Copyright Notice

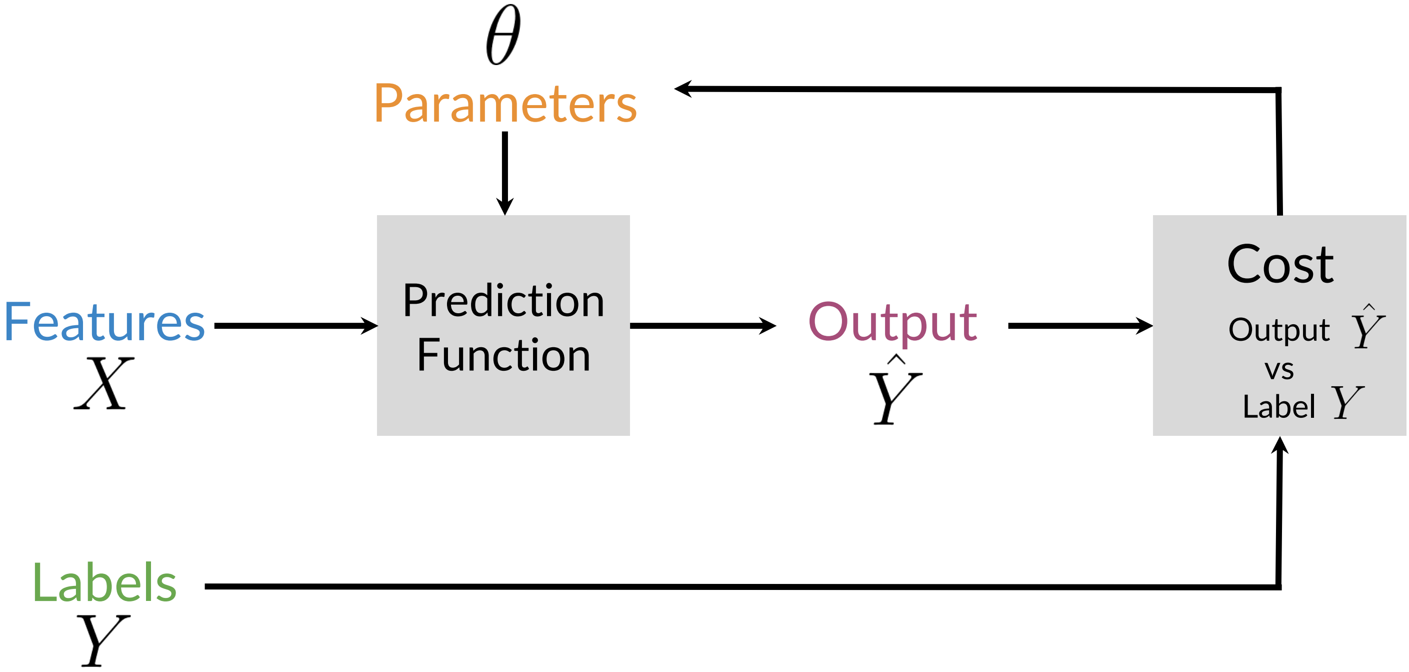
deeplearning.ai

Supervised ML and Sentiment Analysis

Outline

- Review Supervised ML
- Build your own tweet classifier!

Supervised ML (training)



