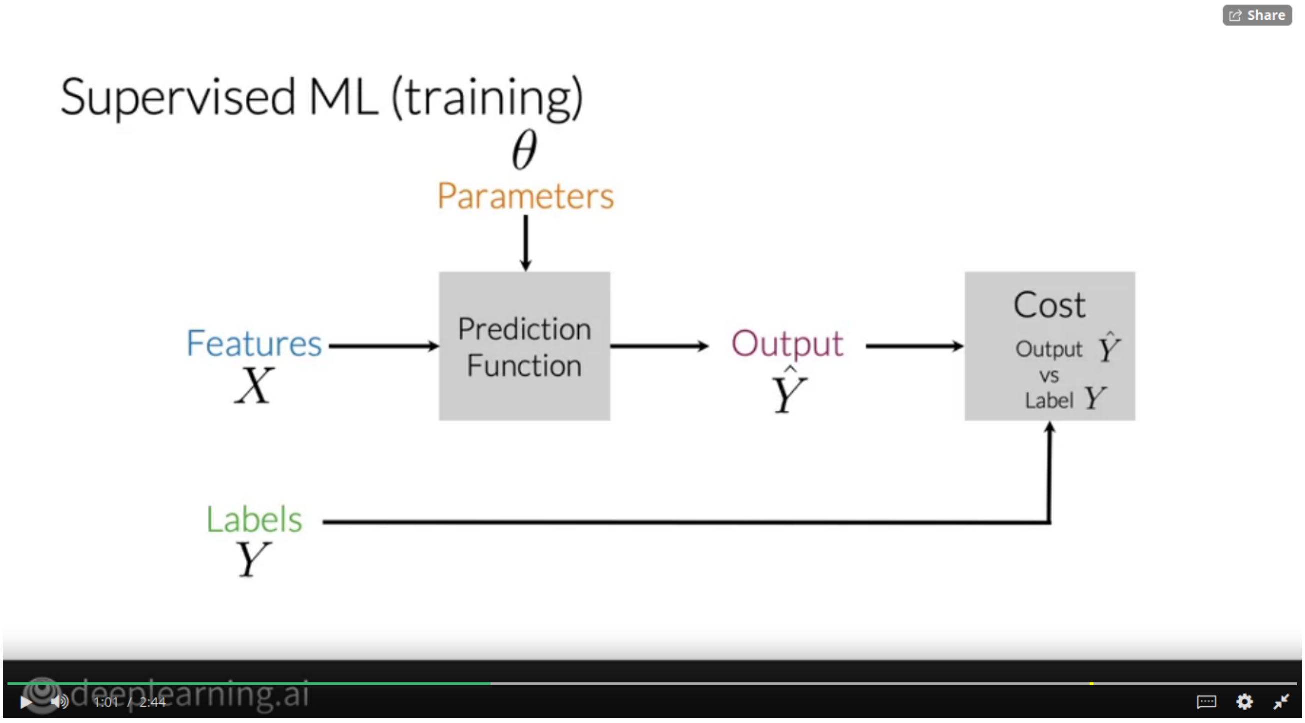
Natural Language Processing with Classification and Vector Spaces

Week1:

Supervised ML & Sentiment Analysis



Supervised ML (training)

1. In supervised learning, we have input X and output Y. We try to fit a function f(X) = Y, such that predicted value of function f is close to Y. We change our parameters at each iteration to minimize cost.

Vocabulary & Feature Extraction

Let's consider a Tweet

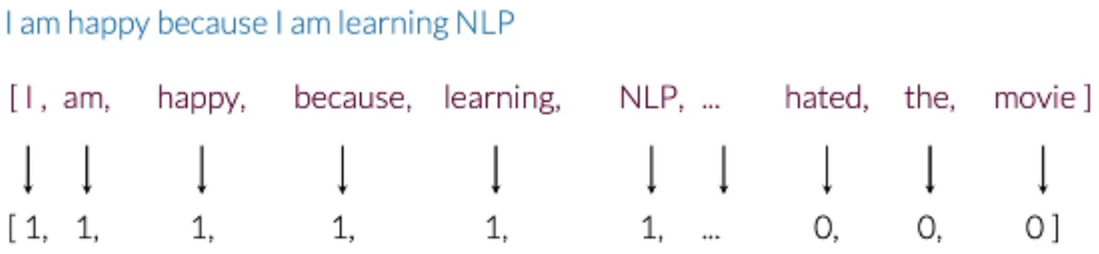
Tweet1: I am happy because I am learning NLP.

