WEEK - 1

Date: / / Page no:

* Machine Learning Algorithms

E T P

E → Experience (No of games)

T → Task of Playing Chess

P → Performance Probability of winning.

① Supervising Learning → Logist Regression, classification Problem.

② Unsupervising Learning → find Structure (Cluster)

* Linear Regression with One Variable

① Model And Cost Function:

m → No. of training examples.

x's → input variable / features

y's → output variable / target

(x, y) → one training example.

(x^i, y^i) → i^{th} training ex.

Training Set

Size in feet (x's)	Price (y's)
2104	460
1416 (x^2)	232 (y^2)
⋮	⋮
m=10	

Date: / / Page no:

Training Set

Learning Algorithm

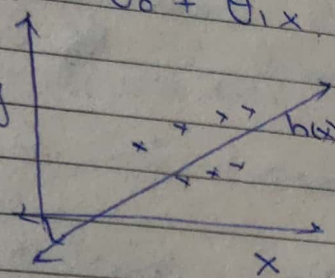
Size of House (x) → h → Estimated price (y)

hypothesis function.

$h_\theta(x)$

Linear case

$h_\theta(x) = \theta_0 + \theta_1 x$



θ_i's → Parameters.

Goal

↳ Minimize $\underset{\theta_0,\theta_1}{\text{Minimize}} \frac{1}{2m} \sum_{i=1}^{m} \left[h_\theta(x^i) - (y^i) \right]^2$

cost function. or Sq. error function

$$J(\theta_0,\theta_1) = \frac{\left[h_\theta(x^i) - (y^i) \right]^2}{2m}$$

$\underset{\theta_0,\theta_1}{\text{Minimize}}\ J(\theta_0,\theta_1)$

$J(\theta_0,\theta_1)$ = Min θ_0

$\left(\frac{1}{2\times3}\right)(1+4+9)$

$14 \left(\frac{9}{3}\right) \approx 2.3$

let

$h_\theta(x) = \theta_1 x$.

$J_{\theta_1}(x) = \sum_{i=1}^{m} \frac{(\theta_1 x^i - y^i)^2}{2m}$

② Gradient Parameter Learning:

• gradient descent to minimization.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0,\theta_1)$$

(2x-1).
(1)

Date: / / Page no:

Correct Simultaneous Update

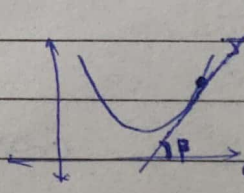
$temp0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$

$temp1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$

$\theta_0 := temp0$

$\theta_1 := temp1$

Let $\min_{\theta_1} J(\theta_1)$



$\left[\theta_1 := \theta_1 - \alpha \frac{dJ(\theta_1)}{d\theta_1}\right]$

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \left(\frac{1}{2m}\right) \sum_{i=1}^{m} \left[h_\theta(x^i) - (y^i)\right]^2$$

$$= \frac{\partial}{\partial \theta_j} \cdot \left(\frac{1}{2m}\right) \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^i - y^i\right)^2$$

~~$\theta_0 (y^i - 0) := \theta - (x^i)$~~

$$\theta_0 \; j=0 : \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \left(\frac{1}{m}\right) \sum_{i=1}^{m} \left(h_\theta(x^i) - y^i\right)$$

$$\theta_1 \; j=0 \quad \frac{\partial}{\partial \theta_1} (J\theta_0, \theta_1) = \left(\frac{1}{m}\right) \sum_{i=1}^{m} \left(h_\theta x^i - y^i\right)(x^i)$$

Batch Gradient Descent → all train set in each step

Date: / / Page no:

③ Linear Algebra Review :

$h_\theta(x) = -40 + 36x$ $\quad x = 2104, 1416, 1534, 852$ $\quad y = ?$

$$A = \begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix}_{4\times 2} \qquad \theta = \begin{bmatrix} -40 \\ 36 \end{bmatrix}_{2\times 1}$$

$C = A \times \theta$

Prediction = Data Matrix × Parameter

WEEK - 2

① Multivariate Linear Regression. $(x_1, x_2, \ldots, x_n)$

x_1	x_2	x_3	x_4	y
-	-	-	-	-

$n \rightarrow$ No. of features.

Vector $\rightarrow x^{(i)} \rightarrow$ input (features) of i^{th} example.

$x_j^{(i)} \rightarrow$ Value of feature j in i^{th} example.

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

for convience $x_0 = 1$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}, \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$h_\theta(x) = \theta^T x$$

Date: / / Page no:

- Use gradient descent for Multivariate linear regression.

$$h_\theta(x) = (\theta^T x)$$

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

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

↳ Then

$$\theta_j := \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} \left[h_\theta(x^i) - y^i \right] (x_j^i)$$

- Scaling the feature

x_1 = Size (0 - 2000 feet²)

x_2 = No. of bedroom (1-5)

(Apply Scaling.

$$x_1 = \frac{\text{Size}}{2000} \qquad x_2 = \frac{\text{No. of bedroom}}{5}$$

→ Take Mean Normalization also.

$x_i \rightarrow \frac{x_i - \mu_i}{s_i}$ (average value, range / standard deviation)

$x_2 \rightarrow \frac{\#\text{bedroom} - 2}{5}$

$x_1 \rightarrow \frac{x_1 - 1000}{2000}$

- Learning rate Alpha :-

- Featured & Polynomial Regression :-

Date: / / Page no:

② Computing Parameters Analytically :-

• Normal eq. method to get optimal θ.