Sentiment analysis

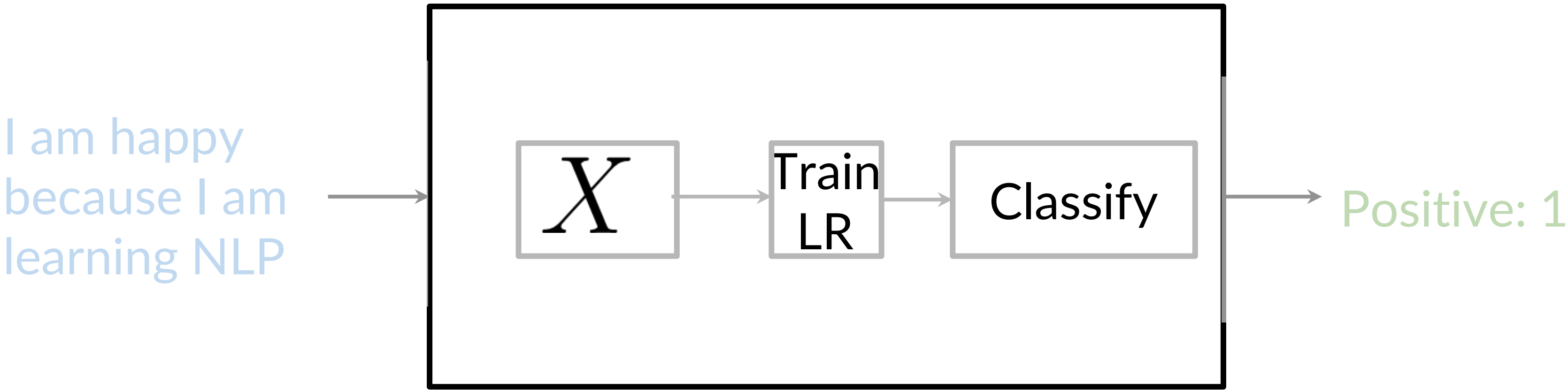
Tweet: I am happy because I am learning NLP

Positive: 1

Negative: 0

Logistic regression

Sentiment analysis



Summary

- Features, Labels $\longrightarrow$ Train $\longrightarrow$ Predict
- Extract features $\longrightarrow$ Train LR $\longrightarrow$ Predict sentiment

Vocabulary and Feature Extraction

deeplearning.ai

Outline

- Vocabulary
- Feature extraction
- Sparse representations and some of their issues

Vocabulary

Tweets:

[tweet_1, tweet_2, ..., tweet_m] →

I am happy because I am learning NLP
...
...
I hated the movie

$V =$

all unique words in all tweets

[I, am, happy, because, learning, NLP, ... hated, the, movie]

Feature extraction

I am happy because I am learning NLP

[I , am, happy, because, learning, NLP, ... hated, the, movie]

[1, 1, 1, 1, 1, 1, ... 0, 0, 0]

A lot of zeros! That's a sparse representation.

Problems with sparse representations

I am happy because I am learning

NLP

All zeros!

[1 , 1 , 1 , 1 , 1 , 1 , ... , 0 , ... , 0 , 0 , 0]

1 $\longleftrightarrow$ $|V|$

$$[\theta_0, \theta_1, \theta_2, \ldots, \theta_n] \longrightarrow$$

$$n = |V|$$

1. Large training time
2. Large prediction time

Summary

- Vocabulary: set of unique words
- Vocabulary, Text $\longrightarrow$ [1 ….. 0 ….. 1 .. 0 .. 1 .. 0]
- Sparse representations are problematic for training and prediction times

deeplearning.ai

Negative and Positive Frequencies

Outline

- Populate your vocabulary with a frequency count for each class

Positive and negative counts

Corpus

I am happy because I am learning NLP

I am happy

I am sad, I am not learning NLP

I am sad

Vocabulary
I
am
happy
because
learning
NLP
sad
not

Positive and negative counts

Positive tweets	Negative tweets
I am happy because I am learning NLP	I am sad, I am not learning NLP
I am happy	I am sad

Positive and negative counts

Positive tweets

I am happy because I am learning NLP

I am happy

Vocabulary	PosFreq (1)
I	
am	
happy	2
because	
learning	
NLP	
sad	
not	

Positive and negative counts

Vocabulary	NegFreq (0)
I	
am	3
happy	
because	
learning	
NLP	
sad	
not	1

Negative tweets

I am sad, I am not learning NLP

I am sad

Freq dict

Word frequency in classes

Vocabulary	PosFreq (1)	NegFreq (0)
I	3	3
am	3	3
happy	2	0
because	1	0
learning	1	1
NLP	1	1
sad	0	2
not	0	1

freqs: dictionary mapping from (word, class) to frequency

Summary

- Divide tweet corpus into two classes: positive and negative
- Count each time each word appears in either class

➔ Feature extraction for training and prediction!