Tweet2: I Hated the movie

To represent these tweets in vector form we need to follow the following steps:

1. List all the unique words from all the available tweets.

2. Assign value =1 if that word appears in dictionary else 0.


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

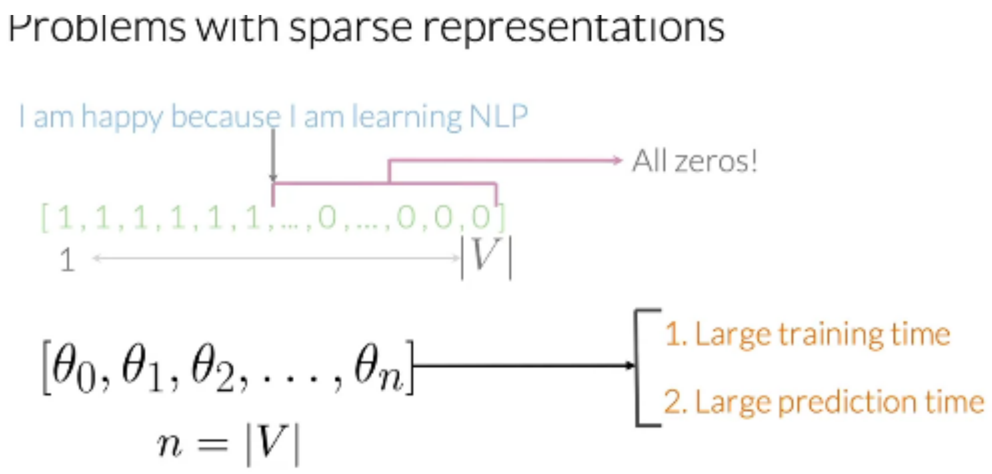
Problem With Sparse Representation

1. Most of the values are zeros if my tweet is small.

2. Logistic Regression will require a V number of parameters to train for each word in the vocabulary.

3. It will take more training time since vector size is very big

4. Prediction will also be slower.

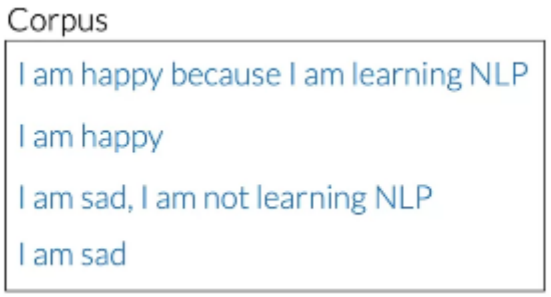


Problems with sparse representations

Negative and Positive Frequencies

Corpus: a collection of written texts

Consider having a corpus of tweets as given below:



Corpus

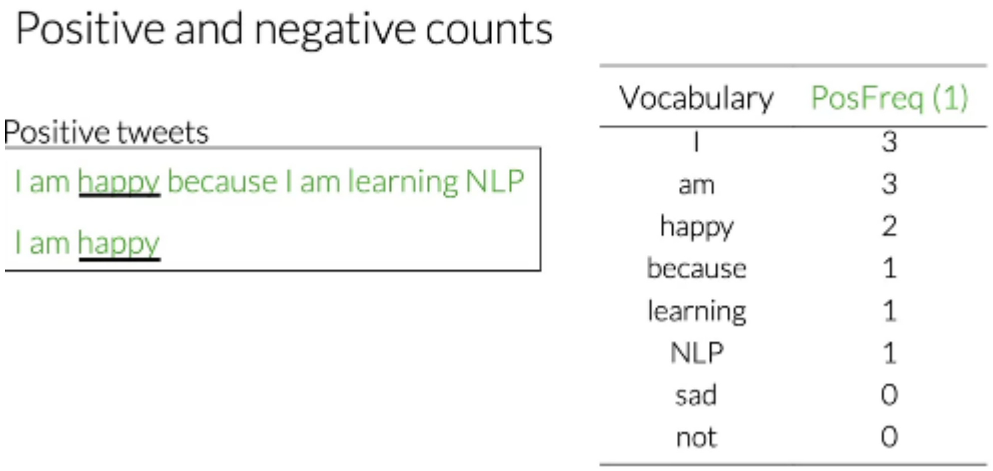
I am happy because I am learning NLP

I am happy

I am sad, I am not learning NLP

I am sad

To count the number of positive and negative frequencies, we will make a table as given below:



