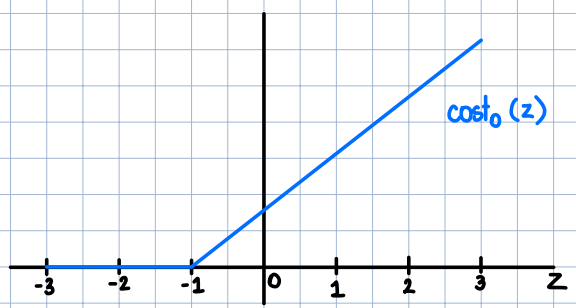
SVM as Large Margin Classifier :-

What we have so far :-

Cost fn $\Rightarrow$ $\min_{\theta} C \sum_{i=1}^{m} \left[-y^{(i)} \, \text{Cost}_1(\theta^T x^{(i)}) - (1-y^{(i)}) \, \text{Cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_i^2$

Plots of Cost_1, Cost_0





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

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

In logistic regression, we said we want $\theta^T x \geq 0$ and $\theta^T x < 0$, but in case of SVM, we want more ; that is why we choose $\theta^T x \geq 1$ and $\theta^T x \leq -1$.

This gives us more safety or builds a safety margin factor into the SVM.

Case study :-

Let's set C = large value. Then we would want to get $A = 0$.

How ? We would the above discussed two cases :-

i) whenever $y^{(i)} = 1$, we want $\theta^T x \geq 1$.

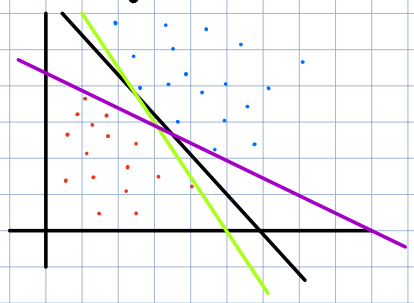
ii) whenever $y^{(i)} = 0$, we want $\theta^T x \leq -1$.

If we meet those conditions, then the minimization problem will become :- $\min \; C \cdot 0 + \frac{1}{2} \sum_{i=0}^{n} \theta_i^2$ subject to (s.t)

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

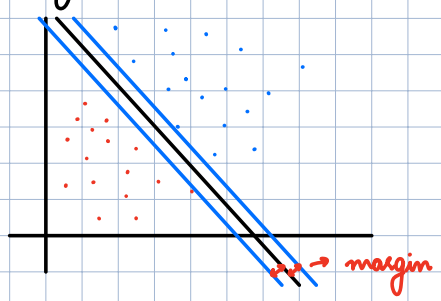
$\theta^T x \leq -1$ if $y^{(i)} = 0$

When we try to solve the above min. problem, we get an interesting decision boundary like below :-



This is dataset of +ve and -ve examples. This data is linearly separable, that means ∃ a straight line which separate the data perfectly.

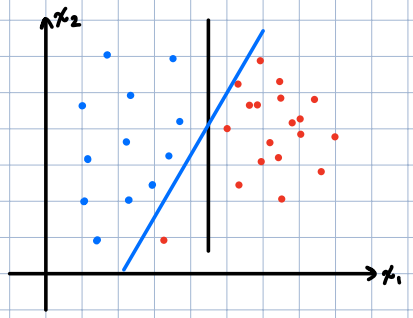
Out of the green, purple and black line, the SVM choices the black line decision boundary.



Why will SVM choose the black line? because the distance btw the black and blue lines is the max ↓ than green and purple. min distance from any of the examples (pts).

SVM is also called large Margin classifier, which is a consequence the optimization problem.

Large Margin classifier in presence of outliers



If C is not too large, then we will get the black line as our decision boundary.

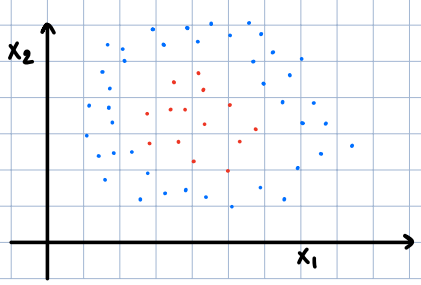
If C was too large, the black line will shift to the blue one.

So SVM will classify points correctly if C is large, otherwise in presence of outliers, the data will not be linearly separable.

Kernels - I

Here, we will start adopting SVM in order to develop complex non-linear classifier.

Non-linear Decision Boundary :-



Suppose we have a training set that looks like the one on left and we want to find a non-linear decision boundary to distinguish the +ve and -ve examples.