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

$$\frac{\partial}{\partial \theta_j} J(\theta) = 0 \rightarrow (\text{for every } j)$$

$$X = \begin{bmatrix} 1 & x_1^1 & x_2^1 & x_3^1 & x_4^1 \\ 1 & - & - & - & - \\ 1 & - & - & - & - \\ 1 & x_1^4 & x_2^3 & x_3^3 & x_4^4 \end{bmatrix}_{m \times (n+1)} \qquad y = \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^m \end{bmatrix}_{m \times 1}$$

$$\theta = (X^T X)^{-1} (X^T)(y)$$

$(n+1) \quad (m \times 1)$

→ optimal answer. Scaling not necessarily.

WEEK - 3.

① classification and Representation.

* Logistic Regression :- $0 \leq h_\theta(x) \leq 1$

$y \in \{0, 1\}$ → binary class.

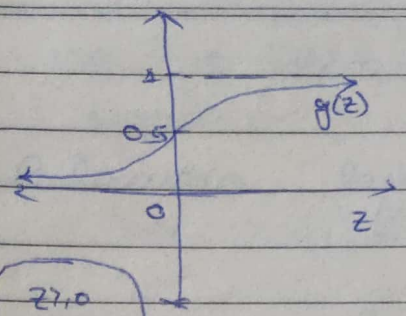
$$h_\theta(x) = g(\theta^T x)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$$\Rightarrow h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

100×3 100×1

Date: / / Page no:



$z \geq 0$
$g(z) \geq 0.5$

$h_\theta(x) = P(y=1|x;\theta)$

↳ Probability $y=1$ for given x, parameterized by θ.

↳ feature.

$P(y=0|x;\theta) + P(y=1|x;\theta) = 1$

$\theta^T x \geq 0$ Predict $y=1$

$\theta^T x < 0$ Predict $y=0$

← Decision boundary.

② Logistic Regression Model

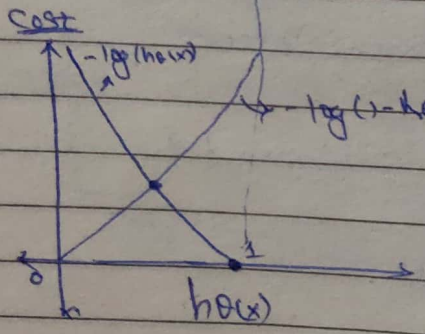
• Cost function:

$x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$ $x_0 = 1$ $y \in [0, 1]$ $\mathbb{R}^{n+1}$

Linear Regression cost func: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \text{cost}(h_\theta(x^i) - y^i)$

$\text{cost}(h_\theta x^i - y^i) = \frac{1}{2}(h_\theta x^i - y^i)^2$

$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$



$$\text{Cost}(h_\theta x, y) = \begin{cases} -\lg(h_\theta(x)) & \text{if } y=1 \\ -\lg(1-h_\theta(x)) & \text{if } y=0 \end{cases}$$

$$\text{Cost}(h_\theta(x), y) = -y \lg(h_\theta(x)) - (1-y) \lg(1-h_\theta(x))$$

60×1

- conjugate gradient
- BFGS
- L-BFGS

Date: / / Page no:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^i \log_{10} h_\theta(x^i) + (1-y^i)\log(1-h_\theta(x^i))\right]$$

↑ log principle of maximum likelihood.

$\min_\theta J(\theta)$ ← use gradient descent

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

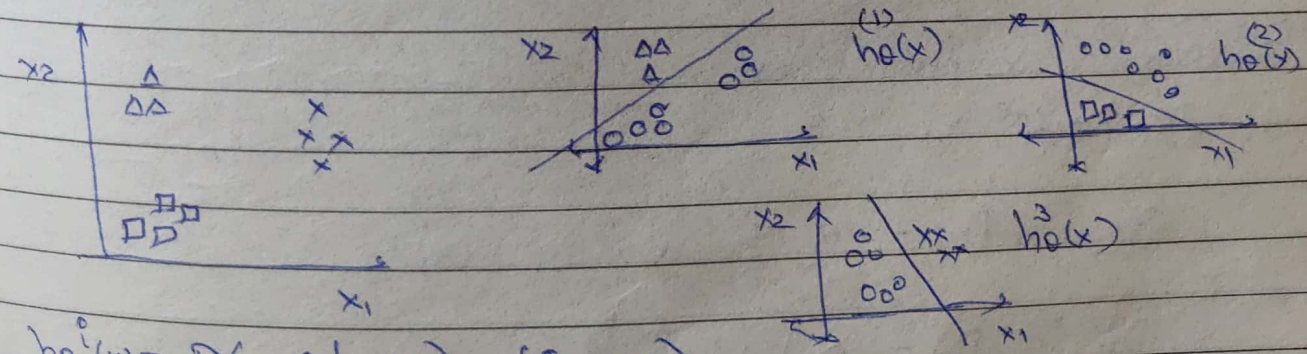
$$\theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}\left[h_\theta(x^i) - y^i\right](x_j^i)$$