Positive and negative counts

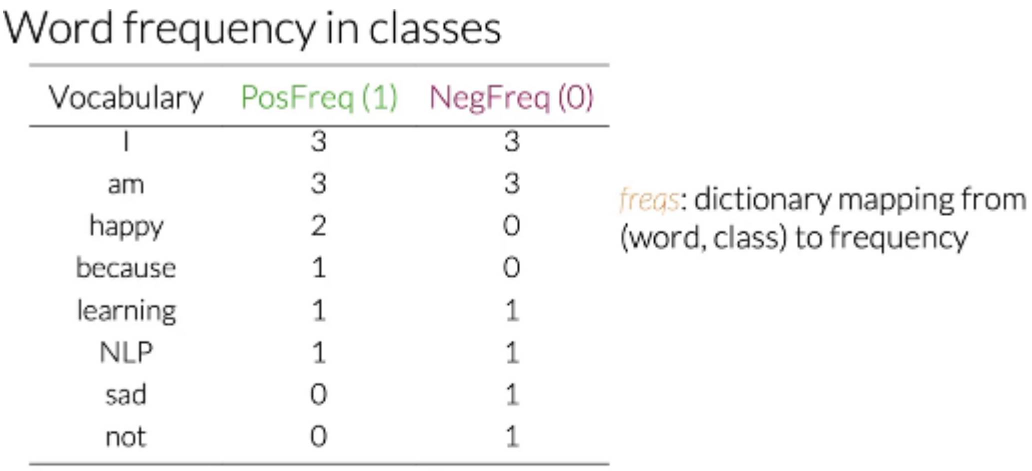
Positive tweets

I am happy because I am learning NLP

I am happy

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

Similarily for negative class, we can count the frequencies.

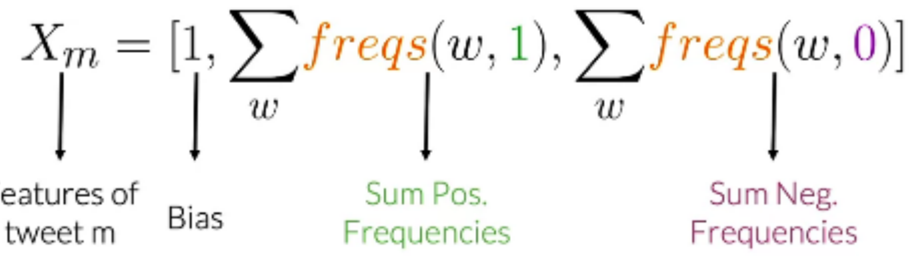


Word frequency in classes

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	1
not	0	1

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

Feature Extraction with Frequencies



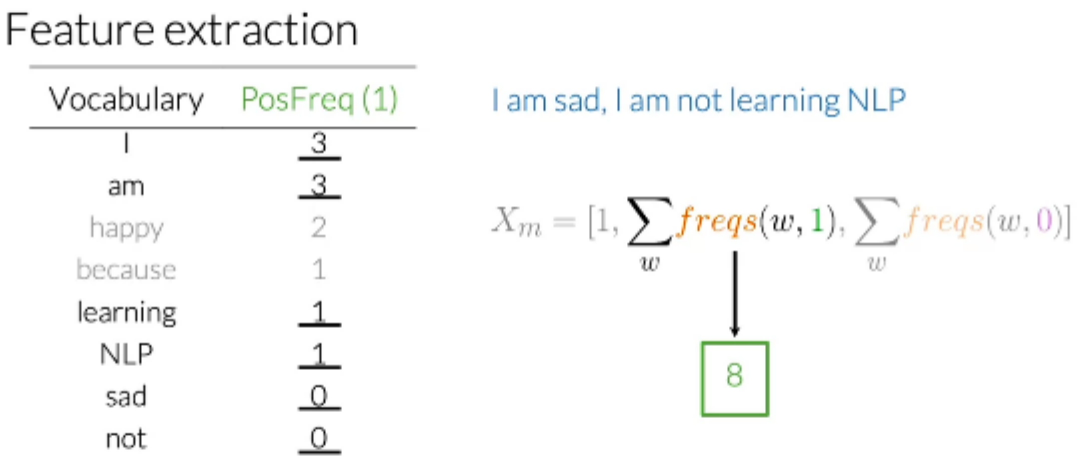
$$X_m = [1, \sum_w freqs(w,1), \sum_w freqs(w,0)]$$

