BITS F464: Machine Learning

21 Logistic Regression

Dr. Kamlesh Tiwari
Assistant Professor, Department of CSIS,
BITS Pilani, Pilani Campus, Rajasthan-333031 INDIA

March 12, 2021 ONLINE (Campus @ BITS-Pilani Jan-May 2021)

http://ktiwari.in/ml

Recap: Linear Regression

Regression predicts value of continuous a target variable

x_1	x_2	x_3	y
10	50	20	10
11	31	22	12
11	12	15	4
20	55	20	22
23	41	27	1
31	12	35	9
13	18	12	23
21	55	16	16
32	56	27	22
8	22	35	??

What is at ??

- Linear model for regression uses a linear combination of the input variables

$$y(x,w) = w_0 + w_1x_1 + ... + w_nx_n$$

here x is a n dimensional vector $(x_1, x_2, ..., x_n)$

- Suitable w, makes the value of $y(x^{(i)}, w)$ very close to $y^{(i)}$

Recap: Cost/Error Function

- Finding w is similar to solving a minimization problem on a **squared error cost function** such as

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

where m is number of training examples.

x_1	x_2	x_3	y	$\hat{y}$	$(\hat{y}-y)^2$
10	50	20	10	8	4
11	31	22	12	9	9
11	12	15	4	3	1
20	55	20	22	26	16
23	41	27	1	1	0
31	12	35	9	4	25
13	18	12	23	30	49
21	55	16	16	13	9
32	56	27	22	21	1

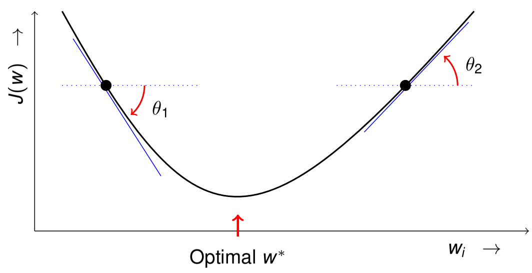
For some w, let us compute $\hat{y} = y(x^{(i)}, w)$ then

$$J(w) = \frac{1}{2\times 9} \times 114 = 6.33$$

One have to minimize the value of $J(w)$ using suitable w

$$\underset{w}{\text{argmin}}\, J(w)$$

Recap: Consider $w_i = w_i - \alpha\frac{\partial}{\partial w_i}J(w)$



- Slope $\tan\theta_1$, representing $\frac{\partial}{\partial w_i}J(w)$ is $-ve$ so the equation $w_i = w_i - \alpha\frac{\partial}{\partial w_i}J(w)$ moves w_i towards w^*
- $\tan\theta_2$, being $+ve$ the equation still moves w_i towards w^*

Recap: Gradient Descent

Algorithm 1: Gradient Descent

1. Initialize w randomly
2. **repeat**
3. Simultaneously update all w_j with $w_j - \alpha\frac{\partial}{\partial w_j}J(w)$
4. **until** *converge*;
5. **return** w

- Here α is a learning rate. If α is small enough then $J(w)$ would decrease in every iteration
- Large α may overshoot the minimum and could fail to converge

Similar Mechanism for Classification

Classification have predefined fixed number of labels

x_1	x_2	x_3	*Class*
10	50	20	1
11	31	22	1
11	12	15	0
20	55	20	0
23	41	27	0
31	12	35	1
13	18	12	0
21	55	16	1
32	56	27	0
8	22	35	??

What is at ??

- Moving from linear regression $y(x,w) = w_0 + w_1x_1 + ... + w_nx_n$ to **logistic regression**

$$y(x,w) = \sigma(w_0 + w_1x_1 + ... + w_nx_n)$$

- where σ is called as sigmoid function defined as

$$\sigma(v) = \frac{1}{1+e^{-v}}$$

$\sigma : (-\infty, \infty) \rightarrow (0, 1)$

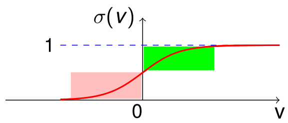
Why sigmoid? it has nice derivative $\sigma'(x) = \sigma(x)(1-\sigma(x))$

