CO2774
Machine Learning
Aug 25, 2021

Announcements:-
- Ass 1 out by this Sunday DW in Python
- About 25 weeks
- Make-up class:- Sat Aug 28, 2021 9am-9am

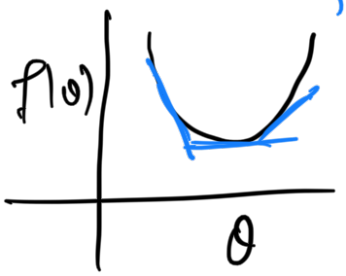
Last class:-

Linear Regression

Convex Functions

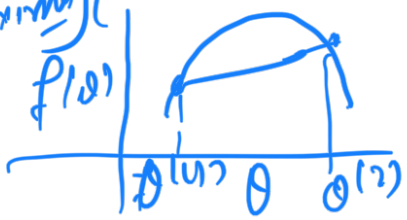
→ $f: R^n \to R$ is convex iff $0 \le \alpha \le 1$

$$f(\alpha\theta^{(1)} + (1-\alpha)\theta^{(2)}) \le \alpha f(\theta^{(1)}) + (1-\alpha) f(\theta^{(2)})$$



Minima of convex Fn:- All local minima = Global minima

Concave Functions



f is concave iff $0 \le \alpha \le 1$

$$f(\alpha\theta^{(1)} + (1-\alpha)\theta^{(2)}) \ge \alpha f(\theta^{(1)}) + (1-\alpha) f(\theta^{(2)})$$

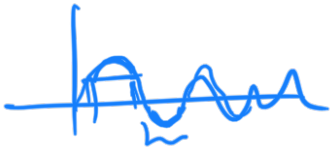
Hessian:- (H)

$z^T H z \ge 0$ (+ve semi-definite)

Hjk = $\frac{\partial^2 f(\theta)}{\partial \theta_j \partial \theta_k}$

$\in R^{(n+1)\times(n+1)}$

Strictly



Hessian:-

$z^T H z \le 0$ −ve semi-definite

Local maxima
Global maxima

Further Topics:-

- SGD (Stochastic Gradient Descent)

② Newton's Method (2nd Order Method)

③ Analytical Solution for Linear Reg

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)} - h_\theta(x^{(i)})\right)^2$$

④ [Probabilistic Interpretation of Linear Regression / Least squared Regression.

↳ MLE :- [Maximum Likelihood Estimation] :- θ_{MLE}

↳ Logistic Regression :-] Classification

↳

Stochastic Gradient Descent :- (Introduction)

GD :-

$t \leftarrow 0$;

$\theta(t) \leftarrow$ init();

do {

$\theta^{(t+1)} \leftarrow \theta^{(t)} - \nabla_\theta J(\theta) \cdot \eta$

$t \leftarrow t+1$;

} while ! converged)

$O(m,n)$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left[y^{(i)} - h_\theta(x^{(i)})\right]^2$$

$$\frac{1}{2m}\sum_{i=1}^{m}\left[y^{(i)} - h_\theta(x^{(i)})\right]\left[\nabla_\theta h_\theta(x^{(i)})\right]^{(-1)}$$

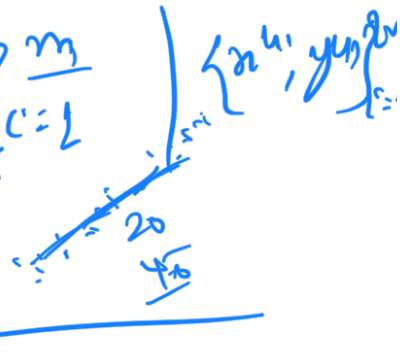
(SGD) :-

n :- # of examples

m :- # of features

SGD :-

$\{\{x^{(i)}, y^{(i)}\}_{i=1}^{m}\}$



$\{x^{(i)}, y^{(i)}\}_{i=1}^{2m}$

$\forall i := (x^{(i)}, y^{(i)}) \sim$ Dist.

i.i.d (identically independently distributed)

Bias -) Variance

$\nabla_\theta J(\theta) \approx \{x^{(i)}, y^{(i)}\}_{i=1}^{|\beta|}$

⇒ SGD 1

$t \leftarrow 0$;
$\theta(t) \leftarrow$ Init();
do {
  $B(t) \leftarrow$ RandomSample($\{x^{(i)}, y^{(i)}\}_{i=1}^{m}, \gamma$);
  $\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \cdot \nabla_\theta f_{B(t)}(\theta)\big|_{\theta^{(t)}}$
  $t \leftarrow t+1$;
} while ! converged)

SGD: 1 will converge (in expectation) to local minima.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)})\right)^2$$

