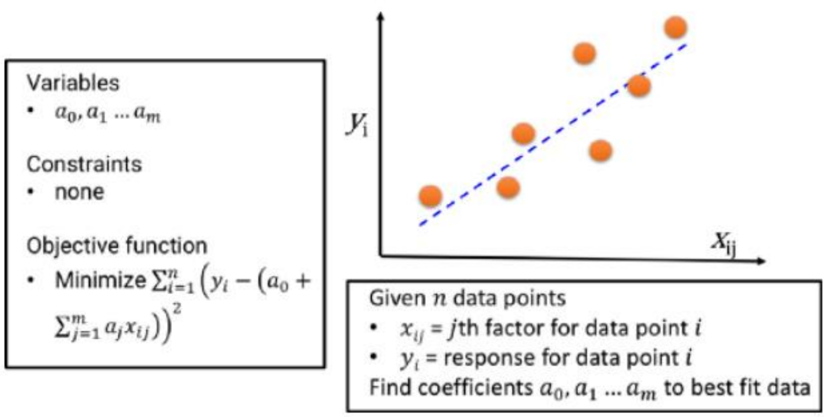
1. What are the basic elements of optimization? Give some examples.

Basic elements of optimization:

There are three basic elements of any optimization problem -

- Variables: These are the free parameters which the algorithm can tune
- Constraints: These are the boundaries within which the parameters (or some combination thereof) must fall
- Objective function: This is the set of goal towards which the algorithm drives the solution. For machine learning, often this amount to minimizing some error measure or maximizing some utility function.



Simple linear regression

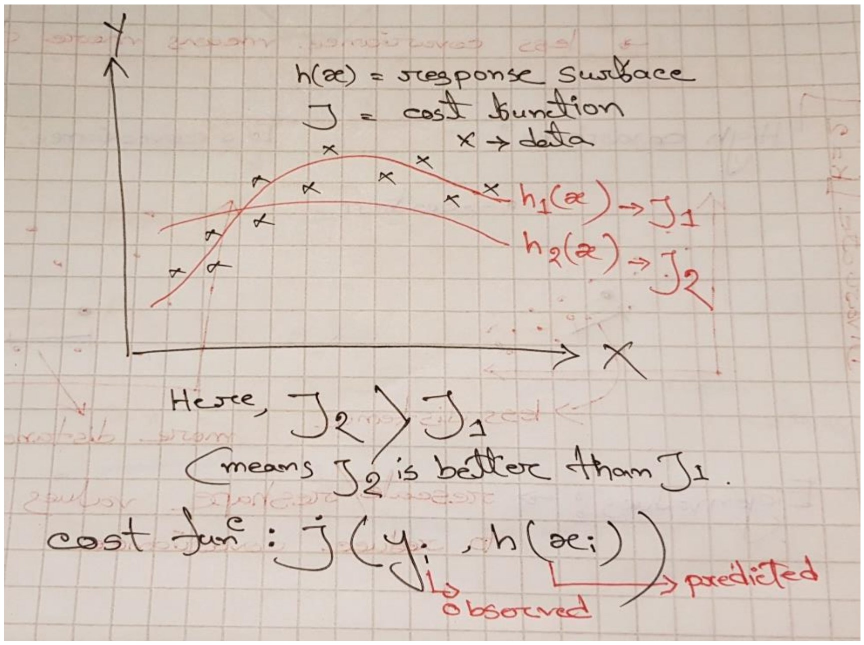
2. Why is optimization important in machine learning? Give some examples.

Optimization is the core of the machine learning, also from the business, social and economic perspective. All the engineering product/solution is an outcome of an optimized problem. Basic science, business organizations, and engineering enterprises have been using optimization techniques and methods since long.