Logistic Regression

$$y(x, w) = \sigma(w_0 + w_1 x_1 + ... + w_n x_n)$$

- Enables "classification" apart from the regression.
- Sigmoid produces values in range 0 to 1 and is defined as

$$\sigma(v) = \frac{1}{1+e^{-v}}$$



$\sigma(+ve) > 0.5$
$\sigma(-ve) < 0.5$

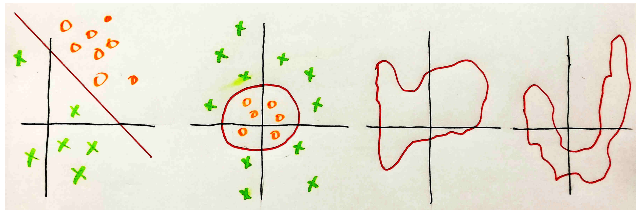
Decision on classification

$$classification = \begin{cases} 1 & \text{if } y(x, w) \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Decision Boundary in Logistic Regression

$$classification = \begin{cases} 1 & \text{if } y(x, w) \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

- Choice of w partitions the space into two sections
- Hyper-plane separating them is called **decision boundary**
- By adding more complex or polynomial terms one can get more complex decision boundary



Liner Regression Cost Function becomes non convex

Liner regression cost function $J(w) = \frac{1}{2m}\sum_{i=1}^{m}(y(x^{(i)}, w) - y^{(i)})^2$

- For logistic regression it is taken as [1]

$$J(w) = \frac{1}{2m}\sum_{i=1}^{m}(\sigma(v) - y^{(i)})^2$$

$$\begin{aligned} \frac{\partial}{\partial w_j}J(w) &= \frac{\partial}{\partial w_j}\frac{1}{2m}\sum_{i=1}^{m}(\sigma(v) - y^{(i)})^2 = \frac{1}{2m}\sum_{i=1}^{m}\frac{\partial}{\partial w_j}(\sigma(v) - y^{(i)})^2 \\ &= \frac{1}{m}\sum_{i=1}^{m}(\sigma(v) - y^{(i)})\frac{\partial}{\partial w_j}(\sigma(v) - y^{(i)}) = \frac{1}{m}\sum_{i=1}^{m}(\sigma(v) - y^{(i)})(\frac{\partial}{\partial w_j}\sigma(v) - 0) \\ &= \frac{1}{m}\sum_{i=1}^{m}(\sigma(v) - y^{(i)})\sigma(v)(1-\sigma(v))\frac{\partial}{\partial w_j}(-v) = -\frac{1}{m}\sum_{i=1}^{m}(\sigma(v) - y^{(i)})\sigma(v)(1-\sigma(v))x_j \end{aligned}$$

The derivative is not a monotonically increasing function
Therefore, $J(w)$ with sigmoid is non convex

[1] let $v = y(x^{(i)}, w)$

Cross Entropy as a Cost Function

- Cost function used for the liner regression

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

becomes a non convex function in case of logistic regression

- Therefore, a different cost function (cross entropy) is chosen

$$J(w) = \frac{1}{m}\sum_{i=1}^{m}Cost(y(x^{(i)}, w), y^{(i)})$$

where

$$Cost(y(x^{(i)}, w), y^{(i)}) = \begin{cases} -\log(y(x^{(i)}, w)) & \text{if } y^{(i)} = 1 \\ -\log(1 - y(x^{(i)}, w)) & \text{otherwise} \end{cases}$$

A simplified version of this cost function is

$$Cost(y(x^{(i)}, w), y^{(i)}) = -y^{(i)}\log(y(x^{(i)}, w)) - (1 - y^{(i)})\log(1 - y(x^{(i)}, w))$$

Convexity for Cross Entropy

Consider
$f_1(u) = -\log\sigma(u) = -\log\frac{1}{1+e^{-u}}$

$$\begin{aligned} \frac{d}{du}f_1(u) &= \frac{d}{du} - \log\frac{1}{1+e^{-u}} \\ &= \frac{d}{du}\log(1+e^{-u}) \\ &= \frac{-e^{-u}}{(1+e^{-u})} \\ &= -1 + \sigma(u) \end{aligned}$$

Derivative of $f_1(u)$ is a monotonically increasing therefore, $f_1(u)$ is convex

Consider $f_2(u) = -\log(1 - \sigma(u))$

$$\begin{aligned} f_2(u) &= -\log(1 - \frac{1}{1+e^{-u}}) \\ &= -\log(\frac{e^{-u}}{1+e^{-u}}) \\ &= -\log(e^{-u}) - \log\left(\frac{1}{1+e^{-u}}\right) \\ &= u + f_1(u) \\ \frac{d}{du}f_2(u) &= 1 + (-1 + \sigma(u)) = \sigma(u) \end{aligned}$$