Feature extraction with frequencies

deeplearning.ai

Outline

- Extract features from your frequencies dictionary to create a features vector

Word frequency in classes

Vocabulary	PosFreq (1)	NegFreq (0)
I	3	3
am	3	3
happy	2	0
because	1	0
learning	1	1
NLP	1	1
sad	0	2
not	0	1

freqs: dictionary mapping from (word, class) to frequency

Feature extraction

freqs: dictionary mapping from (word, class) to frequency

$$X_m = \left[1, \sum_{w} freqs(w,1), \sum_{w} freqs(w,0)\right]$$

Features of tweet m

Bias

Sum Pos. Frequencies

Sum Neg. Frequencies

Feature extraction

Vocabulary	PosFreq (1)
I	3
am	3
happy	2
because	1
learning	1
NLP	1
sad	0
not	0

I am sad, I am not learning NLP

$$X_m = [1, \sum_w freqs(w,1), \sum_w freqs(w,0)]$$

$$\boxed{8}$$

Feature extraction

Vocabulary	NegFreq (0)
I	3
am	3
happy	0
~~because~~	0
learning	~~1~~
NLP	~~1~~
sad	~~2~~
not	~~1~~

I am sad, I am not learning NLP

$$X_m = [1, \sum_w freqs(w, 1), \sum_w freqs(w, 0)]$$

$\boxed{11}$

Feature extraction

I am sad, I am not learning NLP

$$X_m = [1, \sum_w freqs(w,1), \sum_w freqs(w,0)]$$

$$\downarrow$$

now we have 3 features instead of V (whole vocab) features.

$$X_m = [1, 8, 11]$$ a vector of tweet

Summary

- Dictionary mapping (word,class) to frequencies

$$X_m = [1, \sum_w freqs(w, 1), \sum_w freqs(w, 0)]$$

→ Cleaning unimportant information from your tweets

deeplearning.ai

Preprocessing

Outline

- Removing stopwords, punctuation, handles and URLs
- Stemming
- Lowercasing

Preprocessing: stop words and punctuation

@YMourri and @AndrewYNg are tuning a GREAT AI model at https://deeplearning.ai!!!

Stop words
and
is
are
at
has
for
a

Punctuation
,
.
:
!
“
‘

Preprocessing: stop words and punctuation

@YMourri ~~and~~ @AndrewYNg ~~are~~ tuning ~~a~~ GREAT AI model ~~at~~ https://deeplearning.ai!!!

@YMourri @AndrewYNg tuning GREAT AI model https://deeplearning.ai!!!

Stop words
and
is
are
at
has
for
a

Punctuation
,
.
:
!
“
‘

Preprocessing: stop words and punctuation

@YMourri @AndrewYNg tuning GREAT AI model https://deeplearning.ai~~!!!~~

@YMourri @AndrewYNg tuning GREAT AI model https://deeplearning.ai

Stop words
and
is
a
at
has
for
of

Punctuation
,
.
:
!
“
‘

Preprocessing: Handles and URLs

~~@YMourri @AndrewYNg~~ tuning GREAT AI
model
~~https://deeplearning.ai~~

tuning GREAT AI model

Preprocessing: Stemming and lowercasing

tuning GREAT AI model

tun

tune

tuned

tuning

all turn to tun

reduces the vocab

GREAT

Great

great

great

Preprocessed tweet:

[tun, great, ai, model]

final representation

Summary

- Stop words, punctuation, handles and URLs
- Stemming
- Lowercasing
- Less unnecessary info $\longrightarrow$ Better times

deeplearning.ai

Putting it all together

Outline

- Generalize the process
- How to code it!

General overview

I am Happy Because i am learning NLP @deeplearning

Preprocessing

[happy, learn, nlp]

Feature Extraction

Bias ← [1, 4, 2] → Sum negative frequencies

Sum positive frequencies

General overview

$$
\begin{array}{cc}
 & \begin{array}{ccc} \quad & \text{Feat1} & \text{feature 3} \end{array} \\
\begin{array}{c} \text{tweet1} \\ \text{tweet2} \\ \\ \\ \end{array} &
\begin{bmatrix}
1 & X_1^{(1)} & X_2^{(1)} \\
1 & X_1^{(2)} & X_2^{(2)} \\
\vdots & \vdots & \vdots \\
1 & X_1^{(m)} & X_2^{(m)}
\end{bmatrix}
\end{array}
\longleftrightarrow
$$

[[1, 40, 20],
[1, 20, 50],
...
[1, 5,
35]]

General Implementation

```
freqs = build_freqs(tweets,labels) #Build frequencies dictionary

X = np.zeros((m,3)) #Initialize matrix X

for i in range(m): #For every tweet

        p_tweet = process_tweet(tweets[i]) #Process tweet

    X[i,:] = extract_features(p_tweet,freqs) #Extract Features
```

\# of tweets

Summary

- Implement the feature extraction algorithm for your entire set of tweets
- Almost ready to train!