↳ Gradient descent for Logistic ~~Regression~~

Vectorized implementation

$$\theta = \theta - \frac{\alpha}{m} X^T (g(X\theta) - \vec{y})$$

③ Multiclass Classification



$h_\theta^i(x) = P(y=i \mid x;\theta) \quad (i=1,2,3)$

$\max_i h_\theta^i(x)$

$h_x = 12\times1$

$X = 12\times2$
$y = 12\times1$
theta $= 2\times1$
grad $= 2\times1$

$(2\times2) \times ([12\times2][2\times1] - [12\times1]) + [1\times1]$

Date: / / Page no:

(5) Solving the problem of High variance → overfitting. High bias → underfitting. 12×1 Overfitting : 2×1 (2×1)

→ Solving overfitting
(i) Reduce No of features.
→ Manually → Model Selection Algo (later).
(ii) Regularization.
→ Keep all features but reduce their magnitude

• Regularization.
features → $x_1, x_2, \dots, x_{100}$
Parameter → $\theta_1, \theta_2, \dots, \theta_{100}$
for linear Regularization.

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

(Regularization; λ ↑ Parameter)

Gradient descent after Regularization

$$\theta_0 := \theta_0 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta x^i - y^i)(x_0^i)$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta x^i - y^i) + \frac{\lambda}{m}\theta_j\right]$$

Date: / Page no:

Regularization for logistic regression.

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

Gradient descent

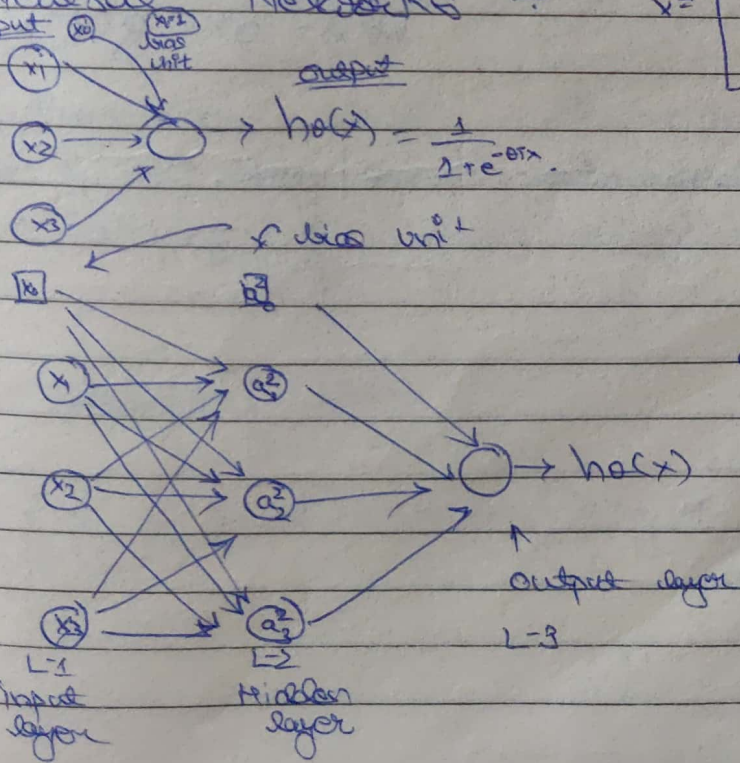
$$\theta_0 := \theta_0 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_0^i$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_j^i + \frac{\lambda}{m}\theta_j^2\right]$$

WEEK-4.

- Non-linear hypothesis need because n features we have to compute $O(n^2)$ total possibilities.

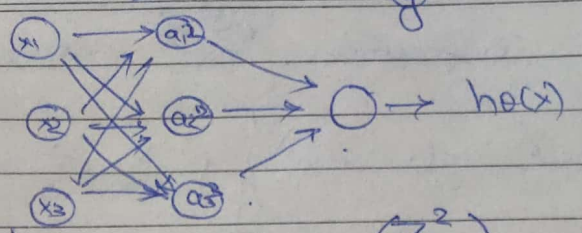
① Neutral Networks :- $x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \end{bmatrix}$ $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \end{bmatrix}$ weight / parameter.



a_i^j ⇒ activation of unit i in layer j.

θ^j ⇒ matrix of weights controlling function mapping from layer j to $j+1$

Date: / / Page no:



$\theta^1 \in R^{3\times 4}$

$a^1 = x$

z_1^2

$a_1^2 = g(\theta_{10}^1 x_0 + \theta_{11}^1 x_1 + \theta_{12}^1 x_2 + \theta_{13}^1 x_3)$

$a_2^2 = g(z_2^2)$

$a_3^2 = g(z_3^2)$

$h_\theta(x) = a_1^3 = g(\theta_{10}^2 a_0^2 + \theta_{11}^2 a_1^2 + \theta_{12}^2 a_2^2 + \theta_{13}^2 a_3^2)$

$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$ $z^2 = \begin{bmatrix} z_1^2 \\ z_2^2 \\ z_3^2 \end{bmatrix}$

$z^2 = (\theta^1 x)$ or $(\theta^1 a^1)$

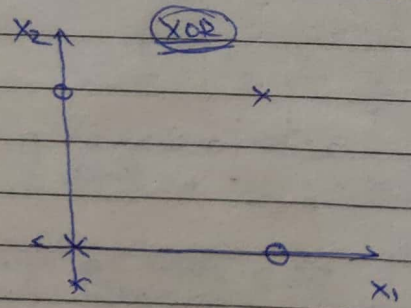
$a^2 = g(z^2)$

Add $a_0^2 = 1$

$z^3 = (\theta^2)(a^2)$

$h_\theta(x) = a^3 = g(z^3)$

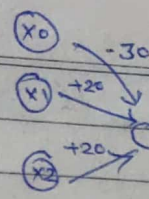
- Examples:
- Non-linear classification XOR/XNOR.