Features of tweet m

Bias

Sum Pos. Frequencies

Sum Neg. Frequencies



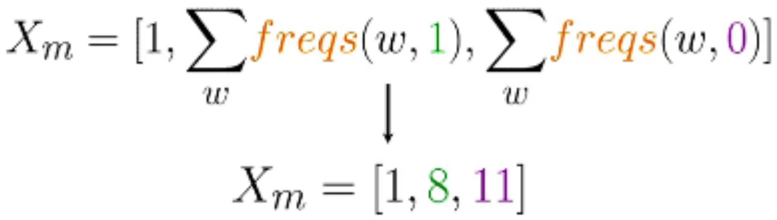
Feature extraction

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

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

$$\boxed{8}$$



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

$$X_m = [1, 8, 11]$$

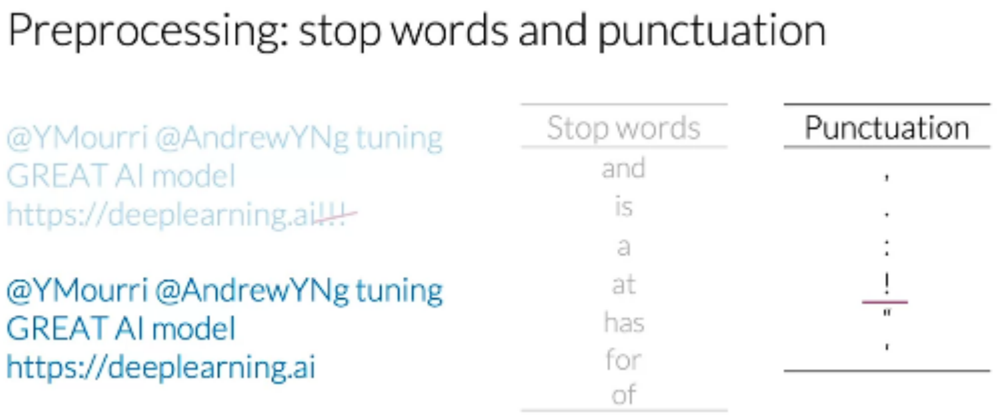
Preprocessing

Preprocess Tweet:

@Ymourri @AndrewNg are tuning a GREAT ai Model at https://deeplearning.ai

1. We need to remove stop words and punctuation mark which does not contribute any meaning in the task of sentiment analysis

After removing StopWords and punctuation from the tweet.



Preprocessing: stop words and punctuation

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

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

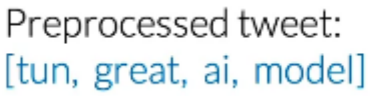
Stop words
and
is
a
at
has
for
of

Punctuation
,
.
:
~~!~~
"
'

tweets having handles and URLs also does not contribute anything to Sentiment Analysis. We will remove them too.

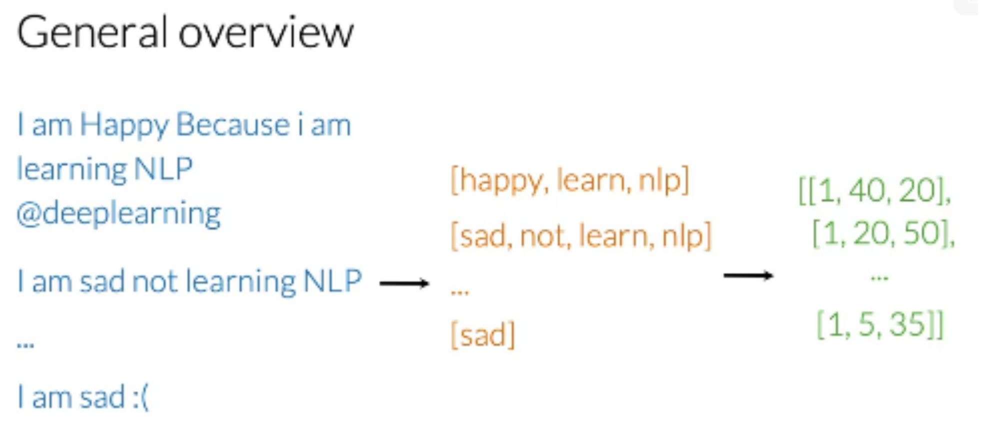
2. We need to perform Stemming(Transforming any word to its base term). SO the word **tune, tuned or tuning** have the same base word, so after stemming it will become **tun.**

3. lower case all the words. GREAT, Great, great will reduced to great. since it does not change the sentiment of the sentence.