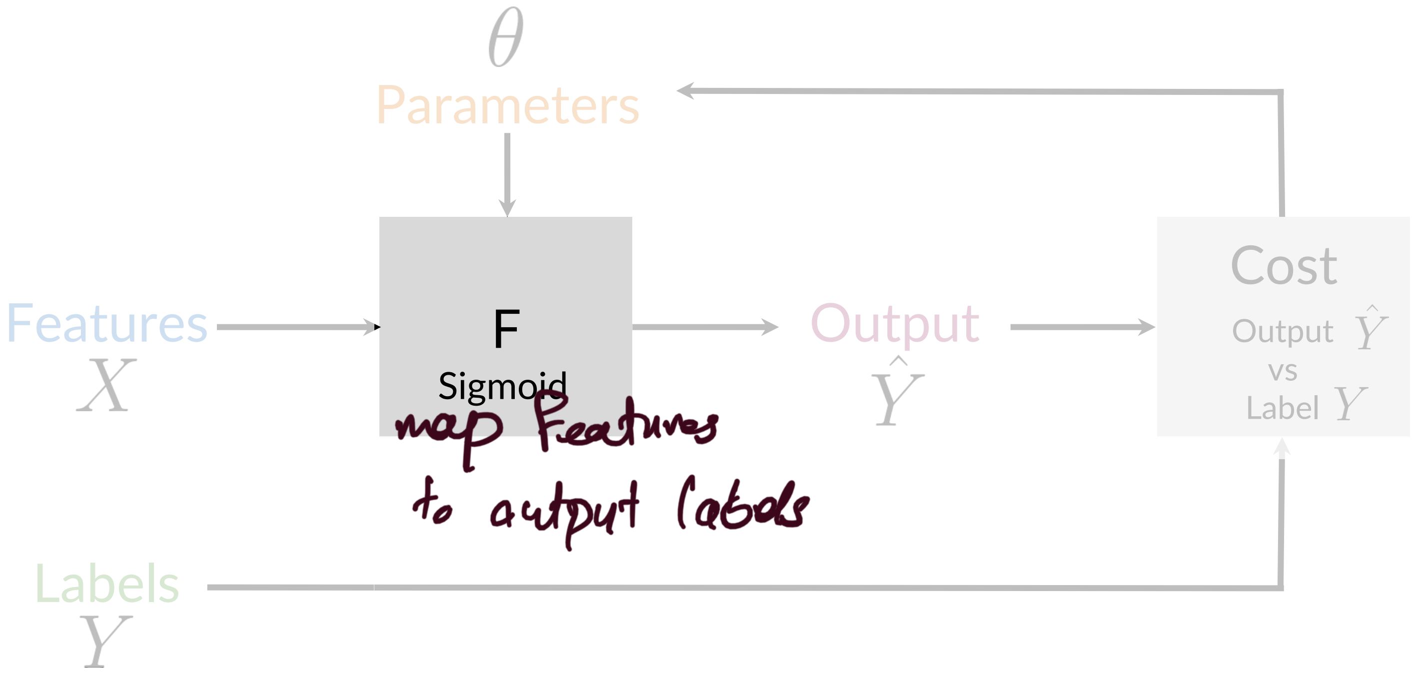
deeplearning.ai

Logistic Regression Overview

Outline

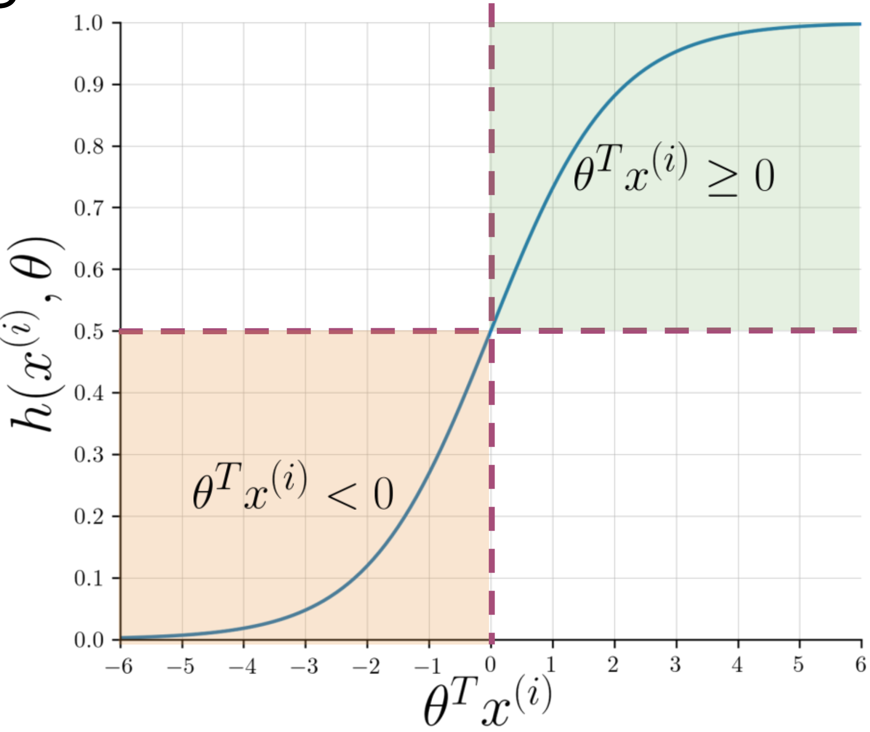
- Supervised learning and logistic regression
- Sigmoid function

Overview of logistic regression



