Andrew Ng Course Note 5

Shu Wang

May 19 2023

1 Classification

Question	Answer "y"
Is this email spam?	No, yes
Is the transaction fraudulent	No,yes
Is the tumor malignant?	No,yes

y can only be one of two values

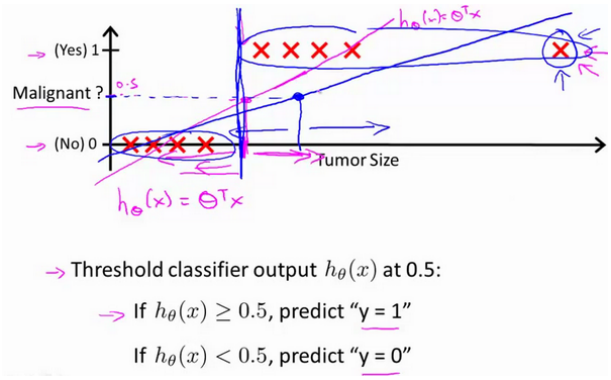
"binary classification"

class/ category

false : 0; true : 1

False/zero class is called negative class; True/one class is called positive class.

negative/positive doesn't correspond to good/bad , they correspond to the absence/presence of a characteristic that you're looking for.



Pick a threshold of 0.5, if the model output ≤ 0.5, then predict $y = 0$.

If $f_{w,b}(x) < 0.5 \longrightarrow \hat{y} = 0$,

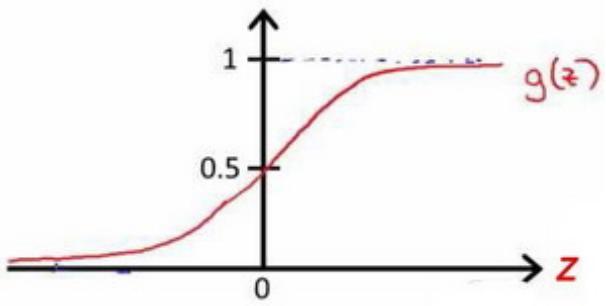
$f_{w,b}(x) \geq 0.5 \longrightarrow \hat{y} = 1$

With the top right training example added, it is clear that linear regressio with a threshold of 0.5 will not make sense.

1.1 Logistic Regression

Fit a S-shaped curve into the dataset.

Sigmoid function:



x-axis has negative values and positive values.

Outputs between 0 and 1.

$g(z) = \frac{1}{1+e^{-z}}$, $0 < g(z) < 1$

As z approaches ∞, $g(z)$ approaches 1.

$f_{\vec{w},b}(x) = \vec{w} \cdot \vec{x} + b$

$z = \vec{w} \cdot \vec{x} + b$

Then input this z into the sigmoid function, $g(z) = \frac{1}{1+e^{-z}}$

$f_{\vec{w},b}(x) = g(\vec{w} \cdot \vec{x} + b) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x} + b)}}$

This is the logistic regression model.

Output: The probability that the class label is 1.

Example: $f_{\vec{w},b}(x) = 0.7$ means 70% that the tumor is malignant.

$P(y = 0) + P(y = 1) = 1$

$f_{\vec{w},b}(x) = P(y = 1|\vec{x}; \vec{w}, b)$

1.2 Decision Boundary

$f_{\vec{w},b}(x) = g(\vec{w} \cdot \vec{x} + b) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x} + b)}}$

$= P(y = 1|\vec{x}; \vec{w}, b)$

A common choice for threshold is 0.5.

Is $f_{\vec{w},b}(x) \geq 0.5$?

Yes: $\hat{y} = 1$

When is $f_{\vec{w},b}(x) \geq 0.5$?

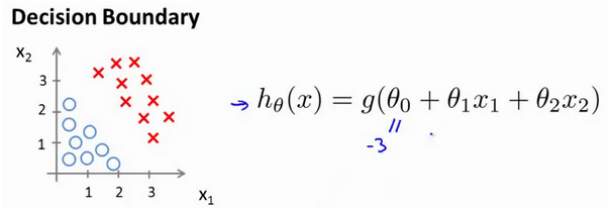
$g(z) \geq 0.5$;

$z \geq 0$;

$\vec{w} \cdot \vec{x} + b \geq 0$

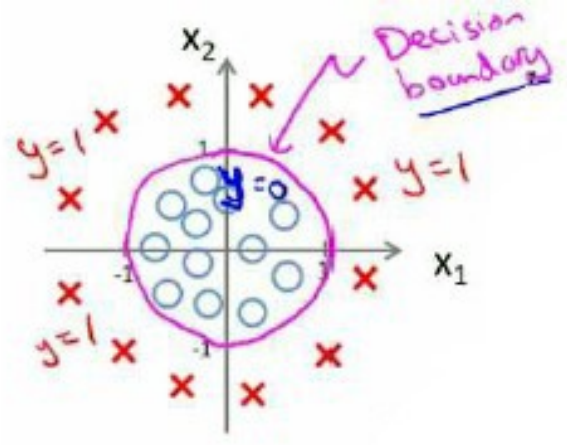
$\hat{y} = 1$

Decision Boundary: $z = \vec{w} \cdot \vec{x} + b = 0$



$x_1 + x_2 = 3$

Non-linear decision boundaries



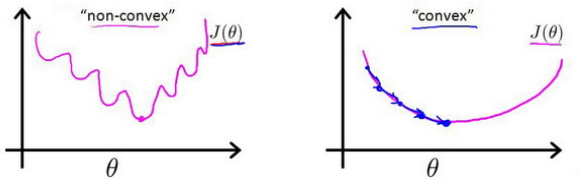
$f_{\vec{w},b}(\vec{x}) = g(z) = g(w_1 {x_1}^2 + w_2 {x_2}^2 + b)$