Preprocessed tweet:
[tun, great, ai, model]

Putting It All Together(Stemming, tokenizing, Removing Stop Words, Punctuation etc)



General overview

I am Happy Because i am learning NLP @deeplearning

I am sad not learning NLP →

...

I am sad :(

[happy, learn, nlp]

[sad, not, learn, nlp]

...

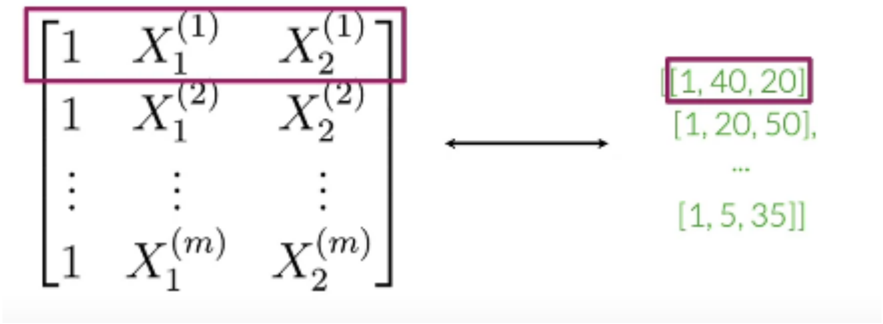
[sad]

→

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

For each tweet, we will use the sum of +ve and -ve frequency to represent it in vector form.

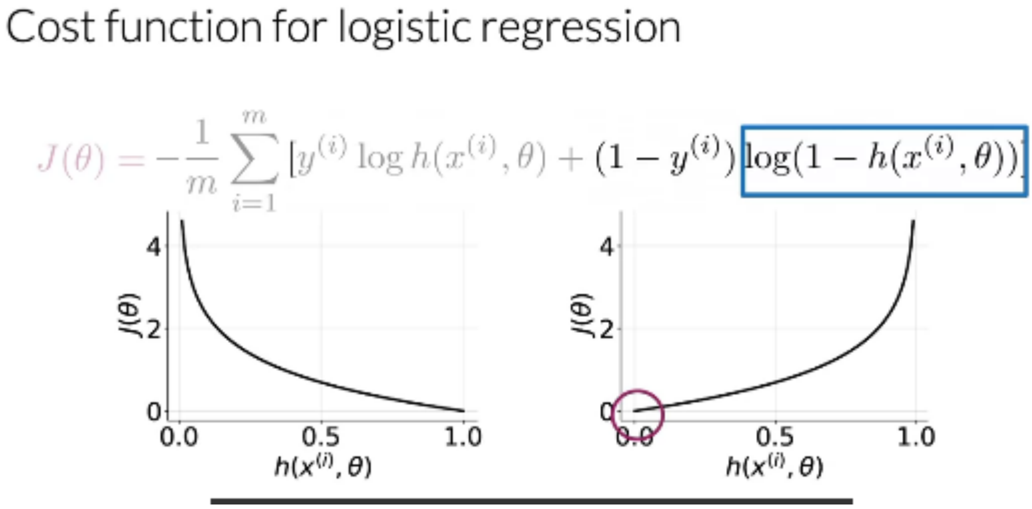
At the end, you will have X matrix with m rows and 3 columns. as shown below.



$$\boxed{\begin{matrix}1 & X_1^{(1)} & X_2^{(1)}\end{matrix}} \begin{bmatrix}1 & X_1^{(1)} & X_2^{(1)} \\ 1 & X_1^{(2)} & X_2^{(2)} \\ \vdots & \vdots & \vdots \\ 1 & X_1^{(m)} & X_2^{(m)}\end{bmatrix} \longleftrightarrow$$

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

Logistic Regression Cost Function



Cost function for logistic regression

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

When label is 0 and output is 0, then Cost = 0, when label is 1 and output is 1 then cost = 0, else cost is +ve inf.