Cost Function - Tells the network its performance on a given example

- We can gain an idea of the performance of the network by calculating the average cost over a testing set
- Example cost function - **sum of squares $\Sigma(out\_ac_v - target_v)^2$**
    - The target can be a **one hot vector**

Learning - we are just solving equations by trying to find the minima, the places on the loss function where the gradient is 0 in the valley of the error

- You can solve this for a simple problem by solving the derivative by equation it to 0
    - This is infeasible for complicated problems, i.e. machine learning
- *Instead, we look at the gradient at our weight and see if the gradient is positive or negative, then use its absolute value to determine how far should we move towards a minimum*
        - The steps get smaller as the gradient gets smaller
    - We can do this repeatedly with our dataset to approach some minimum
    - There is no guarantee that the local minimum you achieve is the lowest minimum possible of the cost f

Gradient - the gradient of a multivariable function is a vector in the direction of steepest increase

- So we can go towards minimum by going small step in negative gradient

Smooth gradients - make it easier to find minima, which is why nodes have continuous values (vs binary)

Stochastic Gradient Descent

- In practice it takes a while to add up and average the gradients of every single training data every single epoch
- Instead, we shuffle our data and group them into mini-batches
- We then compute our gradient descent on the mini-batches
- Drunken man steps that are really quick vs careful man taking slow precise steps