Overview of logistic regression

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



Overview of logistic regression

@YMourri and @AndrewYNg are tuning a GREAT AI model

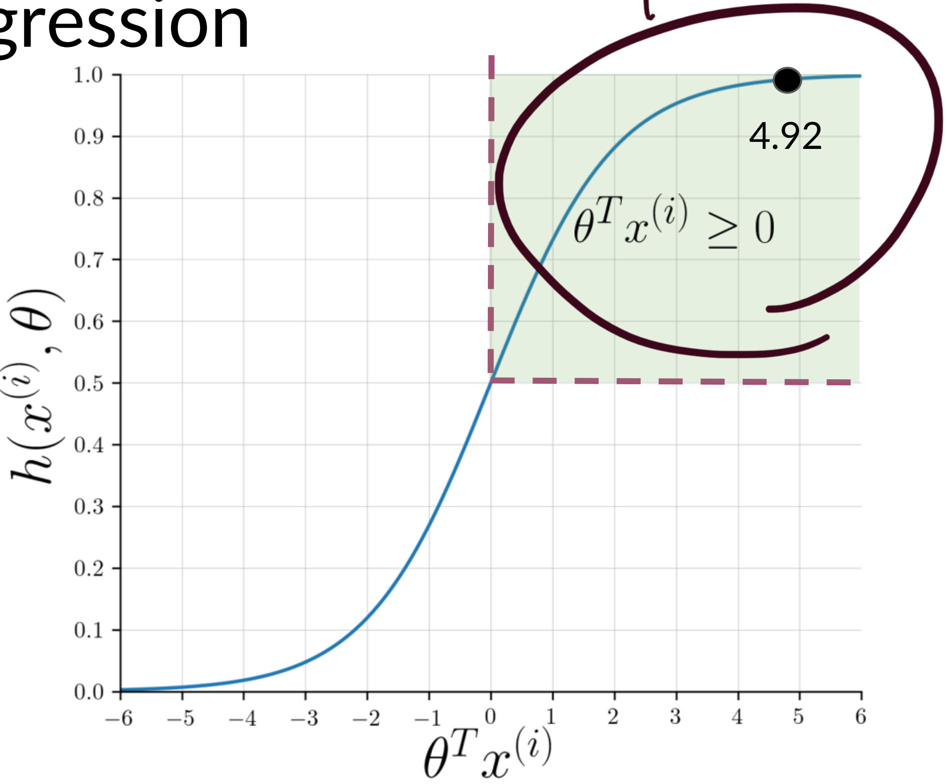
$\downarrow$

[tun, ai, great, model]

