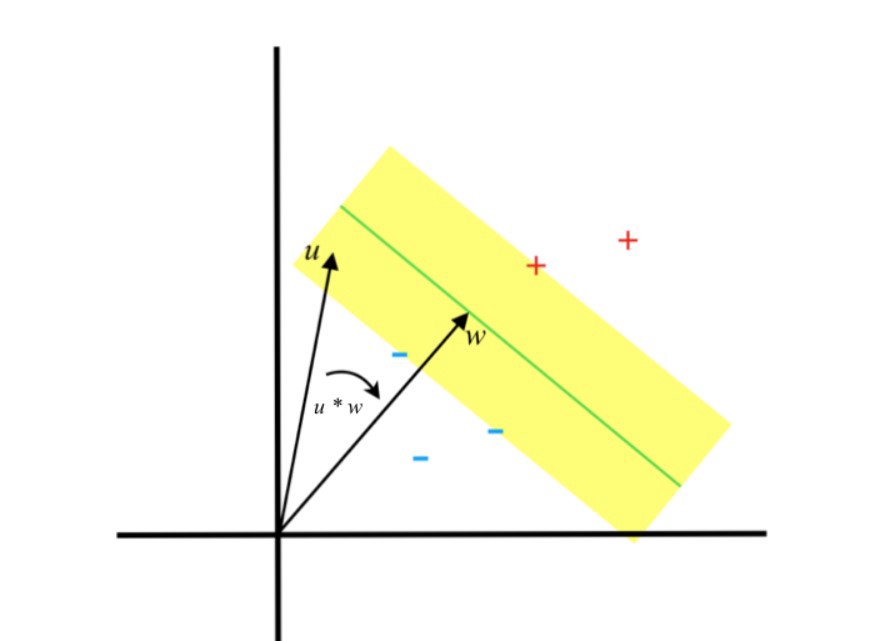
The notes given in this Medium article are very similar to the one given by Andrew Ng’s Machine Learning course.

<https://medium.com/srm-mic/demystifying-support-vector-machine-from-scratch-edaaaba4bda>

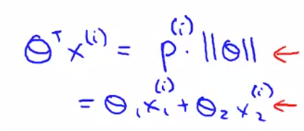


The SVM algorithm projects this new data point vector, the *u* vector (a point in this case) as seen in the above graph, onto the perpendicular vector of the decision boundary, i.e. the *w* vector.

According to our basic mathematics knowledge, when we project one vector on another, we take the **dot product** of one vector with respect to the other, which is compared with a particular constant -- basically the decision boundary.

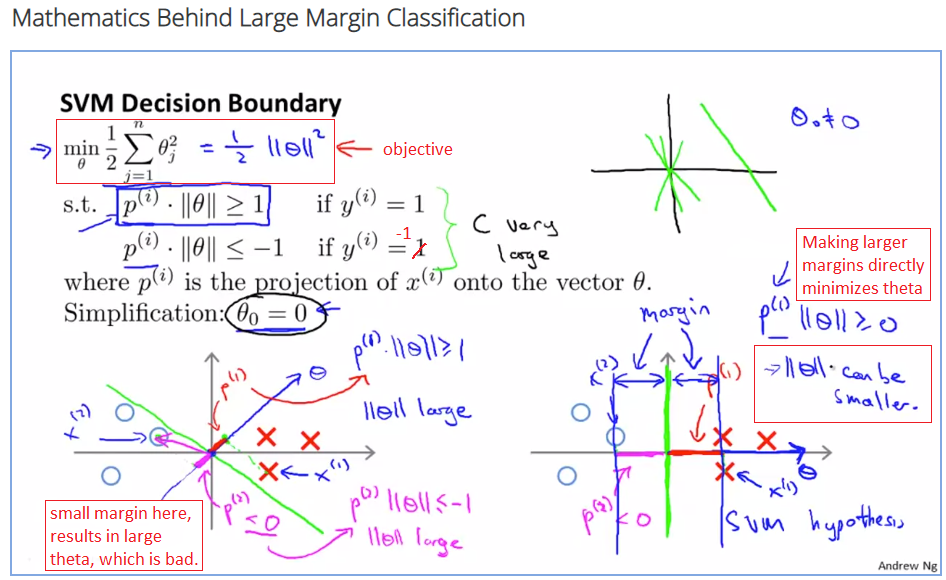
If the projection is greater than the constant, then it is of a particular category, else the other category.

