**Feature in Machine Learning :**

**ML focuses on the development of computer programs** that can access data and use it to learn for themselves. This is achieved by training the machine on large sets of diverse data. We need to design a predictive model, which can predict the future values or logical values, that would help to plan better, improve forecasting, optimize resources etc.

For designing such a predictive models, we need to identify the parameters, variables or attribute. This is called Feature Selection.

We identify the feature for following reasons :

* It helps in simplification of models by breaking it into features.
* Easier training possible with this model.
* It helps in increased generalization and reduce overfitting.
* Shorter training times.
* To avoid the [curse of dimensionality](https://en.wikipedia.org/wiki/Curse_of_dimensionality),
* Enhanced generalization by reducing [overfitting](https://en.wikipedia.org/wiki/Overfitting)[[2]](https://en.wikipedia.org/wiki/Feature_selection#cite_note-Bermingham-prolog-2) (formally, reduction of [variance](https://en.wikipedia.org/wiki/Bias-variance_tradeoff)[[1]](https://en.wikipedia.org/wiki/Feature_selection#cite_note-islr-1))

The Features are identified using the Domain Knowledge. If Feature engineering is done correctly, it increases the accuracy of the prediction by the machine.

**The process of feature engineering**

Repeat, if required

**Feature Scaling :**

**Feature scaling** is a method used to standardize/ normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.

Methods for feature Scaling :

1. **Rescaling :** The simplest method is rescaling the range of features to scale the range in [0, 1] or [−1, 1]. Selecting the target range depends on the nature of the data. The general formula is given as:

**x’ = x – min(x)**

**Max(x) – Min(x)**

1. **Mean Normalization :**

**X’ = x-mean(x)**

**( Max(x) – Min(x) )**

1. **Standardization :**

This method is widely used for normalization in many machine learning algorithms (e.g., [support vector machines](https://en.wikipedia.org/wiki/Support_vector_machine), [logistic regression](https://en.wikipedia.org/wiki/Logistic_regression), and [neural networks](https://en.wikipedia.org/wiki/Neural_network))[[2]](https://en.wikipedia.org/wiki/Feature_scaling#cite_note-:0-2)[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]. The general method of calculation is to determine the distribution [mean](https://en.wikipedia.org/wiki/Mean) and [standard deviation](https://en.wikipedia.org/wiki/Standard_deviation) for each feature. Next we subtract the mean from each feature. Then we divide the values (mean is already subtracted) of each feature by its standard deviation.

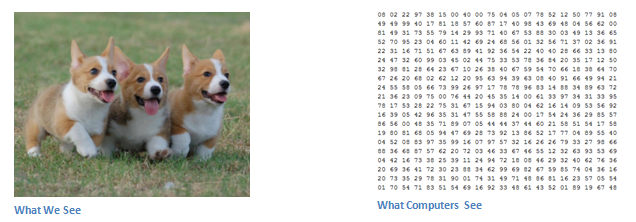
**X’ = x – mean(x)**

**Standard deviation(x)**

**Convolution :**

Convolution is a revolutionary technique in identifying the images. When a human sees at a image, he identify the images in micro-seconds. He subconsciously notice the major feature of the image, like borders, edge, shapes and characterize the component of image and give them labels. But for machine these features are not visible. For machine the image is just a number.

For Example, look at the following images :

****

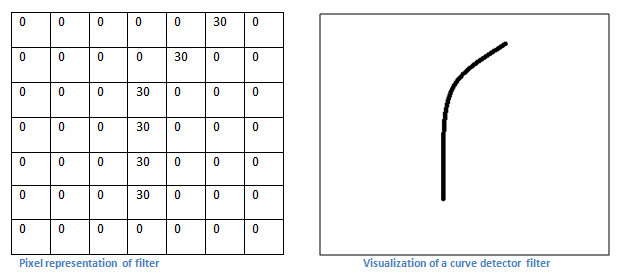
So, based on this number, computer will try to understand the image.

Convolution is the first process in this.

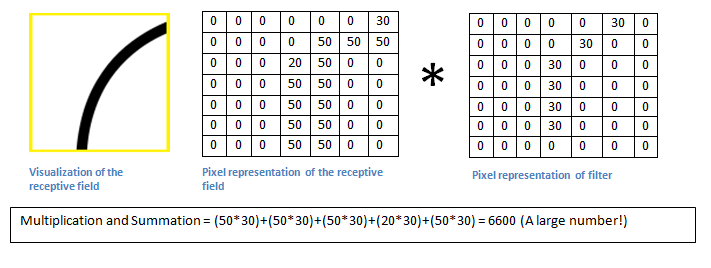
Suppose, we have image of 100\*100\*3 pixels. We will have a filter(filter is a feature identifier, like curve identifier) of 10\*10\*3 pixel that will scan the image from top left corner. The depth of filter is same as depth of image(i.e 3).

As the filter is sliding, or **convolving**, around the input image, it is multiplying the values in the filter with the original pixel values of the image (aka computing **element wise multiplications**). These multiplications are all summed up.

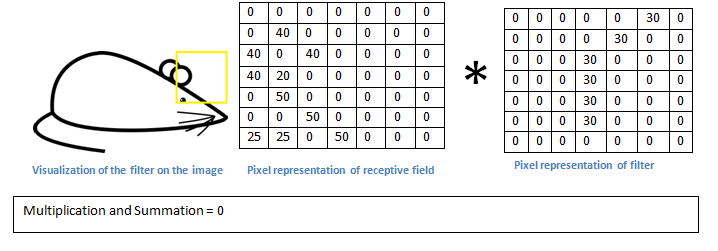
However, let’s talk about what this convolution is actually doing from a high level. Each of these filters can be thought of as **feature identifiers**. When We say features, We are talking about things like straight edges, simple colors, and curves. Think about the simplest characteristics that all images have in common with each other.

****

Remember, what we have to do is multiply the values in the filter with the original pixel values of the image.

****

Basically, in the input image, if there is a shape that generally resembles the curve that this filter is representing, then all of the multiplications summed together will result in a large value! Now let’s see what happens when we move our filter.

****

The value is much lower! This is because there wasn’t anything in the image section that responded to the curve detector filter.

So, Our image processing pipeline is a series of steps: convolution, max-pooling, and finally a fully-connected network.

When solving problems in the real world, these steps can be combined and stacked as many times as you want! You can have two, three or even ten convolution layers. You can throw in max pooling wherever you want to reduce the size of your data.

**PROGRAMMING QUESTIONS :**

1. **Matrix Multiplication :**

****

1. **printFibonacci :**

****

1. **Reverse List :**

****

**GITHUB ACCOUNT :**

**The Assignment is also uploaded to my github account at following URL.**

[**https://github.com/RishiSrivastav/Machine-Learning**](https://github.com/RishiSrivastav/Machine-Learning)