$\downarrow$

$$x^{(i)} = \begin{bmatrix} 1 \\ 3476 \\ 245 \end{bmatrix} \quad \theta = \begin{bmatrix} 0.00003 \\ 0.00150 \\ -0.00120 \end{bmatrix}$$

possitive label



Summary

- Sigmoid function
- $\theta^T x^{(i)} \geq 0 \longrightarrow h(x^{(i)}, \theta) \geq 0.5$, positive
- $\theta^T x^{(i)} < 0 \longrightarrow h(x^{(i)}, \theta) < 0.5$, negative

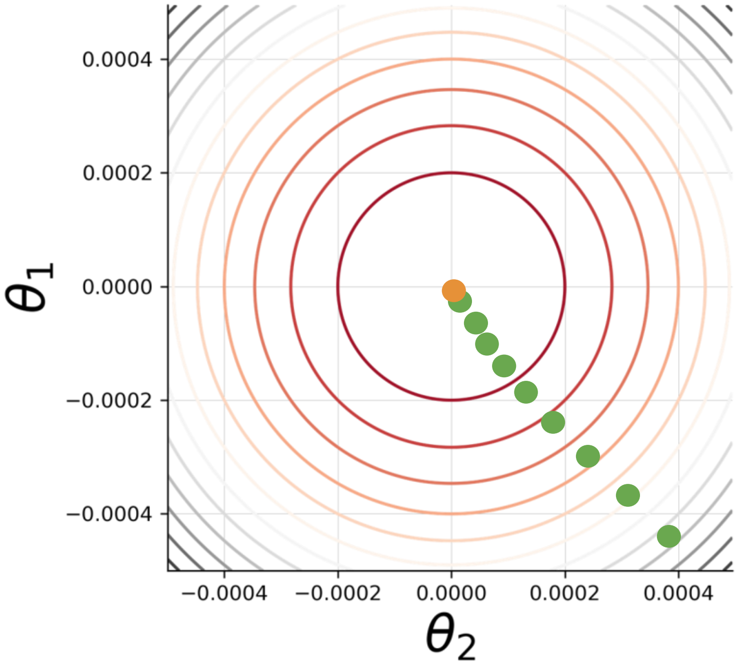
deeplearning.ai

Logistic Regression: Training

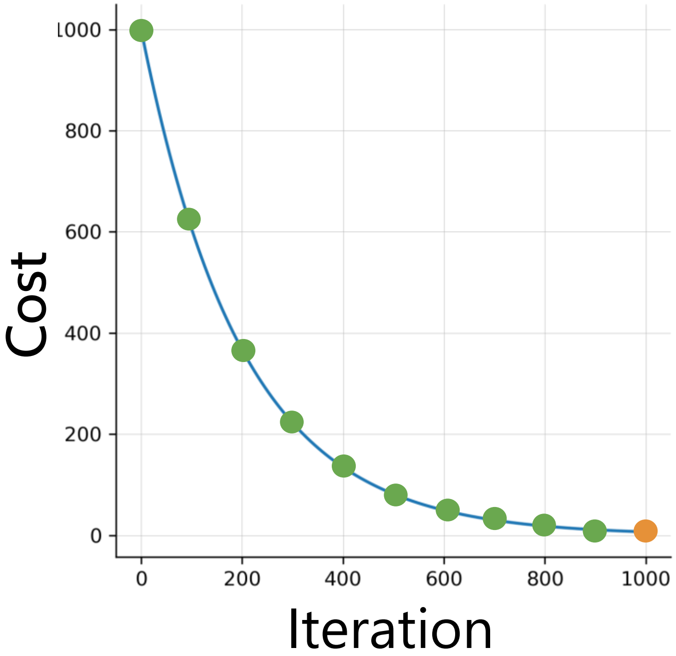
Outline

- Review the steps in the training process
- Overview of gradient descent

Training LR



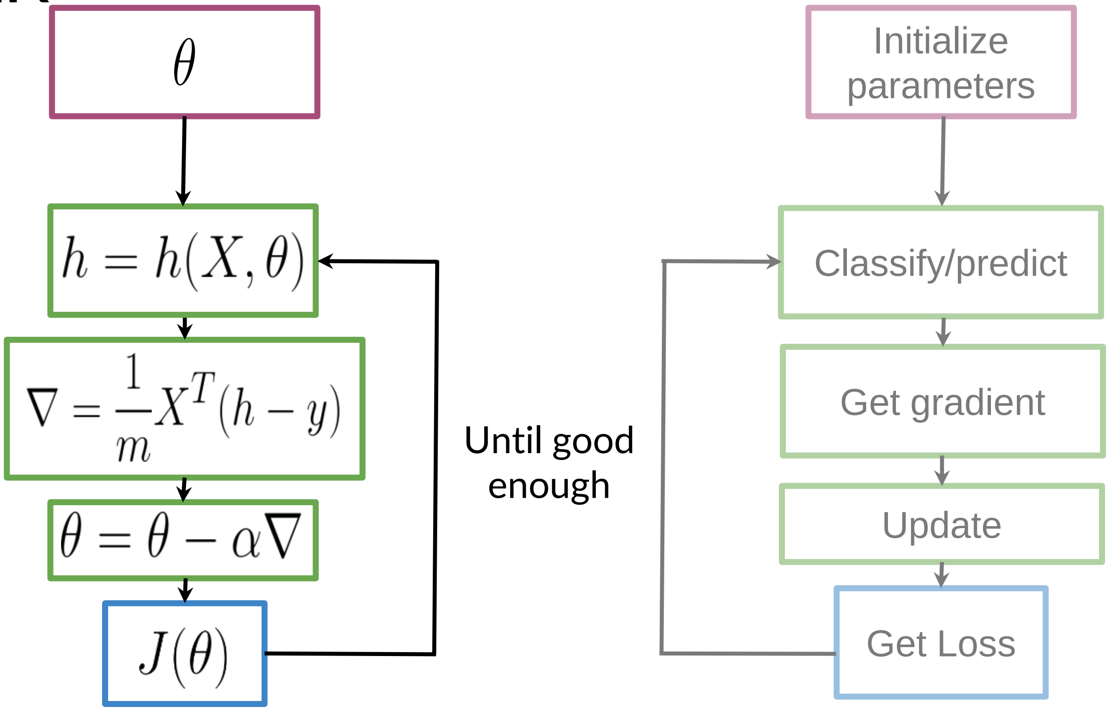
0.0004
0.0002
0.0000
−0.0002
−0.0004
θ_1
−0.0004
−0.0002
0.0000
0.0002
0.0004
θ_2



1000
800
600
400
200
0
Cost
0
200
400
600
800
1000
Iteration

gradient descent

Training LR



Summary

