Date: / /
Page No.

LOGISTIC REGRESSION

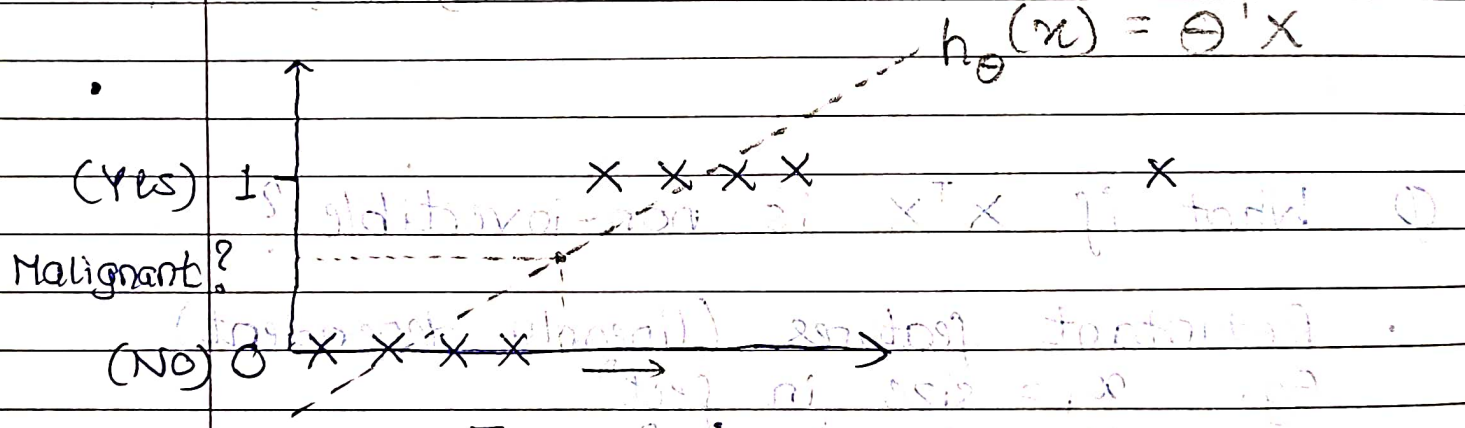
Classification

Email : Spam / Not Spam ?
Online Transactions : Fradulent (Yes/No) ?
Tumor : Malignant / Benign.

$y \in \{0, 1\}$ 0: "Negative Class" (Benign)
1: "Positive Class" (Malignant)

y may even take more values

$y \in \{0, 1, 2, 3\}$ → Multiclass Classification.



Now, one thing that we could do is to apply Linear Regression to this data set and try to fit a straight line.

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

Date: / /
Page No.

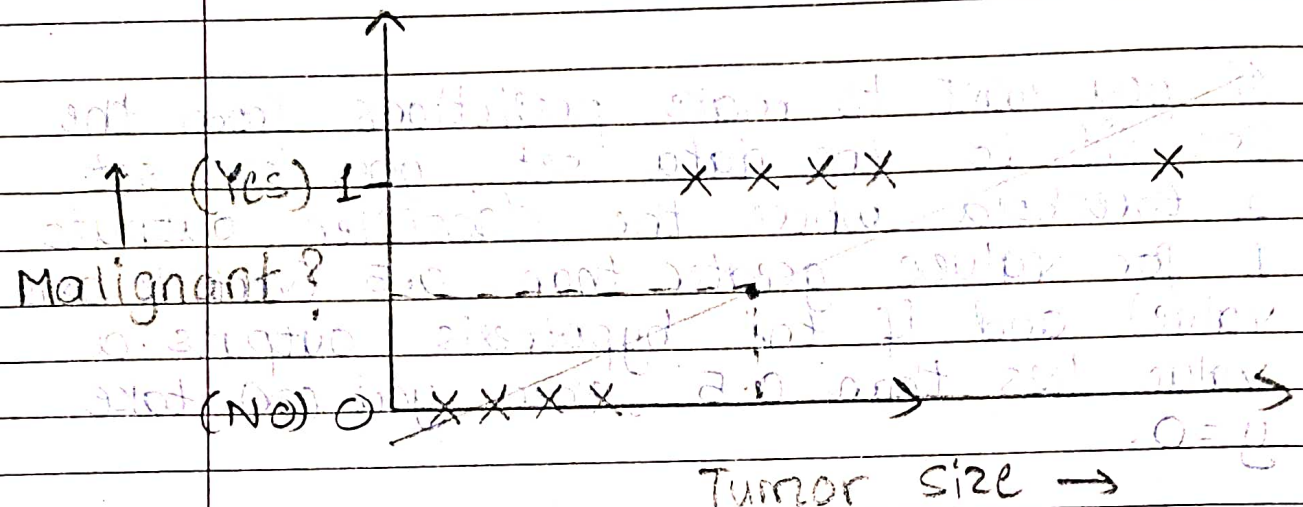
If you want to make predictions from the line fit to the data set, you can set a threshold where the classifier outputs 1 for values greater than 0.5 (verticle axis value) and if the hypothesis outputs a value less than 0.5, then you can take $y = 0$.

Theshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict '$y = 1$'
If $h_\theta(x) < 0.5$, predict '$y = 0$'

Till here, it looks like linear regression is giving us some useful results even in a classification problem.
But now, let's add one more training example.

Notice that, intuitively the new training example doesn't change anything and it's pretty clear that the tumor is malignant. But, now when we fit our line to the data our model might come out to be somewhat like this



And now, clearly, this hypothesis is not accurate since it will predict a malignant tumor as benign since is falls in the (<0.5) category. So, linear regression fails here.

You might get lucky if you use it.

Also, $h_\theta(x)$ can be >1 or <0

Logistic Regression is a classification algorithm and we will use it.

- Logistic Regression Model

Want $0 \leq h_\theta(x) \leq 1$

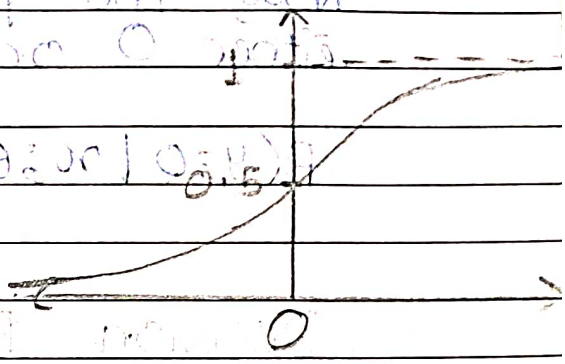
Date: / /
Page No.

Logistic Regression Model

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

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

Sigmoid / Logistic Function



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

∴ The hypothesis is a function of $(\theta^T X)$ that is $g(\theta^T X)$ where $g(z) = \frac{1}{1+e^{-z}}$

Clearly, $g(z)$ will be in range $[0,1]$ for all values of z from $-\infty$ to $+\infty$.

- Interpretation of Hypothesis Output.

$h_\theta(x)$ = estimated probability that $\boxed{y=1}$ on input x.

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$h_\theta(x) = 0.7$

Tell patient that 70% chance of tumor being malignant.

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

Date: / /
Page No.

"probability that $y=1$, given x, parameterized by θ"

Now, the problem is, y can only be either 0 or 1

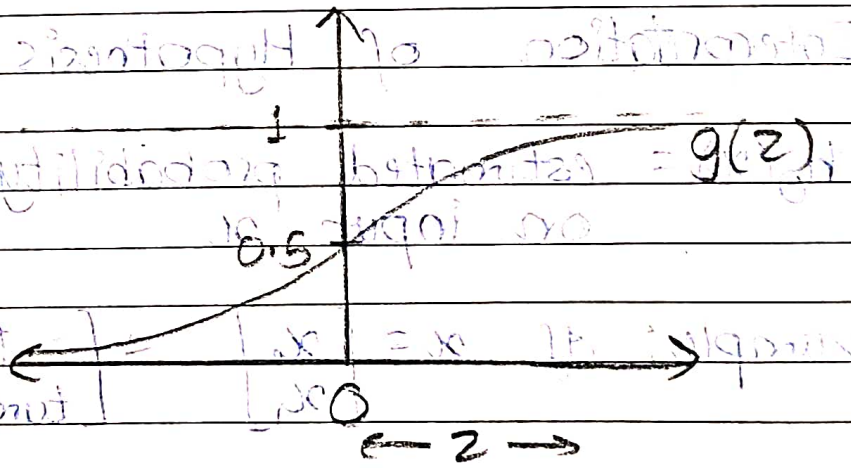
$$P(y=0 \mid x;\theta) = 1 - P(y=1 \mid x;\theta)$$

Decision Boundary

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

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

Suppose, predict "$y=1$" if $h_\theta(x) \geq 0.5$
predict "$y=0$" if $h_\theta(x) < 0.5$