Derivative of $f_2(u)$ is also a monotonically increasing therefore, $f_2(u)$ is also convex

Linear combination of $f_1(u)$ and $f_2(u)$ would also be a convex function

Learning With This Cost Function

- Learning corresponds to the minimization of $J(w)$ by changing w

$$\underset{w}{\operatorname{argmin}}\, J(w) = \frac{1}{m}\sum_{i=1}^{m}Cost(y(x^{(i)}, w), y^{(i)})$$

$$\underset{w}{\operatorname{argmin}}\, J(w) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log(y(x^{(i)}, w)) - (1 - y^{(i)})\log(1 - y(x^{(i)}, w))]$$

- *Gradient descent* could be used for optimization

Algorithm 2: Logistic Regression

```
Initialize w randomly
repeat
    Simultaneously update all w_j with w_j − α ∂/∂w_j J(w)
until converge;
return w
```

The Partial Derivative Term

Recall differentiation

$\frac{d}{dx}x^{-1} = \frac{-1}{x^2}$ $\quad$ $\frac{d}{dx}\log x = \frac{1}{x}$ $\quad$ $\frac{d}{dx}\log \sin x = \frac{1}{\sin x}\frac{d}{dx}\sin x = \frac{1}{\sin x}\cos x$

Let $v = w_0x_0 + w_1x_1 + ... + w_nx_n$ Then

$$\frac{\partial}{\partial w_j}v = \frac{\partial}{\partial w_j}(w_0x_0 + w_1x_1 + \ldots + w_nx_n) = x_j$$

$$J(w) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log(y(x^{(i)},w)) - (1-y^{(i)})\log(1-y(x^{(i)},w))]$$

$$\frac{\partial}{\partial w_j}J(w) = \frac{1}{m}\sum_{i=1}^{m}[-\frac{\partial}{\partial w_j}y^{(i)}\log(y(x^{(i)},w)) - \frac{\partial}{\partial w_j}(1-y^{(i)})\log(1-y(x^{(i)},w))]$$

$$= \frac{1}{m}\sum_{i=1}^{m}[-A-B] \quad (1)$$

The Partial Derivative Term

$$A = \frac{\partial}{\partial w_j}y^{(i)}\log(y(x^{(i)},w))$$

$$= y^{(i)} \times \frac{\partial}{\partial w_j}\log(y(x^{(i)},w))$$

$$= y^{(i)} \times \frac{1}{y(x^{(i)},w)} \times \frac{\partial}{\partial w_j}y(x^{(i)},w)$$

$$= y^{(i)} \times \frac{1}{\frac{1}{1+e^{-v}}} \times \frac{\partial}{\partial w_j}\frac{1}{1+e^{-v}}$$

$$= y^{(i)} \times (1+e^{-v}) \times \frac{-1}{(1+e^{-v})^2} \times \frac{\partial}{\partial w_j}(1+e^{-v})$$

$$= \frac{-y^{(i)}}{1+e^{-v}} \times (0+e^{-v} \times \frac{\partial}{\partial w_j}(-v))$$

$$= y^{(i)} \times \frac{e^{-v}}{1+e^{-v}} \times x_j \quad (2)$$

The Partial Derivative Term

$$B = \frac{\partial}{\partial w_j}(1-y^{(i)})\log(1-y(x^{(i)},w))$$

$$= (1-y^{(i)}) \times \frac{1}{1-y(x^{(i)},w)} \times \frac{\partial}{\partial w_j}(1-y(x^{(i)},w))$$

$$= (1-y^{(i)}) \times \frac{-1}{1-\frac{1}{1+e^{-v}}} \times \frac{\partial}{\partial w_j}y(x^{(i)},w)$$

$$= (1-y^{(i)}) \times \frac{(-1)(1+e^{-v})}{e^{-v}} \times \frac{\partial}{\partial w_j}\frac{1}{1+e^{-v}}$$

$$= (1-y^{(i)}) \times \frac{(-1)(1+e^{-v})}{e^{-v}} \times \frac{-1}{(1+e^{-v})^2} \times \frac{\partial}{\partial w_j}(1+e^{-v})$$

$$= (1-y^{(i)}) \times \frac{(-1)(1+e^{-v})}{e^{-v}} \times \frac{-1}{(1+e^{-v})^2} \times (0+e^{-v}\frac{\partial}{\partial w_j}(-v))$$