- Visualize how gradient descent works
- Use gradient descent to train your logistic regression classifier

➔ Compute the accuracy of your model

deeplearning.ai

Logistic Regression: Testing

Outline

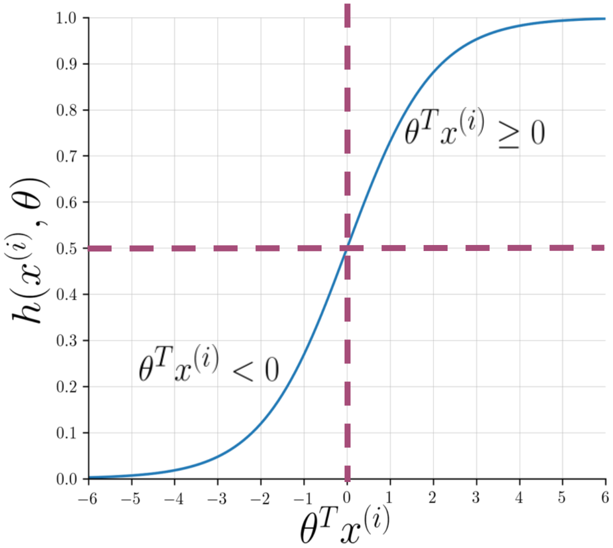
- Using your validation set to compute model accuracy
- What the accuracy metric means

Testing logistic regression

- X_{val} Y_{val} θ

$$h(X_{val}, \theta)$$

$$pred = h(X_{val}, \theta) \geq 0.5$$



Testing logistic regression

- $X_{val}\ Y_{val}\ \theta$

$$h(X_{val}, \theta)$$

$$pred = h(X_{val}, \theta) \geq 0.5$$

$$\begin{bmatrix} 0.3 \\ 0.8 \\ 0.5 \\ \vdots \\ h_m \end{bmatrix} \geq 0.5 = \begin{bmatrix} 0.3 \geq 0.5 \\ 0.8 \geq 0.5 \\ 0.5 \geq 0.5 \\ \vdots \\ pred_m \geq 0.5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ \vdots \\ pred_m \end{bmatrix}$$

