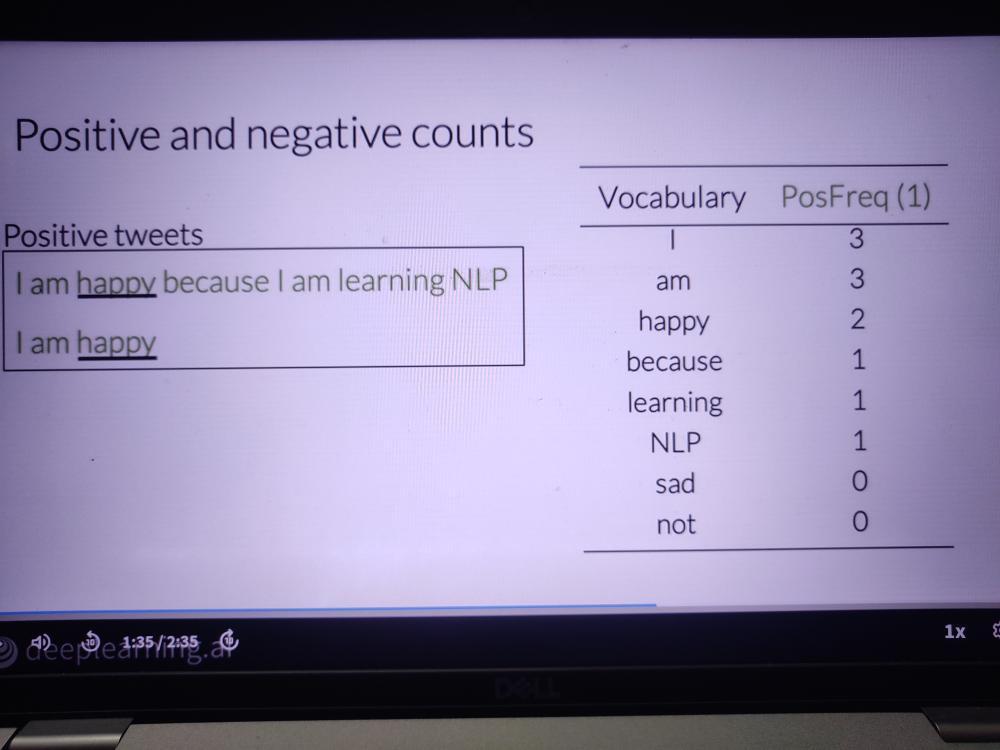
In supervised ML we have input features **X** and a set of labels **Y**. Now to make sure we’re getting the most accurate predictions based on our data, our goal is to minimize our error rates or cost as much as possible.And to do this,we’re going to run our prediction function which takes in parameters data to map our features to output labels **Ŷ**. The best mapping from features to labels is achieved when the difference between the expected values **Y** and the predicted values **Ŷ** is minimized. Which the cost function does, comparing how closely our output Y hat is to our label **Y**. Then we can update our parameters and repeat the whole process until our cost is minimized.