Gradient:

$$\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} - h_\theta(x^{(i)})\right] \cdot \nabla_\theta h_\theta(x^{(i)}) \cdot (-1)$$

$$J_{B(t)}(\theta) = \frac{1}{2\gamma} \sum_{(x^{(i)}, y^{(i)}) \in B(t)} \left(y^{(i)} - h_\theta(x^{(i)})\right)^2$$

$$\nabla_\theta J_{B(t)}(\theta) = \frac{1}{\gamma} \sum_{(x^{(i)}, y^{(i)}) \in B(t)} \left(y^{(i)} - h_\theta(x^{(i)})\right) \nabla_\theta h_\theta(x^{(i)}) \cdot (-1)$$

⇒

$$\nabla_\theta J(\theta) = \mathbb{E}_{B(t)}\left[\nabla_\theta J_{B(t)}(\theta)\right]$$

↓ Expectation

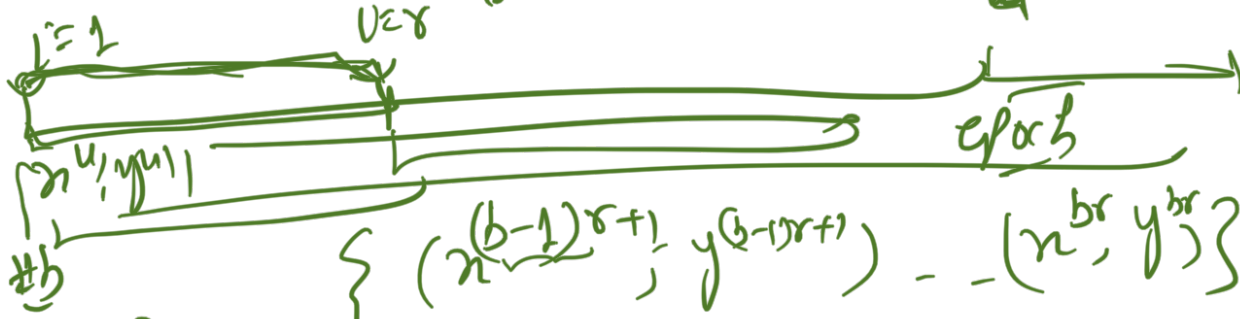
Hint: $B(t) \approx \{x^{(i)}, y^{(i)}\}_{i=1}^{m}$

↳ Batch: ...
(a) Randomly shuffle entire training

data $\{x^{(i)}, y^{(i)}\}_{i=1}^{m}$

typical

(b) $\rightarrow B^{(2)}$ $\rightarrow B^{(m/b)}$

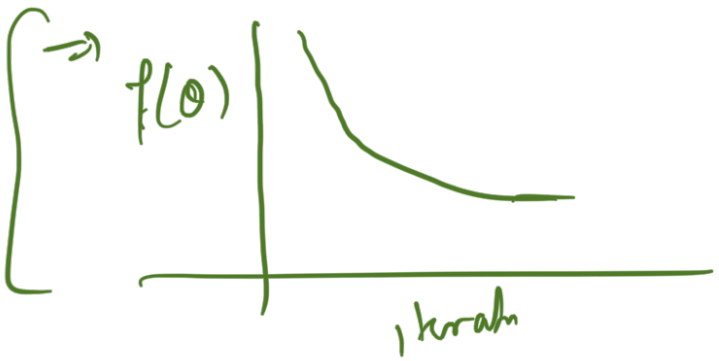


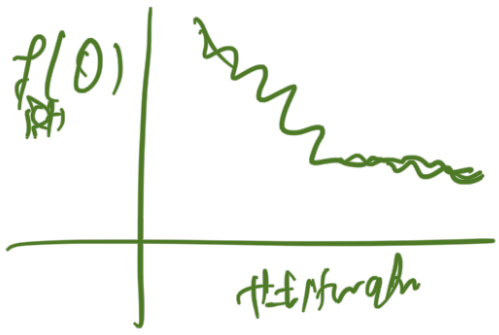
Batch:- $\{(x^{(b-1)b+1}, y^{(b-1)b+1}) \ldots (x^{br}, y^{br})\}$

B:-

$\Rightarrow \equiv$ when examples are randomly sampled

$\Rightarrow$ GPU:- parallelization:-



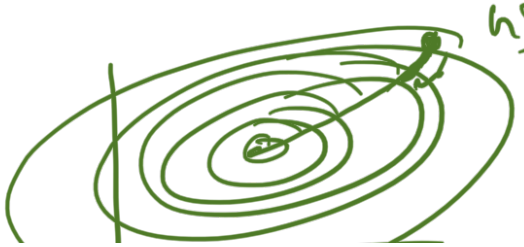


① Why is SGD helpful?

Each parameter update is cheap $\rightarrow$ Time taken for each parameter update? $O(n)$

$n << m$:- $\rightarrow$ Approx. SGD

↳ 100, 1000



SGD:- { Each update is much faster
:- stochasticity

⇒ SGD:: same generalization GD