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

$\therefore$ $h_\theta(x) = g(\theta^T x) \geq 0.5$

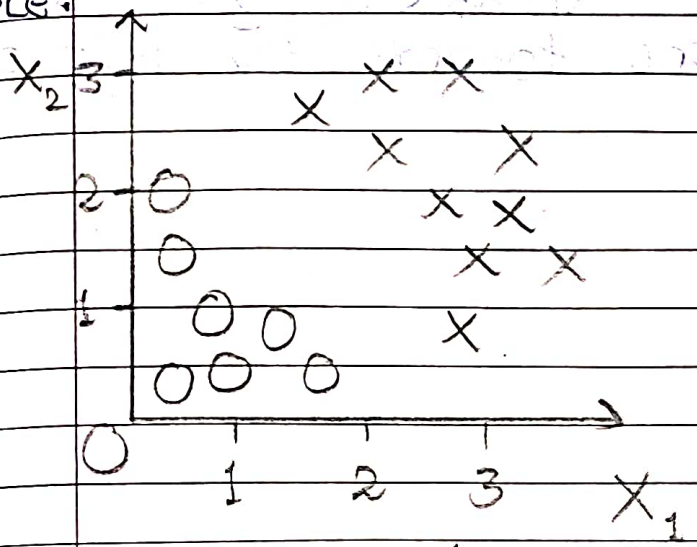
whenever $\theta^T x \geq 0$

Date: / /
Page No.

and $h_\theta(x) = g(\theta^T x) < 0.5$

whenever $\theta^T x < 0$

EXAMPLE:



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

Say we choose $\theta_0 = -3 \quad \theta_1 = 1 \quad \theta_2 = 1$

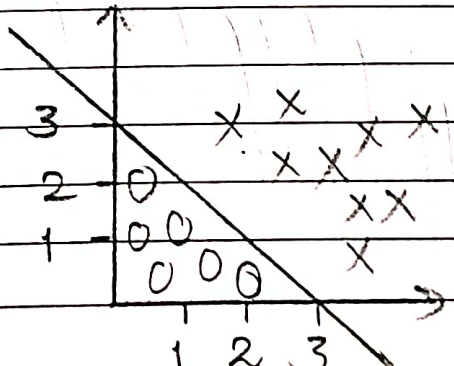
$\Rightarrow \quad \theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

Predict "y = 1" if $\underbrace{-3 + x_1 + x_2}_{(\theta^T x)} \geq 0$

$\Rightarrow \quad x_1 + x_2 \geq 3$

Let's see what that means on the figure

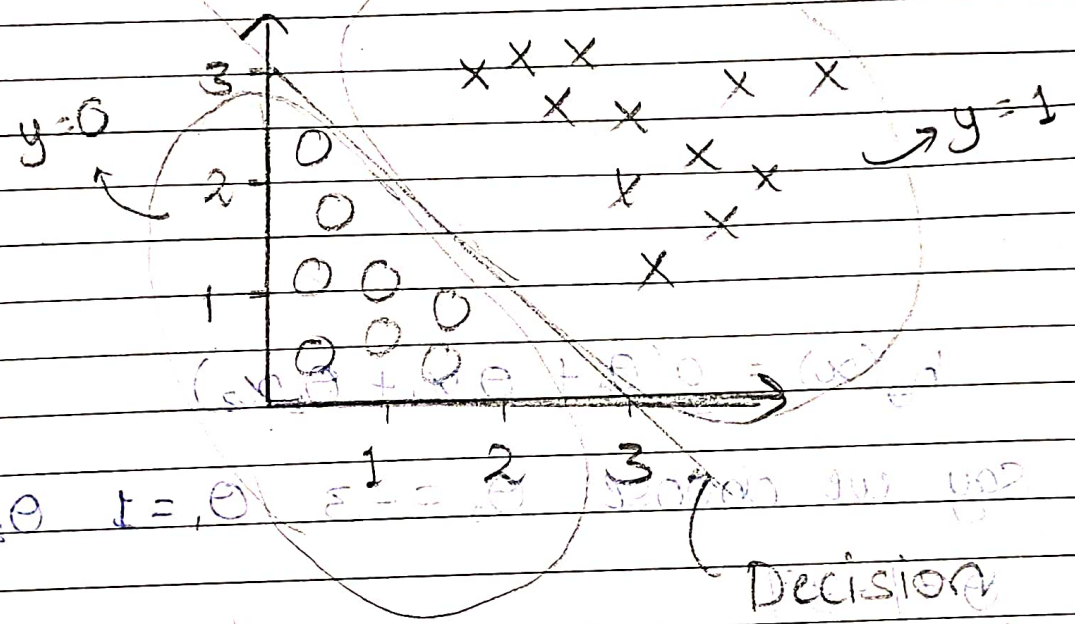
$x_1 + x_2 = 3$



Date: / /
Page No.