Decision boundary: $z = {x_1}^2 + {x_2}^2 - 1 = 0;$

${x_1}^2 + {x_2}^2 = 1$

2 Cost Function

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



Logistic regression will get stuck when using gradient descent method on the cost function because the graph is non-convex.

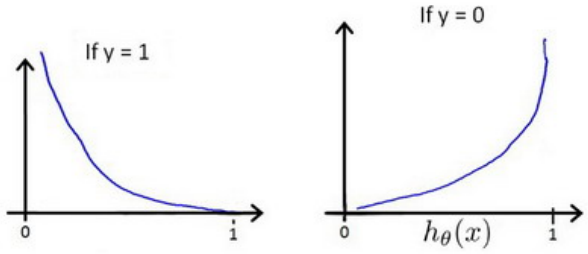
Let L denote the loss on a single training example.

$L(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)})$.

Logistic loss function:

$\mathrm{L}(\mathrm{f}_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) =$

$$\begin{cases} \text{-log}(\text{f}_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) & \text{if y}^{(i)} = 1 \\ \text{-log}(1 - \text{f}_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) & \text{if y}^{(i)} = 0 \end{cases} \tag{1}$$



For left part:

As $f_{\vec{w},b}(\vec{x}^{(i)}) \longrightarrow 1$, then loss $\longrightarrow 0$

As $f_{\vec{w},b}(\vec{x}^{(i)}) \longrightarrow 0$, then loss $\longrightarrow 1$

For right part:

As $f_{\vec{w},b}(\vec{x}^{(i)}) \longrightarrow 1$, then loss $\longrightarrow 1$

As $f_{\vec{w},b}(\vec{x}^{(i)}) \longrightarrow 0$, then loss $\longrightarrow 0$

The further prediction $f_{\vec{w},b}(\vec{x}^{(i)})$ is from target $y^{(i)}$, the higher the loss.

The cost function will be convex, thus we can reach a global minimum.

Cost:

$J(\vec{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L(f_{w,b}(x^{(i)}), y^{(i)})$, L as defined above.

2.1 Simplified lost function

$L(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) = -y^{(i)} log(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) - (1 - y^{(i)}) log(1 - f_{\vec{w},b}(\vec{x}^{(i)}))$.

if $y^{(i)} = 1$: $L(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) = -1 * log(f(\vec{x}))$

if $y^{(i)} = 0$: $L(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) = -1 * log(1 - f(\vec{x}))$

Cost:

$J(\vec{w}, b) = \frac{-1}{m} \sum_{i=1}^{m} y^{(i)} log(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) + (1 - y^{(i)}) log(1 - f_{\vec{w},b}(\vec{x}^{(i)}))$

MLE in statistics.

2.2 Gradient Descent Implementation

Training logistic regresson:

Find $\vec{w}, b$

Given new $\vec{x}$ output $f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-(\vec{w} \cdot \vec{x} + b)}}$

$P(y = 1 | \vec{x}; \vec{w}, b)$

$w_j = w_j - \alpha * \frac{\partial}{\partial w_j} J(\overrightarrow{w}, b)$

$b = b - \alpha * \frac{\partial}{\partial b} J(\overrightarrow{w}, b)$

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

$\frac{\partial}{\partial b} J(\overrightarrow{w}, b) = \frac{1}{m} \sum_{i=1}^{m} (f_{\overrightarrow{w},b}(x^{(i)}) - y^{(i)})$.

update simultaneously

Which is the same as the linear regression.

In linear regression: $f_{\overrightarrow{w},b}(\overrightarrow{x}) = \overrightarrow{w} \cdot \overrightarrow{x} + b$

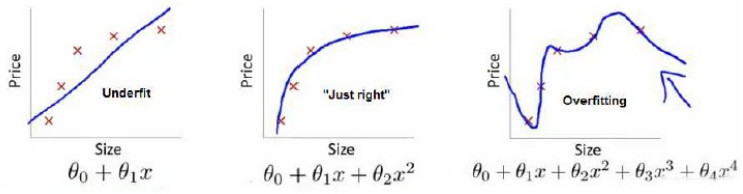
In logistic regression: $f_{\overrightarrow{w},b}(\overrightarrow{x}) = \frac{1}{1+e^{-(\overrightarrow{w} \cdot \overrightarrow{x} + b)}}$

Same concepts: monitor gradient descent (learning curve)

Vectorized implementation

Feature scaling

3 Overfitting



The left graph:

Does not fit the training set well, has high bias.