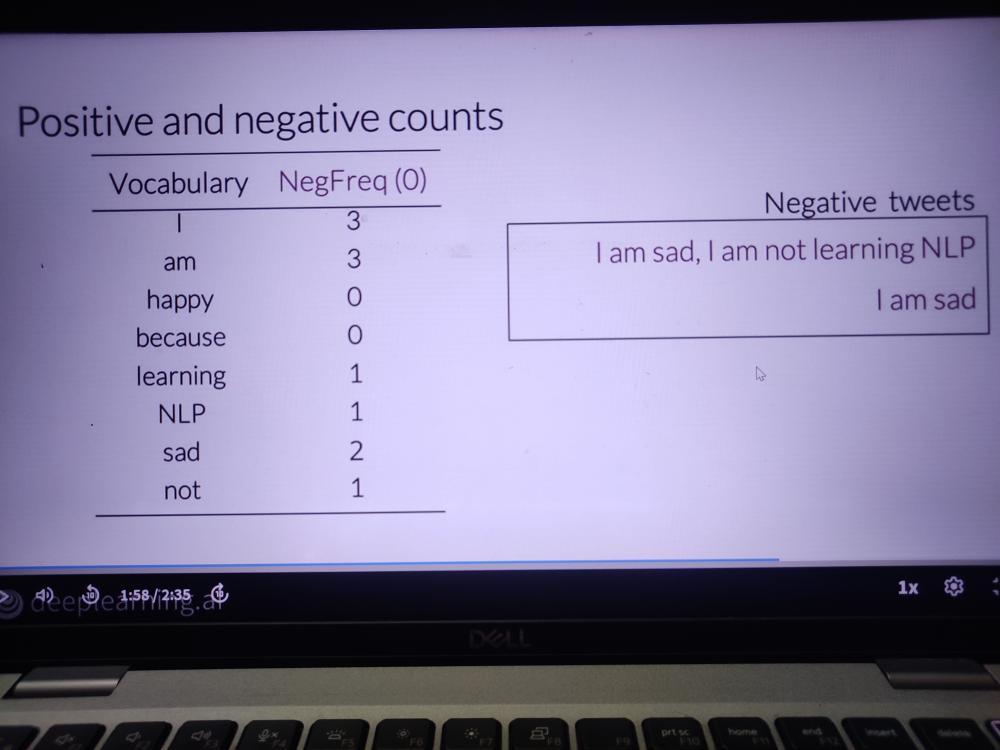
Vocabulary - is a set of unique words

How many parameters would a logistic regression classifier have to learn with inputs represented?

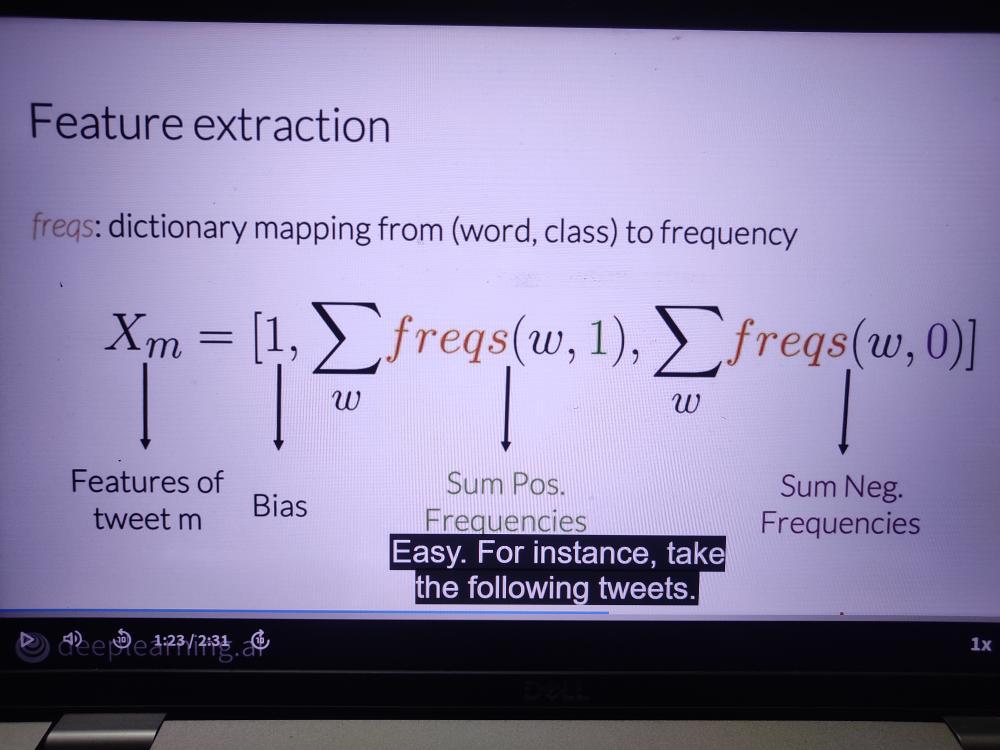
“I am happy because I am learning NLP”, “I hated that movie”, “I love working at DL”