Date: / / Page no:

$x_1, x_2 \in \{0, 1\}$

$y = x_1$ AND x_2

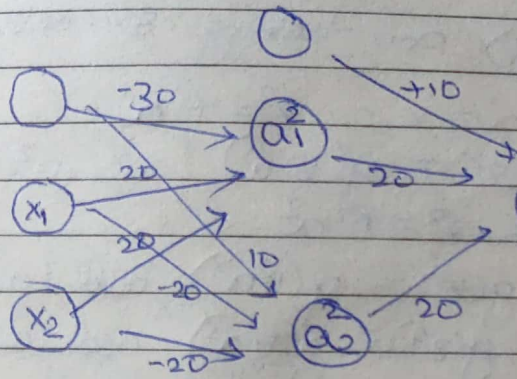


θ_{10}

$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$

x_1	x_2	$h_\theta(x)$
0	0	$g(-30) = 0$
1	0	$g(-10) = 0$
1	1	$g(10) \approx 1$
0	1	$g(-10) = 0$

XOR = (A ~~And~~ B) · ($\bar{A}$ ~~And~~ $\bar{B}$)



x_1	x_2	a_1^2	a_2^2	$h_\theta(x)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

WEEK 5

① Cost function And Back ~~propagation~~

S_L = No. of unit (Not biased) in layer L.

Cost fn $\Rightarrow$ $J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^i \log h_\theta(x^i) + (1-y^i)\log(1-h_\theta(x^i))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$

Neutral Network $R^{(K)}$

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{K=1}^{K} y_K^i \log h_\theta(x^i)_K + (1-y_K^i)\log(1-h_\theta(x^i)_K)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{S_l}\sum_{j=1}^{S_{l+1}}\left[\theta_{ji}^{(l)}\right]^2$$

$$\frac{3(1.01)^4 + 4 - [(3)(0.99)^4 + 4]}{(0.02)}$$

$$\frac{3[(1.01)^4 - (0.99)^4]}{(0.02)}$$

Date: / / Page no:

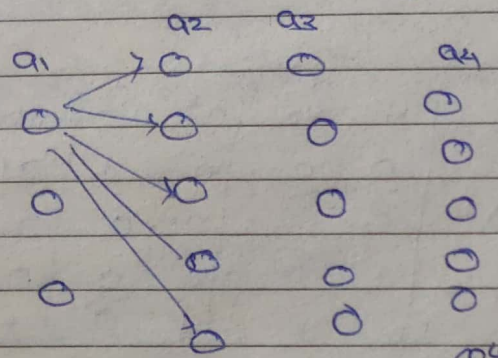
- Back propagation Algorithm :-

$\min_{\theta} J(\theta)$

$J(\theta)$

$\frac{d}{d\theta_j^{(l)}} J(\theta)$

Forward Propagation.



(x, y) are training Example

$a^1 = x$ $z^2 = \theta^1 a^1$

$a^2 = g(z^2) = g(\theta^1 a^1)$ add (a_0^2)

$z^3 = \theta^2 a^2$

$a^3 = g(z^3) = g(\theta^2 a^2)$ add (a_0^3)

$a^4 = g(z^4) = g(\theta^3 a^3)$ add (a_0^4)

Backpropagation Algorithm

δ_j^l = error in node j in layer l.

$(h_\theta(x))_j$

$\delta_j^4 = a_j^4 - y_j$ $\quad \delta^4 = a^4 - y$.

$\delta^3 = (\theta^3)^T \delta^4 .* g'(z^3)$ $\quad a^3 .* (1 - a^3)$

$\delta^2 = (\theta^2)^T \delta^3 .* g'(z^2)$

No δ^1 because no error.

15
24
39

Date: / / Page no:

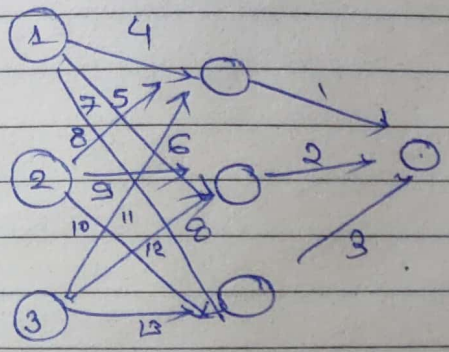
$\frac{d}{d\theta_{ji}^{l}} J(\theta) = a_j^{l} \delta_i^{(l+1)}$ (ignoring λ; if $\lambda = 0$)