There's clear pattern in training data that the algorithm is unable to capture.

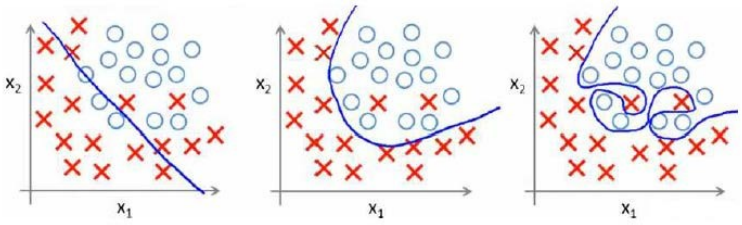
The middle graph:

Fits the training set better, generalization.

The right graph:

Fit a curve that passes through all training examples, seems to do extremely well in training set. Overfitting, has high variance.

Find a model that is neither underfitting nor overfitting.



The degree of polynomial increases, then if fits well, but the ablity of making predictions decrease.

3.1 Addressing Overfitting

Collect more training examples

Select features to include/exclude

if you have too many features with insufficient data, it may lead to overfitting.

Disadvantage: useful features could be lost

Regularization: Reduce size of parameters

Eliminate the feature would set the coefficient = 0.

Shrink the values of parameters without necessarily setting them to 0.

3.2 Cost function with regularization

Suppose you had a way to make parameters w_3, w_4 really small.

$\underset{\vec{w},b}{\text{minimize}} \frac{1}{2m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)})^2 + 1000w_3^2 + 1000w_4^2$

We penalize the model if w_3, w_4 are large.

So if you minimize above function, you end up with w_3, w_4 nearly 0 and get rid of these terms.

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

We penalize all features w_j parameters.

$J(\vec{w}, b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{i=1}^{m} w_j^2$

λ is also called regularization parameter. By convention, $\lambda \geq 0$.

Divided by $2m$ means that both the first and second term are scaled in the same way.bEven if training set growth, the same number of λ we choose is also likely to work.

We are not going to penalize b.

$\underset{\vec{w},b}{\text{minimize}}\, J(\vec{w}, b) = \underset{\vec{w},b}{\text{minimize}} \frac{1}{2m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{i=1}^{m} w_j^2$

mean squared error cost + regularization term.

λ balances both goals to fit the data and keep w_j small.

if $\lambda = 0$, then you end uo fitting overly complex curve.

If $\lambda = 10^{10}$, then the learning algorithm would choose w_i to be 0, the learning algorithm fits the horizontal straight line and underfits.

3.3 Regularized linear regression

$J(\vec{w}, b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{i=1}^{m} w_j^2$

$w_j = w_j - \alpha * \frac{\partial}{\partial w_j} J(\vec{w}, b)$

$b = b - \alpha * \frac{\partial}{\partial b} J(\vec{w}, b)$

Previously:

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

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

Now:

$\frac{\partial}{\partial w_j} J(\vec{w}, b) = \frac{1}{m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)}) * x_j^{(i)} + \frac{\lambda}{m} w_j$.

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

Equation related to b doesn't change.

update simultaneously

Rewrite:

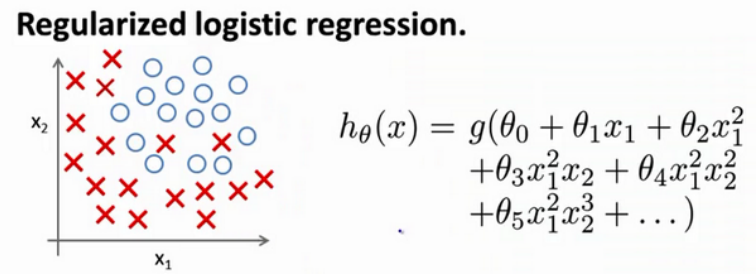
$w_j = w_j - \alpha\frac{\lambda}{m} - \alpha\frac{1}{m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)}) * x_j^{(i)}$

$w_j = (1 - \alpha\frac{\lambda}{m})w_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)}) * x_j^{(i)}$

if $\lambda = 1$, then $\alpha\frac{\lambda}{m}$ is a small positive number.

shrinking w_j just a little.

3.4 Regularized logistic regression



$J(\vec{w}, b) = \frac{-1}{m}\sum_{i=1}^{m} y^{(i)} log(f_{\vec{w},b}(\vec{x}^{(i)}, y^{(i)}) + (1 - y^{(i)}) log(1 - f_{\vec{w},b}(\vec{x}^{(i)})) + \frac{\lambda}{2m}\sum_{j=1}^{m} w_j^2$

$\frac{\partial}{\partial w_j}J(\vec{w}, b) = \frac{1}{m}\sum_{i=1}^{m}(f_{\vec{w},b}(x^{(i)}) - y^{(i)}) * x_j^{(i)} + \frac{\lambda}{m}w_j.$

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

The exact same equation as linear regression.