Machine Learning

Andrew Ng

Summary

1 Machine Learning in a Nutshell

1.1 Supervized vs Unsupervized Learning

Supervised Learning problems are typically regressions and classification problems. These problems aim to find a way to make the most correct predictions. To do this, you have to define criteria for "correct" and the program will iteratively take an enormous number of stabs to find an equation that returns the most correct predictions.

Unsupervised Learning are things like clustering and recommender systems. These algorithms objectively group similar datapoints without any specific instruction in order to higlight patterns that naturally occur in the data.

1.2 General Form of Regression and Cost Functions

Notation

m: The number of training examples	**X**: Input variables / features
y: Output variable / target	**h**: hypothesis, mapping **X** to **y**
(x, y) : one training example	$(x^{(i)}, y^{(i)})$: the i^{th} training example

Generally speaking, for supervised learning algorithms the cost function fits the form

$$h_\Theta(x) = \Theta_0 + \Theta_1 x$$

where Θ_0 is a given and then we multiply Θ_1 by x to get the total cost value. Our supervised learning algorithm adjusts the Θ's (taking the values for x as given) until h_Θ looks like we want it to.

We want to choose our Θ's so as to most accurately predict our y's, thus minimizing the distance between the predictions and our observations.

In math, we call this the *Squared Error Function*:

$$J(\Theta_0, \Theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\Theta(x^{(i)}) - y^{(i)})^2$$

$$\min_{\Theta_0, \Theta_1} J(\Theta_0, \Theta_1)$$

1.3 Gradient Descent

Graphically, our Squared Error Function looks like a hyperplane with a huge sink in the center

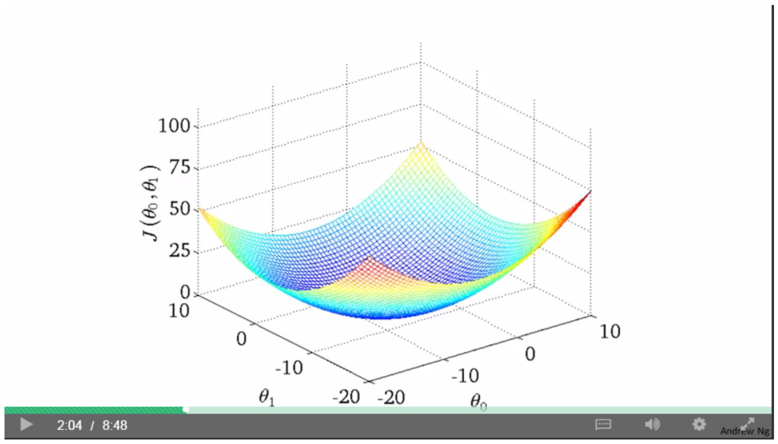

Figure 1: The error function is shaped like a hyperplane

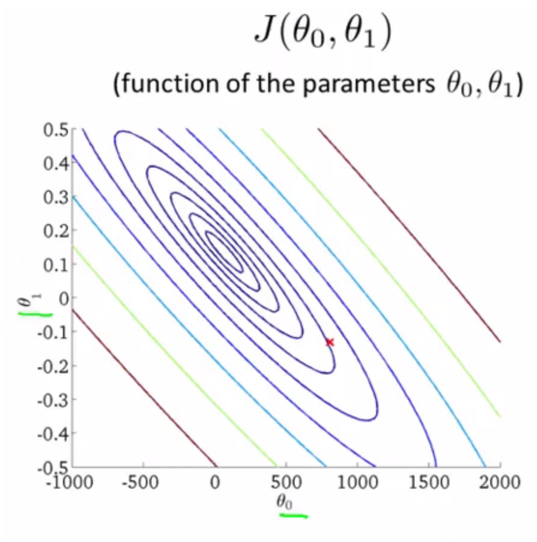

Figure 2: Cross sections of the error function

Iteratively, the supervised learning algorithm will try different values for Θ_0 and Θ_1 until it gets to the minimum (represented by the center of these ellipses).

2 Expanding on One-Variable Regression

For a given data point

```
# comment
import pandas as pd
```

Figure 3: Dummy figure

```
df = pd.DataFrame()
one = 100
string = 'This is a string'
```

List of Figures

List of Tables