** Complete Backpropagation watch Jaichakar **

② Back propagation in detail :-

- gradient checking :-

X



$a_1^2 = g(4)$

$\theta^1 = [\quad]$

$a^2 = [\quad]$ 25×5000

5000×25

5000×25

$x = [\quad]$ 5000×400

$y = [\quad]$ 5000×1

$h(\theta) = [\quad]$ 10×5000

$\theta_1 = [\quad]$ 25×401

$\theta_2 = [\quad]$ 10×26

WEEK-6

DATE
PAGE

- What should we try?
  - Get More training examples → fix high variance check
  - Try Smaller Set of features → fix high variance
  - Try getting additional features → fix high bias
  - Try adding Polynomial features → fix high bias
  - Try decreasing λ → fixes high bias
  - Try increasing λ → fixes high variance

- Evaluating a Hypothesis:
  divide Training Set [ ] Training Set 70%.
  & Pass [ ] Testing Set 30%. (randomly)

Computing Test error

$$J_{test}(\theta) = \frac{1}{2 M_{test}} \sum_{i=1}^{M_{test}} \left(h_\theta(x_{test}^{i}) - y_{test}^{i} \right)^2$$

for logistic Test error:=

$$J_{test}(\theta) = \frac{-1}{m_{test}} \sum_{i=1}^{m_{test}} \left[y_{test}^{i} \log h_\theta(x_{test}^{i}) + (1 - y_{test}^{i}) \log h_\theta(x_{test}^{i}) \right]$$

for classification - Mis classification:

$err(h_\theta(x), y) = 1$ if $h_\theta(x) \geq 0.5$ & $y = 0$, OR, $h_\theta(x) < 0.5$ & $y = 1$

$= 0$ otherwise.

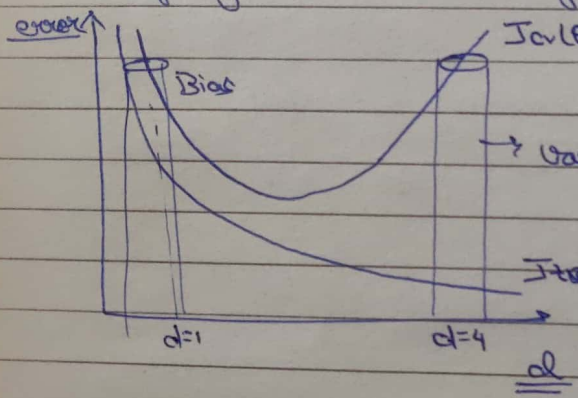
$$\text{Test Error} = \frac{1}{m_{test}} \sum_{i=1}^{m_{test}} err(h_\theta(x_{test}^{i}), y_{test}^{i})$$

DATE
PAGE

- Model Selection and Training Set.

Breaking of Training Set :
① Training Set (60%.) → optimize the (θ)
② Cross Validation Set (20%.) → final Polynomial degree of
③ Test Set (20%.) → $J_{test}(\theta)$

Bias VS Variance :

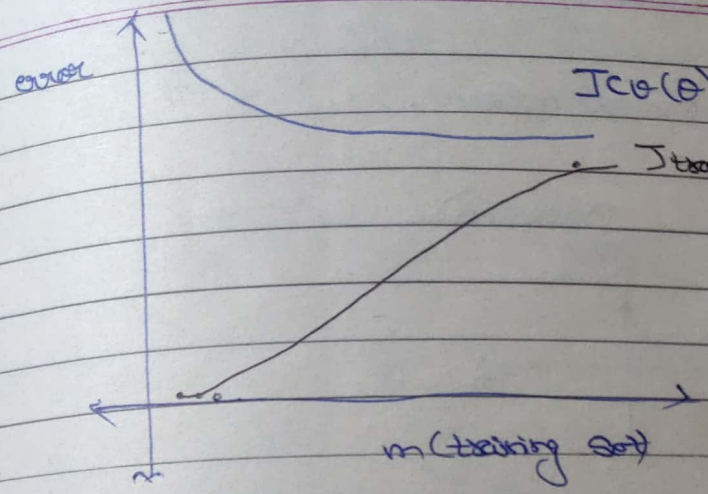


- Bias → J_{train} & J_{cv}
  $J_{train}(\theta)$ ↑ $J_{cv}(\theta)$ ↑
- Variance →
  $J_{train}(\theta)$ ↓ $J_{cv}(\theta)$ ↑
  $J_{cv} >> J_{train}$

- Learning Curves :

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$$

$$J_{cv}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta x^i_{cv} - y^i_{cv})^2$$



Machine Learning Design System :-

① Build a spam classifier :-

② Handling skewed data :-

		Actual class 1	Actual class 0
Predicted class	1	True +ve	False +ve
	0	False -ve	True -ve

$$\text{Precision} = \frac{\text{True +ve}}{\text{True +ve} + \text{False +ve}}$$

$$\text{Recall} = \frac{\text{True +ve}}{\text{True +ve} + \text{False -ve}}$$

Trading off b/w precision and Recall

use F-Scores. $\frac{2ab}{(a+b)}$

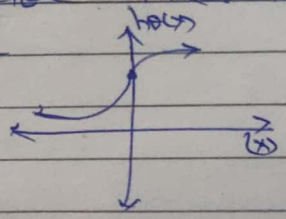
WEEK-7

Large Margin classification.