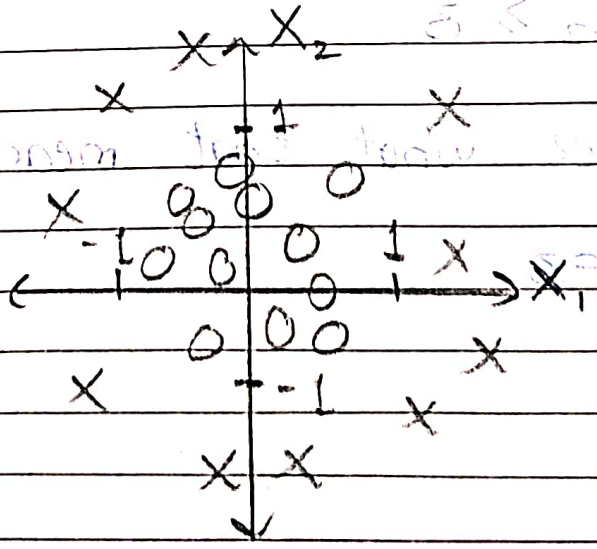
∴ $x_1 + x_2 \geq 3$ means the right half that is, everything to the up and right of the line.

The region where $x_1 + x_2 < 3$, $y = 0$, and the region is the towards the left of the line.



Here, we can also remove the data set.

- Non-Linear Decision Boundaries:



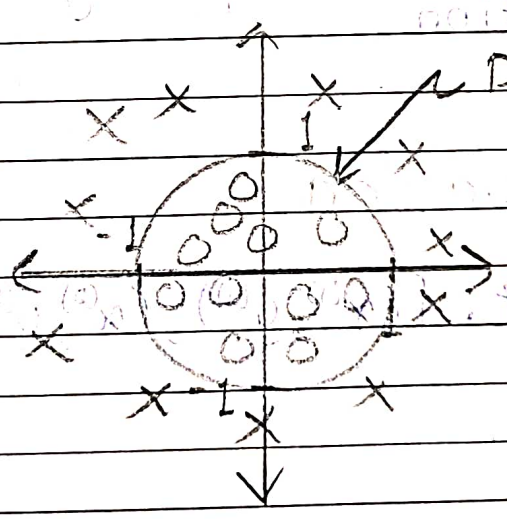
Let's say our hypothesis looks like

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

for now, assume that after some valid procedure to be specified, I end up choosing $\theta_0 = -1,\ \theta_1 = 0,\ \theta_2 = 0,\ \theta_3 = 1,\ \theta_4 = 1$

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Predict "$y = 1$" if $\underbrace{-1 + x_1^2 + x_2^2 \geq 0}_{x_1^2 + x_2^2 \geq 1}$

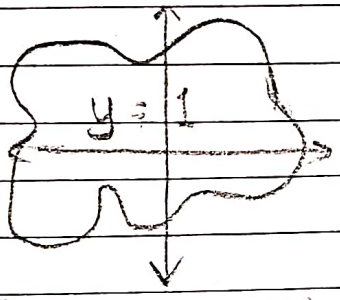


This will mean, everything outside the circle is the "$y = 1$" region, and everything inside is "$y = 0$" region.

Date: / /
Page No.

NOTE: The decision ‡ boundary is a property not of the training set, but the hypothesis under ~~and~~ the parameters. Thus, our vector θ will define the decision boundary.

This is amazing! since we can get almost any type of decision boundary like this.



And this complexity increases proportionally with the complexity of our hypothesis function

- ## Cost Function

Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

m examples $x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$ $x_0 = 1$, $y \in \{0, 1\}$

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

How to choose parameter?

In linear regression,

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

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

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

$Cost\left(h_\theta(x), y\right) = \frac{1}{2}\left(h_\theta(x) - y\right)^2$

This is the cost the learning algorithm will have to pay, if it outputs $h_\theta(x)$ where the correct value was y.

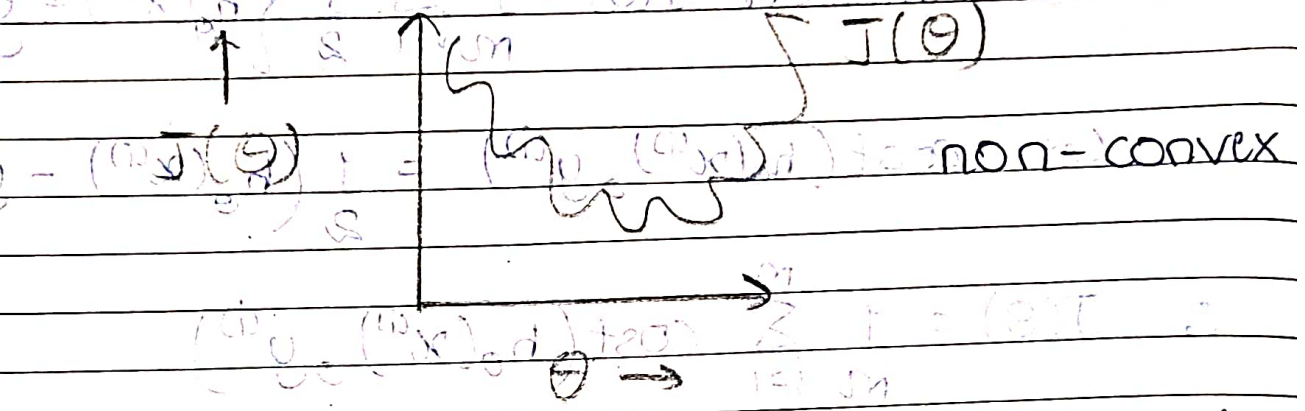
Now, this cost functions works well for linear regression, but here, we're interested in logistic regression.