Testing logistic regression

- $X_{val} \quad Y_{val} \quad \theta$

$$h(X_{val}, \theta)$$

$$pred = h(X_{val}, \theta) \geq 0.5$$

$$\sum_{i=1}^{m} \frac{(pred^{(i)} == y_{val}^{(i)})}{m}$$

$$\begin{bmatrix} 0 \\ 1 \\ 1 \\ \vdots \\ pred_m \end{bmatrix} == \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ Y_{val_m} \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ \vdots \\ pred_m == Y_{val_m} \end{bmatrix}$$

Testing logistic regression

$$Y_{val} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} \quad pred = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (Y_{val} == pred) = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$$\boxed{\text{accuracy} = \frac{4}{5} = 0.8}$$

Summary

- $X_{val}\ \ Y_{val}$ ⟶ Performance on unseen data
- Accuracy ⟶ $$\sum_{i=1}^{m} \frac{(pred^{(i)} == y_{val}^{(i)})}{m}$$

To improve model: step size, number of iterations, regularization, new features, etc.

deeplearning.ai

Logistic Regression: Cost Function

Outline

- Overview of the logistic cost function, AKA the binary cross-entropy function

Cost function for logistic regression

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

avg over

all training samples

(# m)

Cost function for logistic regression

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

Cost function for logistic regression

label prediction

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

$y^{(i)}$	$h(x^{(i)}, \theta)$	
0	any	0
1	0.99	~0
1	~0	-inf

} ok prediction

wrong prediction

Cost function for logistic regression

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\Big[\overbrace{y^{(i)}\log h(x^{(i)},\theta)}^{\text{negative}} + \overbrace{(1-y^{(i)})\log(1-h(x^{(i)},\theta))}^{\text{negative}}\Big]$$

$y^{(i)}$	$h(x^{(i)},\theta)$	
1	any	0
0	0.01	~0
0	~1	-inf

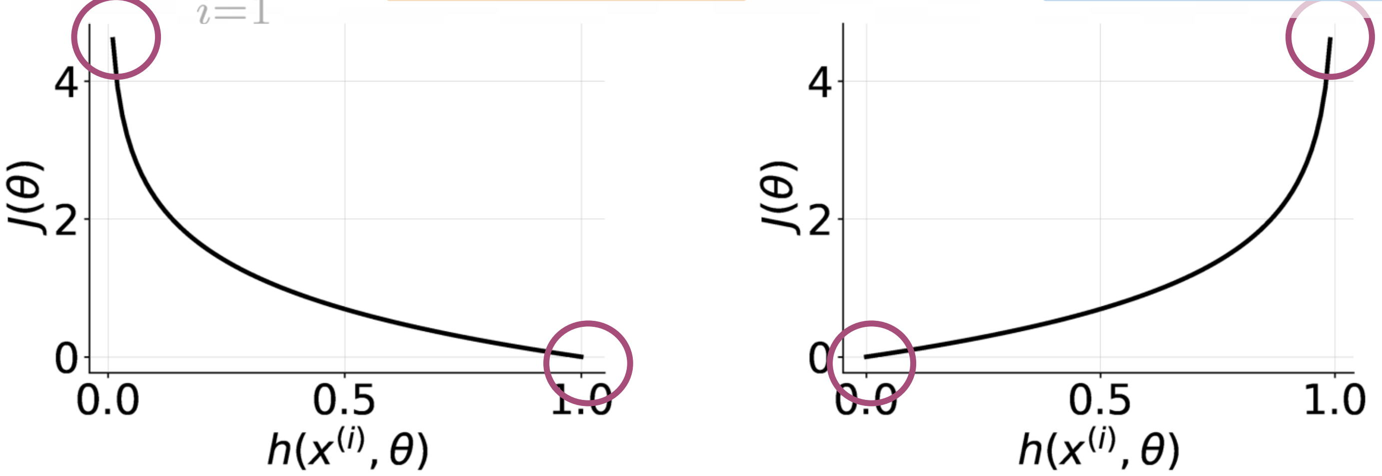
→ wrong prediction

Cost function for logistic regression

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

Cost function for logistic regression

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



Summary

- Strong disagreement = high cost
- Strong agreement = low cost
- Aim for the lowest cost!