When we project an example vector on the perpendicular vector, the formula is given by:



Where is the in this case, and is the projection of the example vector on the perpendicular vector . The **green line** in the figure below is the ***decision boundary*** and is the weight vector. The is like the ***margin*** distance that we want it to maximize.

Larger *p* results in lower , i.e. **maximizing the margin** and **minimizing the** (our objective!). Also refer to the **image below**.



# Linear Model

(or just in the ML course), which is the decision function, or decision boundary (a linear model) for SVM.

Also assume the positive examples are labelled as “1”, while the negative examples are labelled as “-1” instead of the usual “0”. “-1” makes it easier to work with SVM.

To be considered as **correct classifications**, these conditions must be satisfied:

if

if

Which would also result in this **important condition**:

## Hypothesis equation

This equation generates the final output from the model.

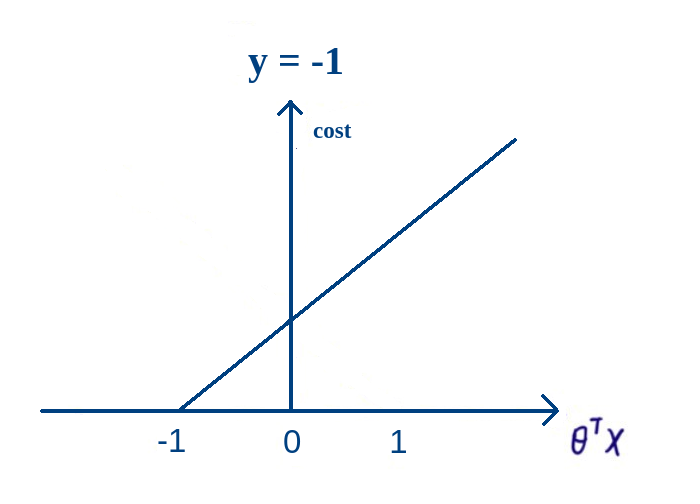
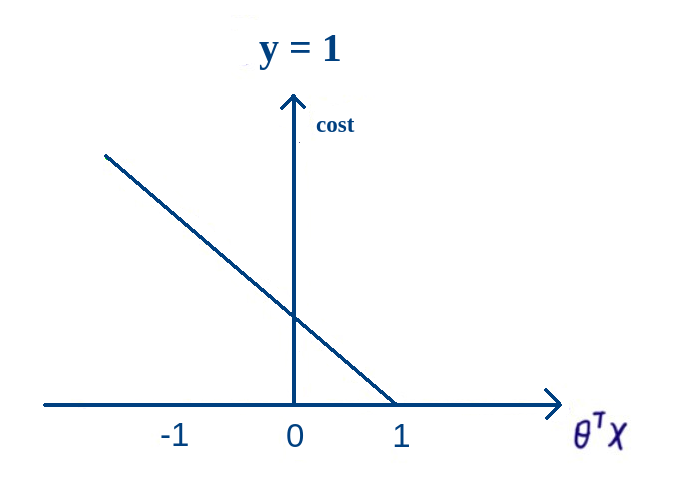
# Cost Function

The cost function is based on hinge loss.

## Hinge loss

. Provided that instead of for the negative label.

Essentially, when y = 1, we want the model to output a value ; when y = -1, we want the model to output a value , which are better than comparing against 0. This is because we penalize the model when misclassification occurs, i.e. , which would increase the loss proportionally with the value of ; whereas the loss would be 0 when the model outputs the correct classification. Refer to the formula and figures below to understand.



## Adding regularization

OR use C as the inverse of regularization.