$$= (1-y^{(i)}) \times \frac{(-1)(1+e^{-v})}{e^{-v}} \times \frac{e^{-v}}{(1+e^{-v})^2} \times \frac{\partial}{\partial w_j}v$$

$$= (1-y^{(i)}) \times \frac{-1}{1+e^{-v}} \times x_j \quad (3)$$

The Partial Derivative Term

$$\frac{\partial}{\partial w_j}J(w) = \frac{1}{m}\sum_{i=1}^{m}[-A-B]$$

$$= \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)} \times \frac{e^{-v}}{1+e^{-v}} \times x_j - (1-y^{(i)}) \times \frac{-1}{1+e^{-v}} \times x_j]$$

$$= \frac{1}{m}\sum_{i=1}^{m}[(1-y^{(i)}) - y^{(i)} \times e^{-v}] \times \frac{x_j}{1+e^{-v}}$$

$$= \frac{1}{m}\sum_{i=1}^{m}[1-y^{(i)} \times (1+e^{-v})] \times \frac{x_j}{1+e^{-v}}$$

$$= \frac{1}{m}\sum_{i=1}^{m}[\frac{1}{1+e^{-v}} - y^{(i)}] \times x_j$$

$$= \frac{1}{m}\sum_{i=1}^{m}[y(x^{(i)},w) - y^{(i)}] \times x_j \quad (4)$$

The Partial Derivative Term

Partial derivative term of $J(w)$

$$\frac{\partial}{\partial w_j}J(w) = \frac{\partial}{\partial w_j}\frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log(y(x^{(i)},w)) - (1-y^{(i)})\log(1-y(x^{(i)},w))]$$

we have seen, it comes out to be

$$\frac{\partial}{\partial w_j}J(w) = \frac{1}{m}\sum_{i=1}^{m}(y(x^{(i)},w) - y^{(i)})x_j^{(i)}$$

Algorithm 3: Logistic Regression

1. Initialize w randomly
2. **repeat**
3. Simultaneously update all w_j with $w_j - \alpha \times \frac{1}{m}\sum_{i=1}^{m}(y(x^{(i)},w) - y^{(i)})x_j^{(i)}$
4. **until** *converge*;
5. **return** w

It looks identical to liner regression but, $y(x^{(i)},w)$ is different $\frac{1}{1+e^{-(w_0+w_1x_1^{(i)}+\ldots+w_nx_n^{(i)})}}$

Example: Logistic Regression

Consider following data

	x_1	x_2	x_3	*Class*
1	2	2	2	1
2	3	2	2	1
3	2	3	2	1
4	2	2	3	1
5	7	6	9	0
6	9	7	6	0
7	9	6	7	0
8	6	8	9	0
9	8	9	6	0
10	8	9	9	0

Learning rate $\alpha = 0.01$

J(w)	$w = (w_0\, w_1\, w_2\, w_3)$
6.912	(0.500 0.500 0.500 0.500)
6.496	(0.494 0.453 0.455 0.454)
5.944	(0.488 0.406 0.410 0.408)
5.316	(0.482 0.360 0.366 0.363)
4.692	(0.477 0.313 0.321 0.317)
4.072	(0.471 0.267 0.277 0.272)
3.460	(0.465 0.221 0.233 0.227)
2.860	(0.460 0.175 0.189 0.182)
2.279	(0.454 0.130 0.146 0.138)
1.735	(0.449 0.086 0.104 0.095)
1.262	(0.445 0.044 0.064 0.054)
0.906	(0.441 0.008 0.029 0.018)
0.685	(0.438 -0.022 0.000 -0.011)
0.566	(0.437 -0.044 -0.020 -0.032)
0.504	(0.436 -0.060 -0.035 -0.048)
0.470	(0.436 -0.072 -0.046 -0.059)
0.451	(0.436 -0.081 -0.055 -0.068)
0.438	(0.436 -0.088 -0.061 -0.074)
0.431	(0.437 -0.093 -0.066 -0.080)
0.425	(0.438 -0.098 -0.070 -0.084)
0.422	(0.439 -0.101 -0.074 -0.088)
0.419	(0.440 -0.105 -0.077 -0.091)
0.417	(0.441 -0.107 -0.079 -0.093)
0.416	(0.443 -0.110 -0.081 -0.095)
0.415	(0.444 -0.112 -0.082 -0.097) **Iteration 25**
0.412	(0.451 -0.119 -0.088 -0.103) **Iteration 30**
0.348	(0.857 -0.179 -0.084 -0.132) **Iteration 300**
0.116	(3.256 -0.409 -0.135 -0.291) **Iteration 3000**
0.012	(7.596 -0.748 -0.361 -0.588) **Iteration 30000**
0.001	(11.975 -1.091 -0.599 -0.896) **Iteration 300000**