- SVM (Support vector Machine)

$z = \theta^T x$

$h_\theta(x) = \frac{1}{1+e^{-z}}$



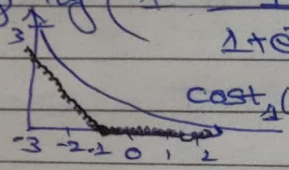
if $y=1$ $[\theta^T x \geq 0]$

$y=0$ $[\theta^T x < 0]$

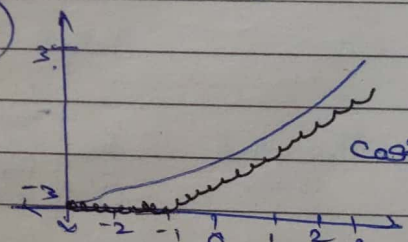
from logistic Regression

$J_\theta(x) = -y \log \frac{1}{1+e^{-\theta x}} - (1-y) \log\left(1 - \frac{1}{1+e^{-\theta x}}\right)$

$y=1$ $J_\theta = -\log\left(\frac{1}{1+e^{-\theta x}}\right)$



$y=0$ $J_\theta = -\log\left(1 - \frac{1}{1+e^{-\theta^T x}}\right)$



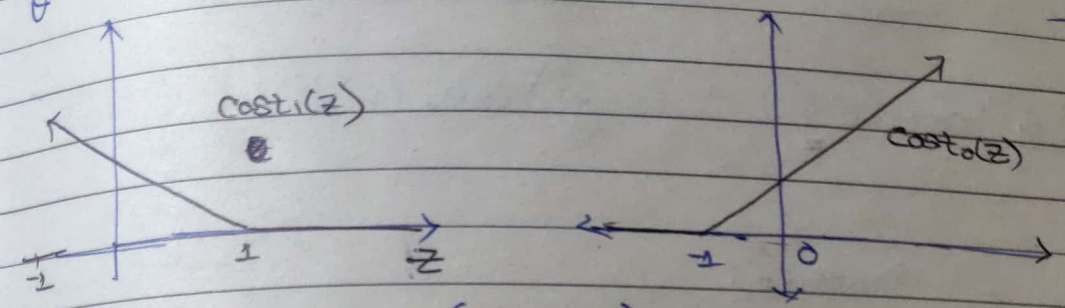
SVM Cost function :-

$$J(\theta) = \min_\theta C\left[\sum_{i=1}^{m} y^i \text{cost}(\theta^T x^i) + (1-y^i)\,\text{cost}(\theta^T x^i)\right] + \frac{1}{2}\sum_{i=1}^{n} \theta_j^2 \quad (2)$$

$$h_\theta(x) = \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{if } \theta^T x < 0. \end{cases}$$

(1)

$$\min_{\theta} C \sum_{i=1}^{m} \left[y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)}) \text{cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$



if $y=1$ $z \gg 1$ $(\theta^T x \gg 1)$

$y=0$ $z \ll -1$ $(\theta^T x \ll -1)$

Kernels :-

$f_1 = x_1$ $f_2 = x_2$ $f_3 = x_3 x_2$ .... (features)

↳ To Think better choice of features.

↳ Gaussian Kernel.

$$f_1 = \text{Similarity}(x, l^{(1)}) = \exp\left[\frac{-\|x - l^{(1)}\|^2}{2\sigma^2} \right] = \exp\left[\frac{-\sum_{j=1}^{n} (x_j - l_j^{(1)})^2}{2\sigma^2} \right]$$

↳ landmark

$$f_i = K(x, l^{(i)})$$

① $x \approx l^{(1)}$

↳ $f_1 \approx 1$

② x is far $l^{(1)}$

↳ $f_1 \approx 0$

Fedram

DATE
PAGE

How to get landmark.

$$\min_{\theta} C \sum_{i=1}^{m} (y^i)\, \text{cost}_1(\theta^T f^i) + (1-y^i)\, \text{cost}_0(\theta^T f^i) + \frac{1}{2} \sum_{j=1}^{m} \theta_j^2$$

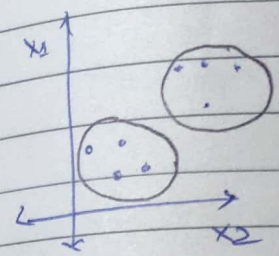
SVM in Practice :-
liblinear, libsvm

① Choice of Parameter C
② Choice of Kernel.

WEEK-8

DATE
PAGE

Clustering (Unsupervising learning)



K-Means Algorithm :-

1. cluster assignment
2. Move Centroid.

- Objective function of K-mean.

$c^i \Rightarrow$ (1 to K) No. of cluster.

$\mu_k \Rightarrow$ cluster centroid. ($\mu_k \in R^n$)

$\mu_{c^{(i)}} \Rightarrow x^i \Rightarrow 5 \quad \mu_{c^i} = \mu_5.$

$c^i = 5$

- Objective function →

$$J(c^1, \ldots, c^m; \underbrace{\mu_1 \cdots \mu_K}_{K}) = \frac{1}{m} \sum_{i=1}^{m} \|x^i - \mu_{c^i}\|^2$$

$$\min_{\substack{c^1,\ldots,c^m \\ \mu_1,\ldots,\mu_K}} J(c^1, \ldots, c^m, \mu_1, \ldots, \mu_K)$$

Dimensionality Reduction (unsupervised).

(↓ more feature → to → less features.

3D ↔ 2D ↔ 1D

Principle Component Analysis :

[PCA] (reduce dimension).

Reduce n Vector to k- Vector. Project k vector $u^1, u^2, \ldots, u^k$ to Minimize errors.