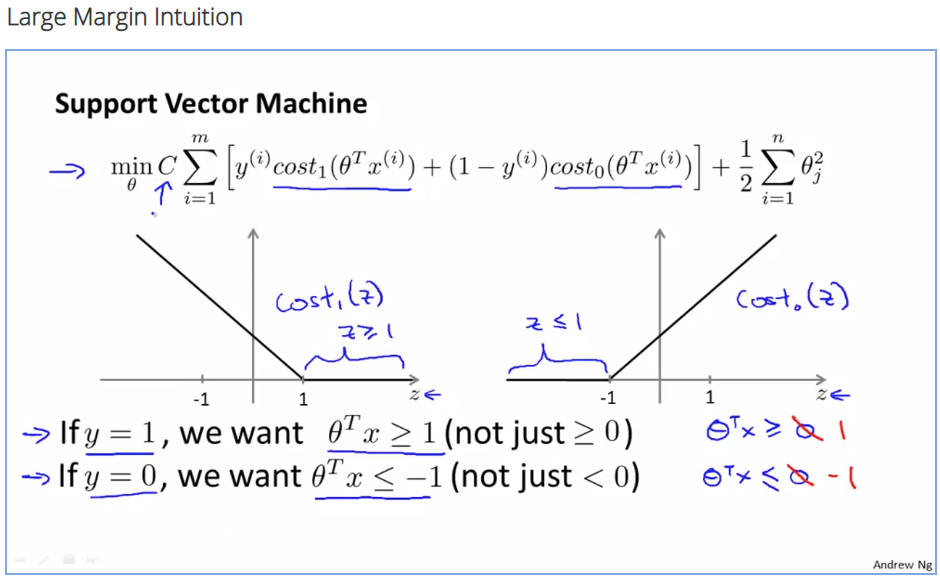
If :

else:

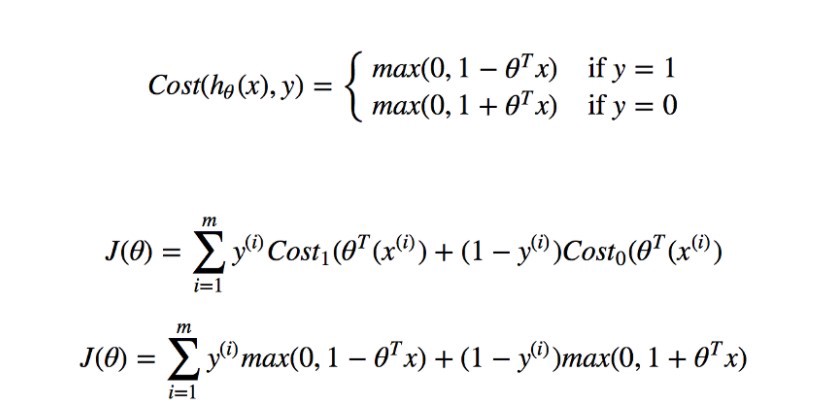
## Derivatives (gradients)

If :

else:

