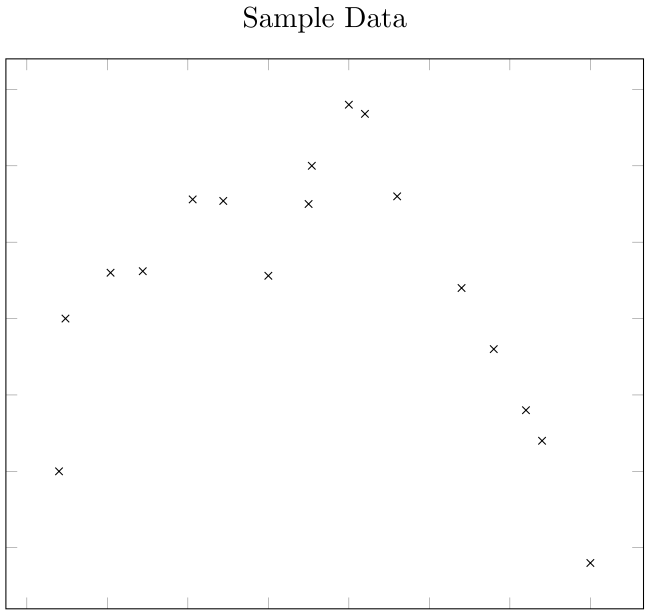
Locally Weighted Linear Regression

Sometimes a curve can be fit by linear regression, but it is very difficult to determine what features need to be used. For example, the following training examples plotted as a function of x and y:



How would we decide what features we want to use for a linear regression algorithm? Because of this difficulty, we can use something called locally weighted regression instead.
In machine learning, there is a difference between parametric and non-parametric learning algorithms. In a parametric learning algorithm, there is a fixed set of parameters that are fit to data. A non-parametric learning algorithm has an amount of data that needs to be kept which grows with the size of the data.
With locally weighted regression, we look at some point x and then take some local set of training examples in that local area of x. Mathematically, we fit theta (the parameters) to minimize the cost function

$$\sum_{i=1}^{m} w^{(i)} \left(y^{(i)} - \theta^T x^{(i)}\right)^2$$

where $w^{(i)}$ is a "weighting" function, which is commonly evaluated as

$$\exp\left(-\frac{\left(x^{(i)} - x\right)^2}{2}\right)$$

x is the location where we want to compute a local area, and we use this weighting function to determine how our fitting line should be drawn. However, we need to know what the size of this local area should be. We can define τ to be the bandwidth, or size of the local area to which we fit our line, and then add that back into our weighting function:

$$\exp\left(-\frac{\left(x^{(i)}-x\right)^2}{2\tau^2}\right)$$

Locally weighted regression is best used with a low-dimensional dataset, meaning there aren't many features, but there are a lot of training examples (meaning you don't want to think about what features you will have to use in an algorithm such as linear regression).