① Preprocessing (Mean Normilization ↔ Scaling)

② n- dimension to k- dimension

$$\text{Sigma} = \frac{1}{m} \sum_{i=1}^{n} \underset{n\times 1}{(x^i)} \underset{1\times n}{(x^i)^T} \quad (n\times n) \text{ matrix}$$

compute eigenvector

*** [u,s,v] = (Singular Value decomposition) svd(Sigma);

$u \to [n \times n] \to \begin{bmatrix} | & | & & | \\ u^1 & u^2 & \cdots & u^n \\ | & | & & | \end{bmatrix}$

$x \in R^n \to z \in R^k$

$z = \begin{bmatrix} | & | & & | \\ u^1 & u^2 & \cdots & u^k \\ | & | & & | \end{bmatrix}^{n\times k}$ ureduce

ureduce = u(:,1:k);

z = ureduce' * x;

$x = (z^T)(z) \Rightarrow k \times 1$

Reconstruction from compressed representation :-

$$x_{approx}^{n} = U_{reduce}^{n \times k} \cdot z^{(i)}_{k \times 1}$$

WEEK-9

Density Estimation :

Q density ka outside stand karna
Q Anomaly detection.

$p(x) < \varepsilon$

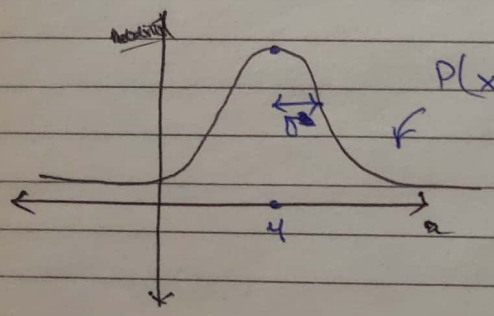
Gaussian (Normal) Distribution :

Say $x \in R$. If x is a distributed Gaussian with Mean μ, variance σ^2.

x distributed as

$$x \sim N(\mu, \sigma^2)$$

$$\mu = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)})$$



$$P(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu)^2$$

Anomaly Detection Algorithm :

(for very few +ve examples) (Not use Supervising Learning Algorithm)

$$P(x_1; \mu_1, \sigma_1^2) \times P(x_2; \mu_2, \sigma_2^2) \cdots \times P(x_n; \mu_n, \sigma_n^2)$$

$$\Rightarrow \prod_{j=1}^{n} P(x_j; \mu_j, \sigma_j^2)$$

Anomaly detection
- fraud detection
- Manufacting (aircraft)
- Monitoring Machine

Supervised Learning
- Email spam
- weather prediction
- Cancer classification

Multivariate Gaussian Distribution :

①

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

$\mu \in \mathbb{R}^n$ $\quad \Sigma \in \mathbb{R}^{n \times n}$

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$$

$$\Sigma = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu)(x^{(i)} - \mu)^T$$

② Now on new example
$(p(x) < \varepsilon)$

* Multivariate Gaussian will be original Gaussian variation if their contour plot are axis aligned mean

$$\Sigma = \begin{bmatrix} \sigma_1^2 & & 0 \\ & \sigma_2^2 & \\ 0 & & \sigma_n^2 \end{bmatrix}$$

Recommender Systems :

n_u → No. of users
n_m → No. of Movies.
$r(i,j)$ → 1 if user j rated Movie i.
$(y^{i,j})$ → rating user j to movie i (if $r(i,j)=1$).

Movie	A	B	C	D	x_1 (Romance)	x_2 (Action)
1 x^1	5	5	0	0	0.9	0
2 x^2	5	? 4.5	? 0	0	1.0	0.01
3	? 5	4	0	? 0	0.99	0
4	0	0	5	4	0.1	1.0
5	0	0	5	? 4	0	0.9

① Content based recommend :

x_1, x_2 feature. $x' = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix}$ ← intercepter

$n=2$ (feature)

* for each user j, learn Parameter $\theta^j \in R^{n+1}$. Predict user j as rating movie i with $(\theta^j)^T (x^i)$ stars.

eg:- $x^3 = \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix}$ $\theta^1 = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$ $(\theta^1)^T (x^3) = (5)(0.99) = 4.95$.

* To learn θ^j (eg linear Regression) :

$$\min_{\theta^j} \frac{1}{2m^j} \sum_{i:r(i,j)=1} \left[(\theta^j)^T (x^i) - y^{i,j} \right]^2 + \frac{\lambda}{2m^j} \sum_{k=1}^{n} (\theta_k^j)^2$$

$\theta^j \in R^{n+1}$

Formally
To learn (θ^j)

$$\min_{(\theta^j)} \left(\frac{1}{2}\right) \sum_{i:r(i,j)=1} \left[\left((\theta^j)^T(x^i) - (y^{i,j})\right)^2\right] + \frac{\lambda}{2}\sum_{k=1}^{n}(\theta_k^j)^2$$

To learn $\theta^1, \theta^2, \ldots (\theta^{n_u})$.

$$\min_{\theta^1,\ldots\theta^{n_u}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left[(\theta^j)^T(x^i) - (y^{i,j})\right]^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^j)^2.$$

$$J(\theta^1, \ldots, \theta^{n_u}).$$

To Gradient descent update :-

$$(\theta_k)^j = \theta_k^j - (\alpha)\sum_{i:r(i,j)=1}\left[(\theta^j)^T x^i - (y^{i,j})\right](x_k^i) \quad (k=0)$$

$$(\theta_k)^j = \theta_k^j - \alpha\left[\underbrace{\sum_{i:r(i,j)=1}\left((\theta^j)^T(x^i) - y^{i,j}\right)(x_k^i) + \lambda\theta_k^j}_{\frac{\partial}{\partial\theta_k^j}J(\theta^1,\ldots\theta^{n_u})}\right] \quad (k\neq 0)$$

(2) Collaborative filtering Algorithm :-
(automatic featuring adding).