We can achieve the above by coming up with a set of complex polynomial features.

for example, predict $y = 1$ if $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_2^2 + \dots\dots\dots \geq 0$

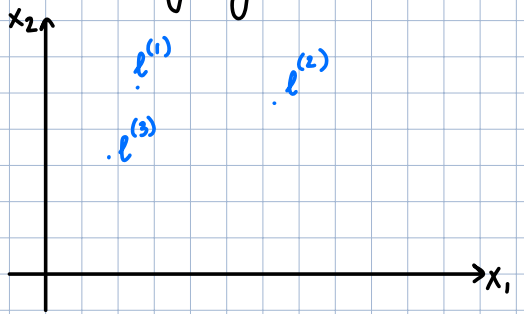
In hypothesis notation,

$$h_\theta = \begin{cases} 1 & \text{the above condition} \\ 0 & \text{otherwise} \end{cases}$$

Replace $f_1 = x_1$, $f_2 = x_2$, $f_3 = x_1 x_2$, $f_4 = x_2^2$, ...

We can see that computing these polynomials can get expensive, so is there a different / better choice of the features $f_1, f_2, f_3, \dots$?

In this section, we will only define 3 new features. But for real life problems, we are going to define much larger number.



We manually pick up few points, marked in blue $l^{(i)}$, where $1 \leq i \leq 3$.

Now Given x, let's define f_1 = some measure of similarity btw the training example x & $l^{(1)}$.

$$= \text{similarity}(x, l^{(1)})$$

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

$\|w\|$ represents length of $\vec{w}$. i.e ecuclidean dis.

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

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

And $f_3 = $ = similarity $(x, \ell^{(3)})$

$$= \exp\left(-\frac{\|x - \ell^{(3)}\|^2}{2\sigma^2}\right)$$

Similarity in the above formula means Kernal, k (mathematically).

The Kernal that we are using here is the Gaussian Kernal.

Kernals and Similarity

Given: $f_1 = k(x, \ell^{(1)}) = \exp\left(-\frac{\|x - \ell\|^2}{2\sigma^2}\right)$

Now, if $x \approx \ell^{(1)}$: (i.e if x is close to one of the landmark's, $\ell^{(i)}$)

- $\|x - \ell^{(1)}\| \approx 0$

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

Conversly, if x is far from $\ell^{(1)}$:

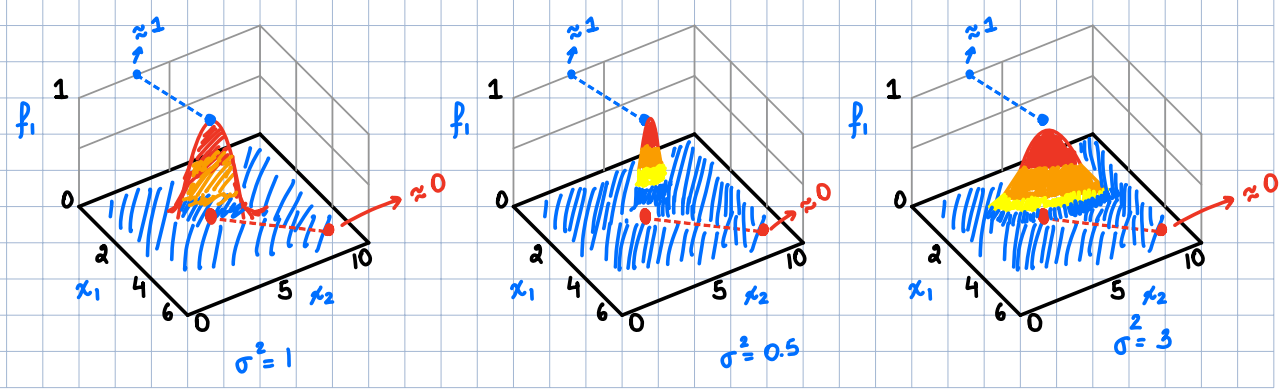
- $f_1 = \exp\left(-\frac{(\text{large num})^2}{2\sigma^2}\right) \approx 0$

Now, given x, we can compute three new features f_1, f_2, f_3 given ℓ^1, ℓ^2, ℓ^3.

Example :-

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

Plot :-



Let definition of the features be as above (Gaussian).