If we minimize ~~of~~ $cost(h(x), y)$ and then plug it in the $J(\theta)$ then linear regression will work ok. But, it turns out that the $h_\theta(x)$ function has non-linearity which can be easily seen by the fact that

Date: / /
Page No.

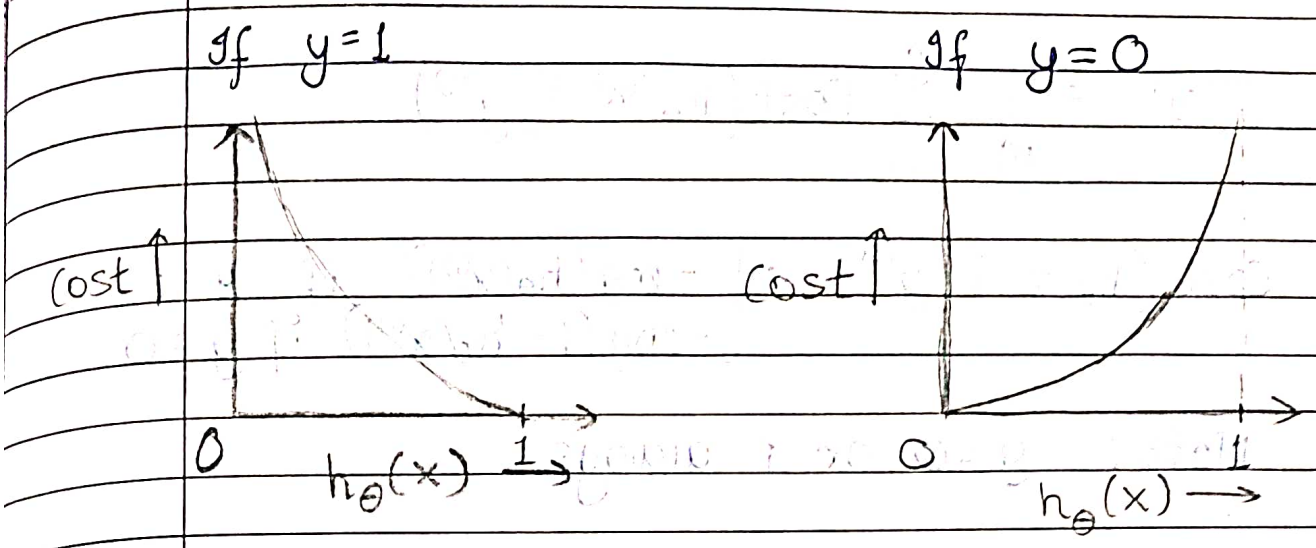
$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

So, the non-linear sigmoid function is plugged in the cost function and then is the $J(\theta)$ function and then plot to see what $J(\theta)$ looks like, we find, it can look like



non-convex and it's easy to see that if we run gradient descent at a given point, it's not gauranted that we will reach the global minima.
But we want a convex $J(\theta)$

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



If $y = 1$, $h(x) = 1 \rightarrow$ cost $= 0$

If $y = 1$, $h(x) = 0 \rightarrow$ cost $\rightarrow \infty$

This means if the algorithm predicts certainly that the output is 0 or the $h_\theta(x) = 0$, but if actually $y = 1$ or the actual output is 1, then we 'penalize' the model by cost $= \infty$ from the graph (very large cost)

Similarly, if $y = 0$ & $h_\theta(x) = 1$, the cost will be extremely large $\rightarrow \infty$ and we will penalize the algo by a very large value

$$\text{Cost}(h_\theta(x), y) = 0 \quad \text{if } h_\theta(x) = y$$
$$\text{Cost}(h_\theta(x), y) \Rightarrow \infty \quad \text{if } \begin{pmatrix} y = 0 \ \& \ h_\theta(x) = 1 \\ y = 1 \ \& \ h_\theta(x) = 0 \end{pmatrix}$$

Date: / /
Page No.

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

$$Cost\left(h_\theta(x^{(i)}), y^{(i)}\right) = \begin{cases} -\log(h_\theta(x^{(i)})) & \text{if } y = 1 \\ -\log(1 - h_\theta(x^{(i)})) & \text{if } y = 0 \end{cases}$$

Note: $y = 0$ or 1 always

$$Cost(h(x), y) = -y \log(h(x)) - (1-y) \log(1 - h(x))$$

If $y = 1$, $Cost(h(x), y) = -\log(h(x))$

If $y = 0$, $Cost(h(x), y) = -\log(1 - h(x))$

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} Cost\left(h(x^{(i)}), y^{(i)}\right)$$

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

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

This is nice as this is convex.