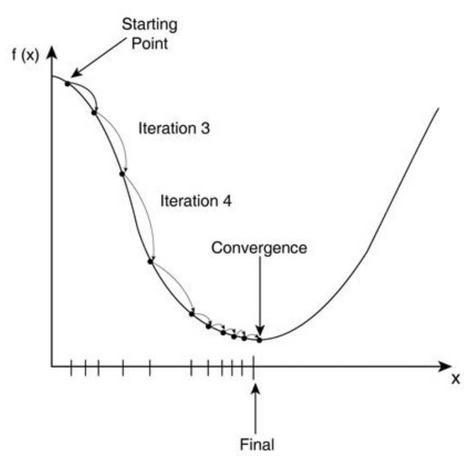
In this sense, almost every engineering product is a compact physical (or virtual) form of a solution of an optimization problem. Engineers are specifically trained to work under resource constraint and to produce '*good enough*' solutions from incomplete or noisy data or input. Essentially they solve optimization problems everyday with computers, semiconductor ICs, furnaces, or combustion engines.

3. *What is a loss function /cost function?*

Cost function is used to solve the supervised learning problem. It describes how well the current response surface fits with the available data. Smaller cost function is our main target in Machine learning. Because smaller cost functions means better fit with the data.



We can minimize the cost functions using gradient descent. Gradient descent is an efficient optimization algorithm that attempts to find a local or global minima of a function.



The loss function (or error) is for a single training example, while the cost function is over the entire training set (or mini-batch for mini-batch gradient descent).

4. What is a gradient operator?
   **Trick question :What are its possible eigenfunctions and eigenvalues

Gradient operator: ∇ called nabla

also called Hamilton operator.

The gradient is a vector operator for any n-dimentional scalar function

For example, in image processing if we use 2D, then-

$$\nabla = \begin{bmatrix} \partial/\partial x \\ \partial/\partial y \end{bmatrix}$$

if we have a function, $f(x,y)$ and if we apply gradient, it produces a vector function - g.

$$g = \nabla f(x,y) = \begin{bmatrix} \partial/\partial x \\ \partial/\partial y \end{bmatrix} = \begin{bmatrix} \partial f/\partial x \\ \partial f/\partial y \end{bmatrix}$$

Example: if $f(x,y) = x^2 + y^2$

$$\nabla f(x,y) = \langle 2x, 2y \rangle$$

Eigen value of a gradient operator:

Eigen function : $\phi(\vec{r}) = e^{a\vec{k}\cdot\vec{r}}$

Eigen value : ak

We can prove $\phi(\vec{r}) = e^{a\vec{k}\cdot\vec{r}}$ is an eigen function by applying differential operator.

$$\boxed{\nabla \phi(\vec{r}) = ak\,\phi(\vec{r})}$$

For x component,

$$\nabla_x e^{ak\cdot r} = \frac{\partial}{\partial x} \exp a(k_x x + k_y y + k_z z)$$

$$= ak_x \cdot \exp(k_x x + k_y y + k_z z)$$

$$= ak_x \cdot \boxed{e^{a\vec{k}\cdot\vec{r}}}$$

$$= ak_x \cdot \phi(\vec{r})$$

For y component,

$$\nabla_y e^{\boxed{ak\cdot r}} = \frac{\partial}{\partial y} \exp^{a\left(k_x x + k_y y + k_z z\right)}$$

$$= a \cdot \exp^{a\left(k_x x + k_y y + k_z z\right)} \cdot \frac{\partial}{\partial y}\left(k_x x + k_y y + k_z z\right)$$

$$= a \cdot \exp^{a\left(k_x x + k_y y + k_z z\right)} \cdot \underbrace{\left(0 + \frac{\partial}{\partial y}\left(k_y y\right) + 0\right)}_{k_y\left(\frac{\partial y}{\partial y}\right) = k_y}$$

$$= a \cdot k_y \cdot \exp^{a\left(k_x x + k_y y + k_z z\right)}$$

$$= a \cdot k_y \cdot \boxed{\exp^{a\vec{k}\cdot\vec{r}}} \rightarrow \phi(\vec{r})$$

$$= a k_y \cdot \phi(\vec{r})$$

For z-component: $a k_z \cdot \phi(\vec{r})$

$\therefore$ Eigen values: ak_x, ak_y, ak_z.