Example: Find J(w)

Let (w_0, w_1, w_2, w_3)=(0.5, 0.5, 0.5, 0.5),
By definition $v = w_0 + w_1x_1 + w_2x_2 + w_3x_3$, and $y(x^{(i)}, w) = \sigma(v)$ then
$cost = -y^{(i)}\log(y(x^{(i)}, w)) - (1 - y^{(i)})\log(1 - y(x^{(i)}, w))$

i	x_1	x_2	x_3	$y^{(i)}$	v	$y(x^{(i)}, w)$	*cost*
1	2	2	2	1	3.5	0.970	0.029
2	3	2	2	1	4.0	0.982	0.018
3	2	3	2	1	4.0	0.982	0.018
4	2	2	3	1	4.0	0.982	0.018
5	7	6	9	0	11.5	0.999	11.49
6	9	7	6	0	11.5	0.999	11.49
7	9	6	7	0	11.5	0.999	11.49
8	6	8	9	0	12	0.999	11.51
9	8	9	6	0	12	0.999	11.51
10	8	9	9	0	13	0.999	11.51
						Total/10:	6.9118

Example: Find next W

Let (w_0, w_1, w_2, w_3)=(0.5, 0.5, 0.5, 0.5) and $t_i = (y(x^{(i)}, w) - y^{(i)})x_j^{(i)}$
Then $\frac{1}{m}\sum_{i=1}^{m}(y(x^{(i)}, w) - y^{(i)})x_j^{(i)} = \frac{1}{m}\sum_{i=1}^{m} t_i$ let $\hat{y}^{(i)} = y(x^{(i)}, w)$
Then update w_j with $w_j - \alpha \times \frac{1}{m}\sum_{i=1}^{m} t_i$ we have set $\alpha = 0.01$

i	x_0	x_1	x_2	x_3	$y^{(i)}$	$\hat{y}^{(i)}$	t_0	t_1	t_2	t_3
1	1	2	2	2	1	0.970	-0.029	-0.058	-0.058	-0.058
2	1	3	2	2	1	0.982	-0.017	-0.053	-0.035	-0.035
3	1	2	3	2	1	0.982	-0.017	-0.035	-0.053	-0.035
4	1	2	2	3	1	0.982	-0.017	-0.035	-0.035	-0.053
5	1	7	6	9	0	0.999	0.999	6.999	5.999	8.999
6	1	9	7	6	0	0.999	0.999	8.999	6.999	5.999
7	1	9	6	7	0	0.999	0.999	8.999	5.999	6.999
8	1	6	8	9	0	0.999	0.999	5.999	7.999	8.999
9	1	8	9	6	0	0.999	0.999	7.999	8.999	5.999
10	1	8	9	9	0	0.999	0.999	7.999	8.999	8.999
						total	5.916	46.815	44.815	45.815
						$w_i - \alpha \times (total/m)$	0.494	0.453	0.455	0.454

Example: Classification across Iterations

Following table shows classification as the weights get modified along 1^{st}, 100^{th}, 300^{th} and 500^{th} iteration

i	x_1	x_2	x_3	$y^{(i)}$	1	100	300	500
1	2	2	2	1	1	0	1	1
2	3	2	2	1	1	0	0	1
3	2	3	2	1	1	0	1	1
4	2	2	3	1	1	0	1	1
5	7	6	9	0	1	0	0	0
6	9	7	6	0	1	0	0	0
7	9	6	7	0	1	0	0	0
8	6	8	9	0	1	0	0	0
9	8	9	6	0	1	0	0	0
10	8	9	9	0	1	0	0	0

$J(w) > 0$ even if with perfect classification, and the iteration continues

Thank You!

Thank you very much for your attention! (Reference[2])

Queries ?

[2] [1] Book - *Pattern Recognition And Machine Learning*, Bishop, Springer-2006 (CH-3), [2] Book - *Machine Learning*, ch-6, Tom M. Mitchell.