**Gradient Descent Simple Explanation**

An intuitive way to think of Gradient Descent is to imagine the path of a river originating from top of a mountain.  
The goal of gradient descent is exactly what the river strives to achieve - namely, reach the bottom most point (at the foothill) climbing down from the mountain.  
  
Now, if the terrain of the mountain is shaped in such a way that the river doesn't have to stop anywhere completely before arriving at its final destination (which is the lowest point at the foothill, then this is the ideal case we desire. In Machine Learning, this amounts to saying, we have found the global minimum (or optimum) of the solution starting from the initial point (top of the hill).  
  
However, it could be that the nature of terrain forces several pits in the path of the river, which could force the river to get trapped and stagnate. In Machine Learning terms, such pits are termed as local minima solutions, which is not desirable. There are a bunch of ways to get out of this (which I am not discussing).  
  
Gradient Descent therefore is prone to be stuck in local minimum, depending on the nature of the terrain (or function in ML terms). But, when you have a special kind of mountain terrain (which is shaped like a bowl, in ML terms this is called a Convex Function), the algorithm is always guaranteed to find the optimum. You can visualize this picturing a river again. These kind of special terrains (a.k.a convex functions) are always a blessing for optimization in ML.  
  
Also, depending on where at the top of the mountain your initial start from (ie. initial values of the function), you might end up following a different path. Similarly, depending on the speed at the river climbs down (i.e.. the learning rate or step size for the gradient descent algorithm), you might arrive at the final destination in a different manner. Both criteria can affect whether you fall into a pit (local minima) or are able to avoid it.