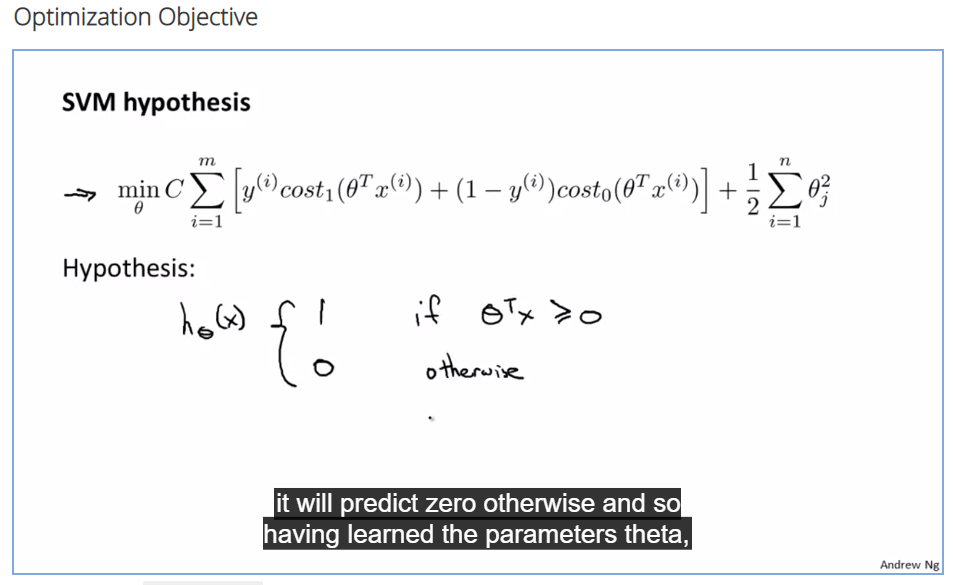


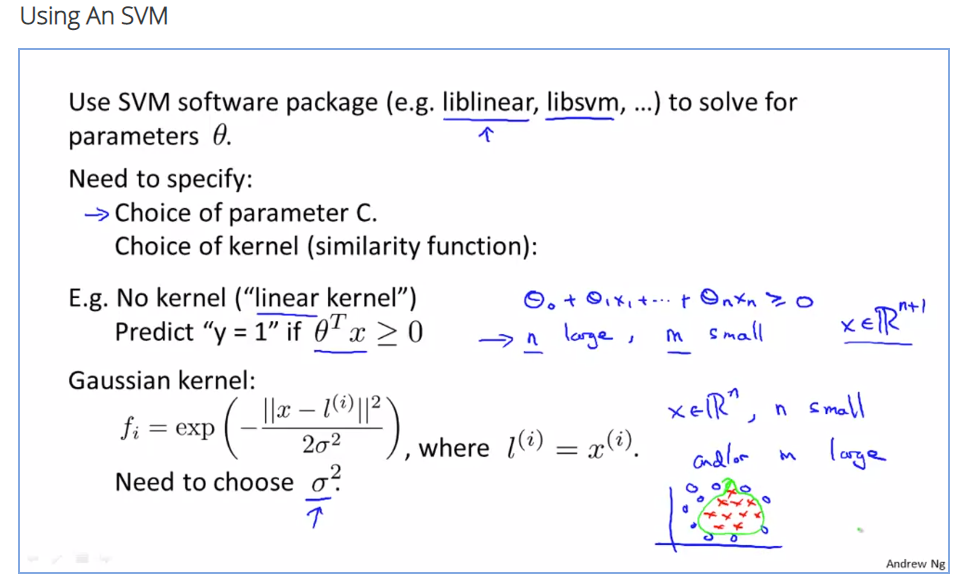
The cost1 is the hinge loss when y = 1 here, while the cost0 is the hinge loss when y = 0 here.



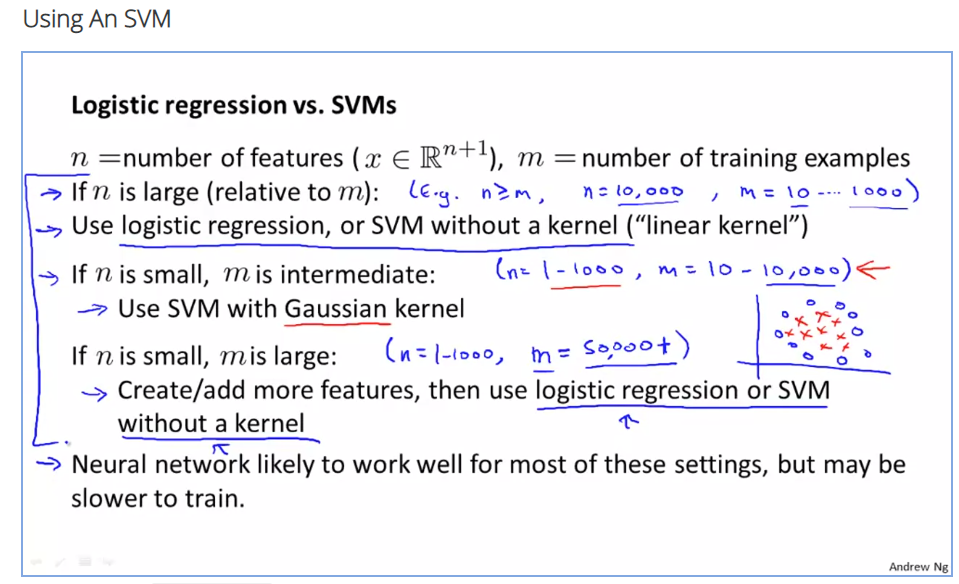
<https://medium.com/srm-mic/demystifying-support-vector-machine-from-scratch-edaaaba4bda>

These can be **simplified** by changing y to -1 instead of 0, as shown in the equations earlier.





<https://www.coursera.org/learn/machine-learning/lecture/sKQoJ/using-an-svm>



The similarity function used here is the Gaussian kernel, aka Radial Basis Function (RBF) kernel, where each *x* is replaced by the similarity function, i.e. RBF kernel function. The , and the landmarks are chosen to be equal to each *x*, .

The shape of should be (number of features, 1), while the shape of ***f*** should be (number of samples, number of features), as each has the length of (+1 if including the bias, ) as shown in the image below. So, they should be in the right shapes to get their dot product.