Date: / /
Page No.

To fit parameters θ:

$\min_{\theta} J(\theta)$ ↪ Get θ

To make a prediction given new x:

Output $h_\theta(x) = \frac{1}{1 - e^{-\theta^T x}}$ $\left(P(y=1 \mid x;\theta)\right)$

- GRADIENT DESCENT:

Want $\min_\theta J(\theta)$:

Repeat {

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

(simultaneously update all θ_j)

}

where $J(\theta)$ is the function defined before.

Repeat {

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

(simultaneously update all θ_j)

}

Date: / /
Page No.

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

Vectorized Implementation:

$h = g(X\theta)$ where $X \in \mathbb{R}^{m \times n}$, $\theta \in \mathbb{R}^{n \times 1}$

$$J(\theta) = \frac{1}{m} \cdot \left(-y^T \log(h) - (1-y)^T \log(1-h) \right)$$

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

- Advanced Optimization Algorithm:

Cost function $J(\theta)$. Want $\min_\theta J(\theta)$

Given θ, we have code that can compute
- $J(\theta)$
- $\frac{\partial}{\partial \theta_j} J(\theta)$ (for $j = 0, 1, \ldots, n$)

Gradient Descent:

Repeat {

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

(simultaneous)

}

So, for gradient descend, we need code to supply $J(\theta)$ {for monitoring the correctness of g.d.} and $\frac{\partial}{\partial \theta_j} J(\theta)$

Date: / /
Page No.

and plug in the value in the g.d. function to get the updated values of θ for minimizing $J(\theta)$.

There are many more optimization algos.:

- Conjugate Gradient
- BFGS
- L-BFGS

Advantages:

- No need to manually pick α
- Often faster than g.d.

These algorithms have a clever inter-loop like line search algorithm that automatically that tries out different values of α and then chooses a good learning rate so, it can even pick different values of α for every iteration automatically. They also do much more than just picking a good learning rate α and so, they often end up converging much faster than gradient descent.

Disadvantage:

- Complex

Date: / /
Page No.

Explaination with an Example:

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$$J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2$$

$$\frac{\partial J(\theta)}{\partial \theta_1} = 2(\theta_1 - 5)$$

$$\frac{\partial J(\theta)}{\partial \theta_2} = 2(\theta_2 - 5)$$

Ans: $\theta_1 = 5 \quad \theta_2 = 5$

But how to do it with some optimization algorithm.

→ function [jVal, gradient] = costFunction(theta)

```
jVal = (theta(1) - 5)^2 + (theta(2) - 5)^2;

gradient = zeros(2,1);
gradient(1) = 2*(theta(1) - 5);
gradient(2) = 2*(theta(2) - 5);
```

→ options = optimset('GradObj', 'on', 'MaxIter', 10

```
initialTheta = zeros(2,1);

[optTheta, functionVal, exitFlag] =

    fminunc(@costFunction, initialTheta, options
```

Date: / /
Page No.

For the implementation of this in Octave, the θ vector must be at least a 2-dimensional vector.

$$\theta \in \mathbb{R}^d \quad | \quad d \geq 2$$

$$\text{theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_{n+1} \end{bmatrix}$$

```
function [jVal, gradient] = costFunction (theta)

    jVal = [code to compute J(θ)];
  gradient(1) = [code to compute ∂/∂θ1 J(θ)];
  gradient(2) = [code to compute ∂/∂θ2 J(θ)];
      :
      :
gradient(n+1) = [code to compute ∂/∂θn+1 J(θ)];
```

Date: / /
Page No.