Given $(\theta^1, \ldots, \theta^{n_u})$ to learn (x^i)

$$\min_{x^i} \frac{1}{2}\sum_{j:r(i,j)=1}\left((\theta^j)^T(x^i) - y^{i,j}\right)^2 + \frac{\lambda}{2}\sum_{k=1}^{n}(x_k^i)^2.$$

Given $(\theta^1, \ldots, \theta^{n_u})$ to learn $x^1, \ldots, x^{(nm)}$.

$$\min_{(x^1, \ldots x^{nm})} \frac{1}{2} \sum_{i=1}^{nm} \sum_{j:r(i,j)=1} \left[\left((\theta^j)^T (x^i) - y^{i,j} \right)^2 \right] + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} (x_k^i)^2$$

Gradient descent think yourself.

Both ① & ② use algo.

$\theta \to x \to \theta \to x \to \theta \to x$

↑ Guessing ~~not~~.

(OR) Combining both the eqn:

$$J[(x^1, \ldots, x^{nm})(\theta^1, \ldots, \theta^{n_u})] = \frac{1}{2} \sum_{(i,j):r(i,j)=1} \left[(\theta^j)^T (x^i) - (y^{i,j}) \right]^2 + \left(\frac{\lambda}{2} \sum_{i=1}^{nm} \sum_{k=1}^{n} (x_k^i)^2 + \left(\frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} (\theta_k^j)^2 \right) \right.$$

$$\min_{\substack{(x^1, \ldots x^{nm}) \\ (\theta^1, \ldots, \theta^{n_u})}} J[(x^1, \ldots, x^{nm})(\theta^1, \ldots \theta^{n_u})]$$

$\theta \to x \to \theta$.

① Initialize $(x^1, \ldots, x^{nm})(\theta^1, \ldots \theta^{n_u})$ to small random value.

② Min $J(x^1 \ldots x^{nm}, \theta^1 \ldots \theta^{n_u})$

$$x_k^i = x_k^i - \alpha \left[\sum_{j:r(i,j)=1} \left((\theta^j)^T (x^i) - (y^{i,j}) \theta_k^j + \lambda x_k^i \right) \right]$$

$$\theta_k^i = \theta_k^j - \alpha \left[\sum_{i:r(i,j)=1} \left[(\theta^j)^T x^i - (y^{i,j}) \right] x_k^i + \lambda \theta_k^i \right]$$

Vectorization implementation :-
(Low Rank Matrix)

$$X = \begin{bmatrix} -(x^{(1)})^T - \\ -(x^{(2)})^T - \\ \cdots \\ -(x^{(n_m)})^T - \end{bmatrix} \qquad \Theta = \begin{bmatrix} -(\theta^{(1)})^T - \\ -(\theta^{(2)})^T - \\ \cdots \\ -(\theta^{(n_u)})^T - \end{bmatrix}$$

Predicting Rating $= X\Theta^T$

Mean Normalization :-
(जब Rating नहीं दी गयी)

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \qquad \mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix} \Rightarrow Y = \begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & 1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

for user j on movie i

Learn (θ^j, x^i)

$$(\theta^j)^T (x^i) + (y_i^j)$$

$$\theta = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

~~$\theta^T x^i$~~

DATE
PAGE

WEEK-10

★ dealing with largeer dataset :-

① Plot Learning curve & check. coherse adding Training set curve help or not.

Stochastic Gradient descent :-

Batch

$h_\theta(x) = \sum_{j=0}^{n} \theta_j x_j$

$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$

{

$\theta_j := \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} (h_\theta x^i - y^i) x_j^i$ → expensive

$(j = 0 \cdots n)$

300,000,000

}

Stochastic

$cost(\theta, (x^i, y^i)) = \frac{1}{2}(h_\theta x^i - y^i)^2$

$J_{train}(\theta) = \frac{1}{m} \sum_{i=1}^{m} cost(\theta, (x^i, y^i))$

① Randomly Shuffle dataset.

② Repeat {

for $i = 1$ to m {

$\theta_j = \theta_j - \alpha (h_\theta x^i - y^i) x_j^i$

for $(j = 0$ to $n)$

}

Mini-batch Gradient descent :-

b/w (batch & stochastic)

$[b = 2\text{-}100]$

$(b = 10)$ $m = 1000$

Repeat {

for $i = 1, 11, \cdots$ {

$\theta_j := \theta_j - \frac{\alpha}{10} \sum_{k=i}^{i+9} (h_\theta x^k - y^k)(x_j^k)$

(for every $j = 0, \cdots, n$)

}

}

DATE
PAGE

WEEK-11

Photo OCR :-

Pipeline :- Image → Text detection → Char. Segmentation → Character Recognition