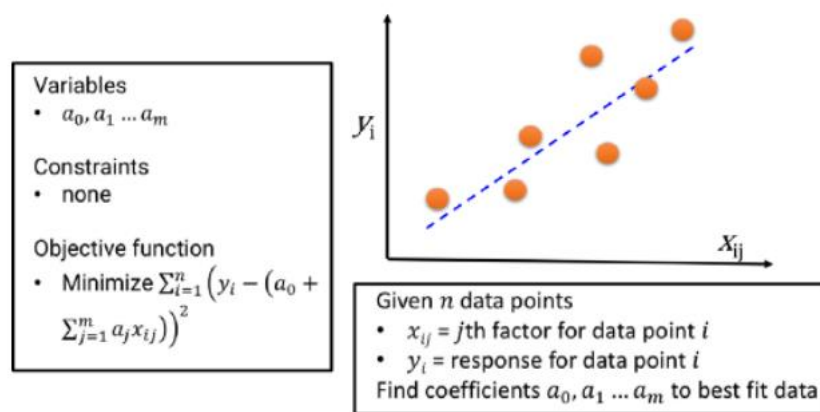


1. What are the basic elements of optimization? Give some examples.

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

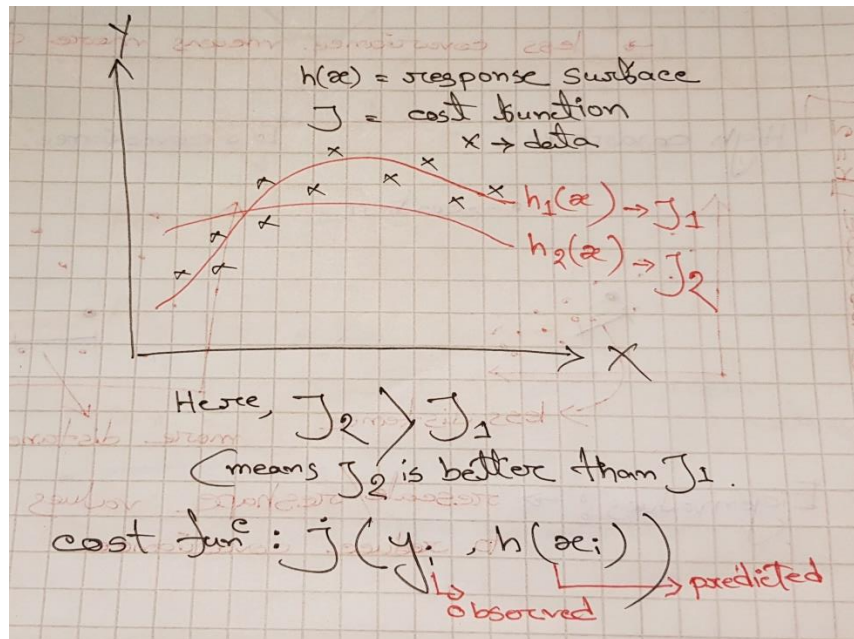
Why is optimization important in machine learning? Give some examples.

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.

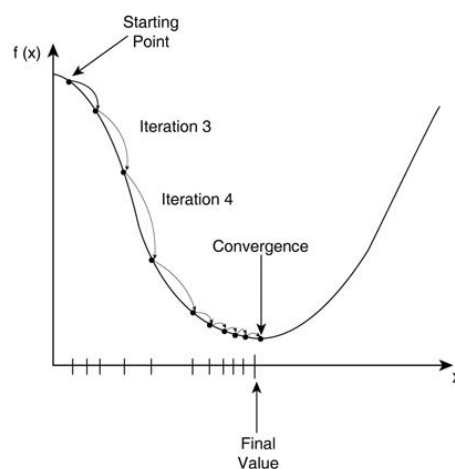
In this sense, almost every engineering product is a compact physical (or virtual) form of a solution of an optimization problem. Engineers are specifically trained to work under resource constraint and to produce 'good enough' solutions from incomplete or noisy data or input. Essentially they solve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

2. 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).

3. What is a gradient operator?

**Trick question : What are its possible eigenfunctions and eigenvalues

Gradient operator: ∇ is called
also called Hamilton operator.

The gradient is a vector operator for
any n-dimensional 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, $k(x, y)$ and if we
apply gradient, it produces a vector
function - g .

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

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

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

Eigen value of a gradient operator:

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

Eigen value: $\hbar k$

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

$$\nabla \phi(\vec{r}) = i\vec{k} \phi(\vec{r})$$

For x component,

$$\nabla_x e^{i\vec{k} \cdot \vec{r}} = \frac{\partial}{\partial x} e^{i(k_x x + k_y y + k_z z)}$$

$$= i k_x e^{i(k_x x + k_y y + k_z z)}$$

$$= i k_x \cdot e^{i\vec{k} \cdot \vec{r}}$$

$$= i k_x \phi(\vec{r})$$

For y component,

$$\nabla_y = \frac{\partial}{\partial y} e^{a k \cdot \vec{r}} = \frac{\partial}{\partial y} e^{a(k_x x + k_y y + k_z z)}$$

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

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

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

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

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

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

\therefore Eigen values: $a k_x, a k_y, a k_z$.