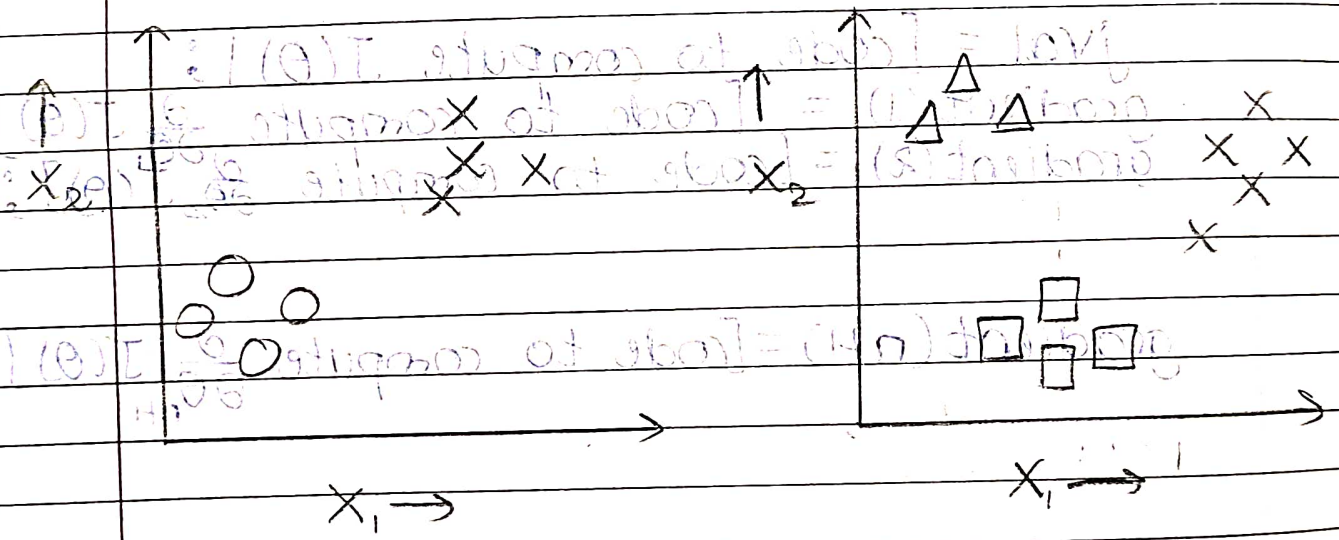
Multiclass Classification

Emails : Work, Friends, Family, Hobby
$y=1$ $y=2$ $y=3$ $y=4$

Medicine : Not ill, Cold, Flu

Weather : Sunny, Cloudy, Rainy, Snowy.

Binary Classification: Multi-Class Classificat

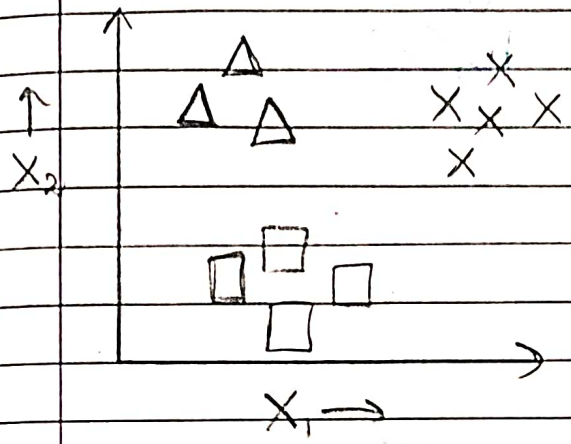


Solution -

One-vs-all (One-vs-rest):

Date: / /
Page No.

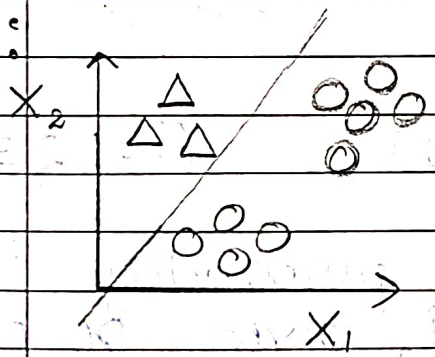
ONE-VS-ALL (ONE-VS-REST):



Class 1: △
Class 2: □
Class 3: ×

We take our training set and turn it into 3 separate two class classification problems.

Class 1:

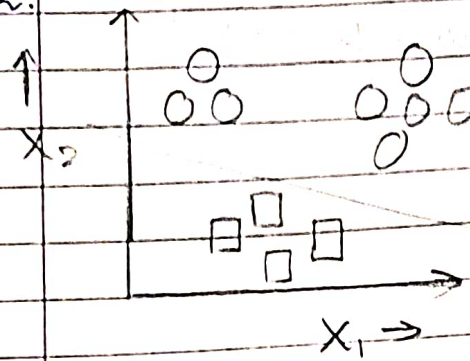


So here this is a sort of fake training set for class 1 where class 1 is the positive class and classes 2 and 3 get assigned to the negative class.

Now we will fit a classifier, say $h_\theta^{(1)}(x)$

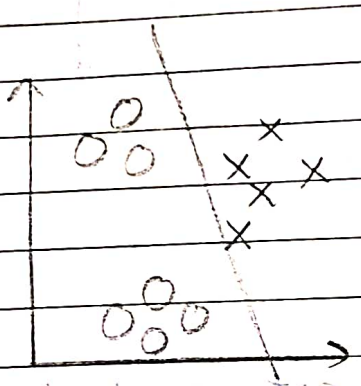
$h_\theta^{(1)}(x)$

Class 2:



$h_\theta^{(2)}(x)$

Class 3:



$h_\theta^{(3)}(x)$

We have fit 3 individual classifiers for the 3 classes. So, for $i = 1, 2, 3$, we have fit a classifier

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

which estimates the probability that y is equal to class i, given x parametrised by θ.

