**Multivariate Linear Regression**

Consider that instead of having just a single feature, we have **multiple features**. Thus, the value would no longer be a single value but rather a **vector**, .

If we consider all the data points, we can store them in **transposed form**, like this:

This is called a **data matrix**.

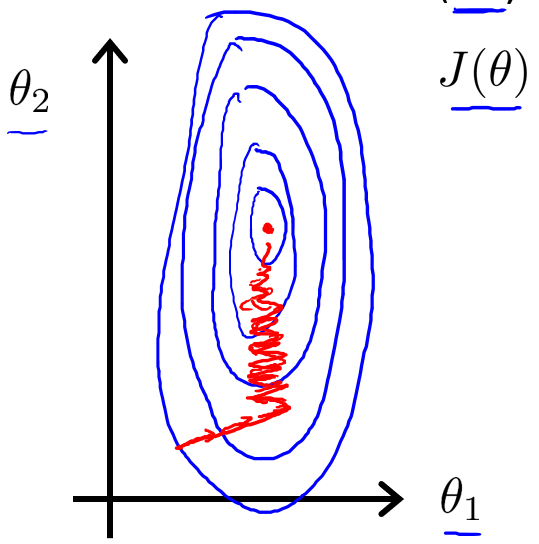
For the features in a particular data point, we will also have coefficients. Thus, the **hypothesis** will become like this:

Both the **cost function** and the **gradient descent algorithm** will remain exactly the same.

## Feature Scaling

**Feature Scaling** is a pre-processing technique that can be used to improve the performance of the gradient descent algorithm when using it with multiple features.

Suppose is the area of a house and is the number of rooms. The number of rooms can vary from while the area can vary from . Thus, the ranges of the two features are wildly different. In a scenario like this, we will have a **narrow contour plot**. A change in will cause a significant change in the cost function. The parameter for , , will pull the gradient descent algorithm away from the minima, thus slowing down the process.



To solve this issue, we can use the following formula on all values of .

For example, if the values of are from to , this changes the range to to . If this is inconvenient, it can be doubled to to .

## Mean Normalization

Another optimization technique is called **mean normalization**.

Here, is the **average** of all the values of and is either the **range** or the **standard deviation**. We can also use the **variance** instead of .