Given the def, let's learn the types of hypothesis we can make

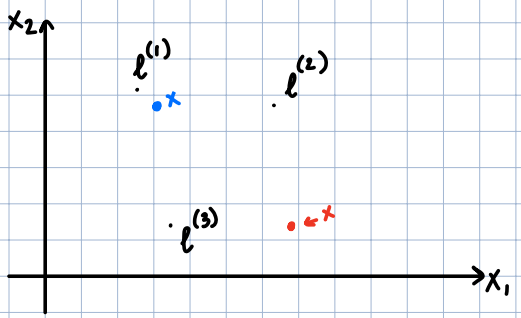
Thus, given the training example x, we are going to compute these features f_1, f_2 and f_3. Our hypothesis is going to predict 1 when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$.

"Predict 1 when

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

Let's say we already ran the algorithm and found that

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



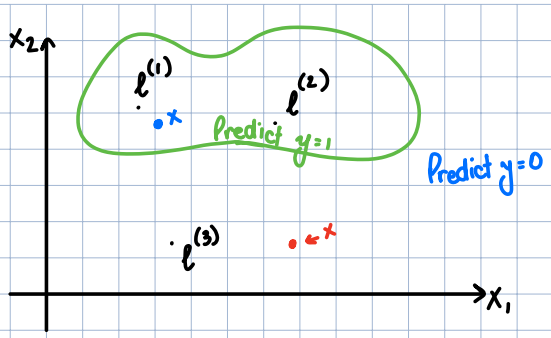
- Added a new training example in red
- $f_1 \approx 0$, $f_2 = 0$, $f_3 = 0$
- Then, $\theta_0 + \theta_1 \cdot 0 + \theta_2 \cdot 0 + \theta_3 \cdot 0$

  $\Rightarrow -0.5 = -0.5$
- At x, predict $y = 0$.

- Added a new training example
- $f_1 \approx 1$, $f_2 = 0$, $f_3 = 0$
- Then, $\theta_0 + \theta_1 \cdot 1 + \theta_2 \cdot 0 + \theta_3 \cdot 0$

  $\Rightarrow -0.5 + 1 = 0.5$
- At x, predict $y = 1$.

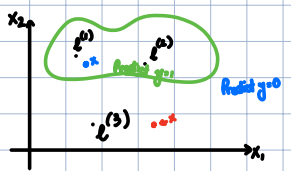
If we keep adding more points, we will see that examples near $l^{(1)}$ and $l^{(2)}$ will predict $y = 1$. So the decision boundary becomes :-



Kernals II

So far we have

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

But where to get $l^{(1)}, l^{(2)}, l^{(3)}$ from ?

Here's where .

Given: $(x^{(i)}, y^{(i)})$

Choose:- $l^{(i)} = x^{(i)}$ $\quad 1 \leq i \leq m.$

we choose the location of landmarks exactly as the location of M training examples.

Then given : example x :

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

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

.....

Let $f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$

Thus for $(x^{(i)}, y^{(i)})$,

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

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

$$\vdots$$

$$f_m^{(i)} = \text{similarity}(x^{(i)}, l^{(m)})$$

for convention $f^{(i)} = \begin{bmatrix} f_1^{(i)} \\ f_2^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix}$

SVM with Kernals

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

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

Training :-

$$\min_{\theta} C \sum_{i=1}^{m} \left[-y^{(i)} \text{Cost}_1(\theta^T f^{(i)}) - (1-y^{(i)}) \text{Cost}_0(\theta^T f^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n=m} \theta_i^2$$

Solve the above minimization problem to get the values for our parameters for SVM.

Another way to write the last term above is using linear algebra,

$$\sum_{j=1}^{m} \theta_j^2 = \theta^T \theta \quad , \text{ where } \theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_m \end{bmatrix} \quad \text{ignoring } \theta_0$$

$$= \theta^T M \theta \quad , \text{ where } M \text{ is a matrix which depends upon the Kernel we use.}$$

The reason SVM adds the additional M is that with this modification it allows the SVM to scale to much bigger training sets. Doesn't really needs to know this because SVM models does this trick at the back hand.

SVM parameters :-

Another thing worth knowing about is how do we choose the parameters.

C (similar to $1/\lambda$).
regularization parameter used for logistic regression

Large C : lower bias , high variance → prone to overfitting

Small C : Higher bias, lower variance → prone to underfitting

σ^2 : (appeared in Gaussian Kernal)

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.

Logistic regression v.s SVM

When to use them ?

Let n = # of features ($x \in R^{n+1}$)

m = # of training examples.

- If n is large (relative to m):

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

- If n is small, m is intermediate:

  Use SVM Gaussian kernal

- If n is small, m is large:

  Create / add more features, then use logistic regression

  or SVM without a kernal.