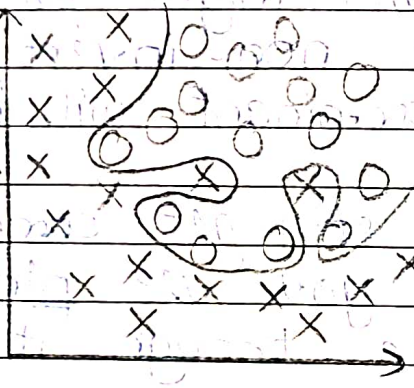
And then, we will pick the value i that maximizes $h_\theta^{(i)}(x)$.

$$\max_i \; h_\theta^{(i)}(x)$$

Regularized Logistic Regression

We have previously discussed that even logistic regression is prone to overfitting by making the decision boundary have a large variance in order to make it fit all the existing training examples, resulting in failiure to generalize the output for new data. This can happen when there are a large number of features and their higher-powered products in the $h(x)$.

Eg.



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^3 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost Function:

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

After regularization, the cost function becomes:

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

Minimizing this cost function will lead to smaller values of θ, causing a simpler hypothesis

Gradient Descent:

Repeat{

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

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

$(j = 1, 2, 3, \ldots, n)$

}

Gradient Descent for Regularized Logistic Regression

Repeat {

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

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

}

Date: / /
Page No.

MATH BEHIND LOGISTIC REGRESSION:

$$\hat{y} = \frac{1}{1 + e^{-(\theta^T X)}} \quad \text{where} \quad \theta \in \mathbb{R}^{(n+1)} \text{ and } X \in \mathbb{R}^{m \times (n+1)}$$

Now, for logistic regression -

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

For simplicity, let's ignore m and let's vectorize this function.

$$J = -\{ \mathbf{y}^T \log(\mathbf{h}) + (1 - \mathbf{y})^T \log(1 - \mathbf{h}) \}$$

where I have darkened the vectors,
$y \in \mathbb{R}^{(m)}$
$h \in \mathbb{R}^{(m)}$

We want to find the derivative of this cost function with respect to θ (b and w)

$$\min_{b,w} J = \frac{\partial J}{\partial \theta}$$

Let's first find $\frac{\partial J}{\partial w}$

Clearly, by chain rule,

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial h} \cdot \frac{\partial h}{\partial w}$$

Date: / /
Page No.

To make this clearer, assume that

① $a = h(=\hat{y})$

② $z = w^T X + b$

So, $a = \sigma(z) \Rightarrow \frac{e^z}{1+e^z} = a \rightarrow e^z = \frac{a}{1-a}$

$\therefore \frac{\partial J}{\partial w} = \frac{\partial J}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w}$

$\Rightarrow \frac{\partial J}{\partial a} = -\left\{ \frac{y^T}{a} - \frac{(1-y)^T}{(1-a)} \right\} = \frac{-y^T + (1-y)^T}{a \quad (1-a)}$

$\frac{\partial a}{\partial z} = \frac{\partial}{\partial z}\left\{ \frac{1}{1+e^{-z}} \right\} = -(1+e^{-z})^{-2} \cdot -e^{-z}$

$\frac{\partial a}{\partial z} = \frac{e^{-z}}{(1+e^{-z})^2} = \frac{(1-a) \times 1}{a \quad (1+a)^2} = a(1-a)$

$\frac{\partial z}{\partial w} = X$

$\frac{\partial J}{\partial w} = \left\{ \frac{-y^T}{a} + \frac{(1-y)^T}{(1-a)} \right\} \{ a(1-a) \} \{ X \}$

$\frac{\partial J}{\partial w} = \left\{ \frac{-(1-a)y^T + (1-y^T)a}{a(1-a)} \right\} \{ a(1-a) \} \{ X \}$

$\frac{\partial J}{\partial w} = (-y^T + y^T a + a - y^T a) X$

$\frac{\partial J}{\partial w} = (A - Y) X \approx \underset{(n \times m)}{X^T} \cdot \underset{(m \times 1)}{(A - Y)}$

$$\frac{\partial J}{\partial b} = \frac{\partial J}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial b}$$

$$\frac{\partial J}{\partial b} = \left\{ \frac{-y}{a} + \frac{(1-y)}{(1-a)} \right\} \{a(1-a)\} \{1\}$$

$$\frac{\partial J}{\partial b} = -y(1-a) + (1-y)a = -y + ya + a - ya$$

$$\boxed{\frac{\partial J}{\partial b} = (A - Y)} \approx np.sum(A - Y)$$

$$w = w - \alpha \cdot \frac{\partial J}{\partial w}$$

$$b = b - \alpha \frac{\partial J}{\partial b}$$