4. How is Hessian related to the gradient, eigenvalues, optimization and convergence?
5. What is the relation between quadratic form, Hessian, precision matrix, Mahalanobis distance and optimization?

The solution of 4 and 5 has been combined as:

The first derivative of the polynomial equation gives the gradient of the equation. And the second order derivative w.r.t each dimension (if exists) results into Hessian. These second order derivative represent curvature or the rate of change of gradient with respect to each dimension. Hessian represented into the matrix form gives hessian matrix which is symmetric in nature.

In the neighbourhood of $(x, y) = (a, b)$

$$F(x, y) \approx F(a, b) + F_x(a, b)(x-a) + F_y(a, b)(y-b) + \frac{1}{2!} [F_{xx}(a, b)(x-a)^2 + 2F_{xy}(a, b)(x-a)(y-b) + F_{yy}(a, b)(y-b)^2]$$

The first three terms are linear & used for linear approximation while the other quadratic terms adds for quadratic approximation for optimization

If X represents (x, y) & $a = (a, b)$ then

$$F(X) = f(a) + [(X-a) \nabla f(a)] + [(X-a) \cdot (H(X) \cdot (X-a))]$$

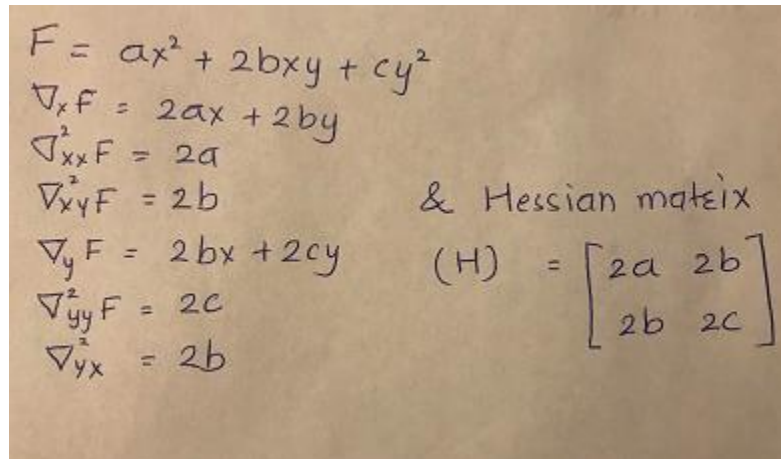
Where H is the matrix of second derivative

$$H(X) = H(x, y) = \begin{bmatrix} \nabla_{xx}^2 f(x, y) & \nabla_{xy}^2 f(x, y) \\ \nabla_{xy}^2 f(x, y) & \nabla_{yy}^2 f(x, y) \end{bmatrix}$$

also known as Hessian matrix.

Let's consider a second order polynomial equation.

$$F = ax^2 + 2bxy + cy^2$$


$$\begin{aligned} F &= ax^2 + 2bxy + cy^2 \\ \nabla_x F &= 2ax + 2by \\ \nabla_{xx}^2 F &= 2a \\ \nabla_{xy}^2 F &= 2b \\ \nabla_y F &= 2bx + 2cy \\ \nabla_{yy}^2 F &= 2c \\ \nabla_{yx}^2 F &= 2b \end{aligned} \quad \text{\& Hessian matrix}$$
$$(H) = \begin{bmatrix} 2a & 2b \\ 2b & 2c \end{bmatrix}$$

Any symmetric matrix can be decomposed into the respective eigen value and eigen vectors. Since, hessian matrix is also symmetric, so the decomposition follows as:

$$\begin{aligned} H\mathbf{V} &= \lambda\mathbf{V} \\ (H - \lambda\mathbf{I})\mathbf{V} &= \mathbf{0} \end{aligned}$$

Eigen values can be calculated by equating the determinant of term $H - \lambda\mathbf{I}$ to zero.

The eigen vector associated with the largest eigen value gives the direction of convergence in the optimization. In the contour plots of cost function, the greatest eigen values and associated vector is in the direction of the minor axis of the ellipse.

6. *What is momentum and what problem is it solving?*

When we have a very larger difference in the eigen values of hessian matrix, then it is difficult to converge the cost function. when we have a larger value of step size in the direction of larger curvature, there is chance of over shooting the minimum and if there is smaller step size in direction of smaller curvature, it takes a lot of time in converging. This is called ill conditioning. So, momentum is the parameter that increases the step size in the direction of smaller curvature and decreases the step sizes in the direction of larger curvature thereby directing the optimization which is better conditioned.