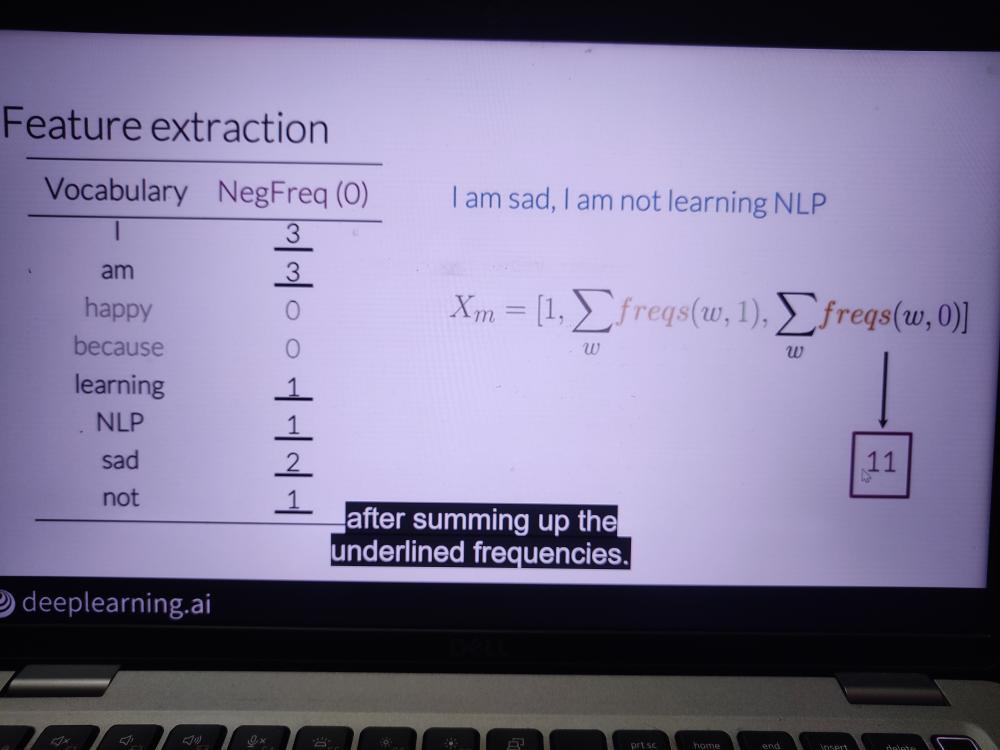
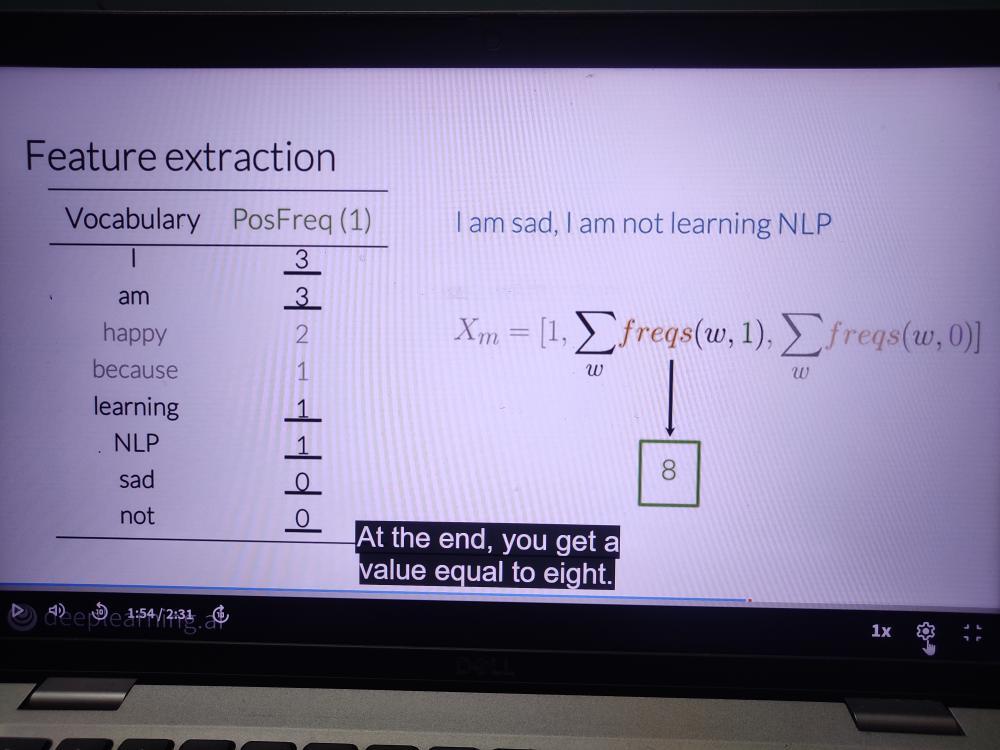
Answer - 14

****

****

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





Some stop words - and, is, are, at, has, for, a

**Preprocessing: Stemming and lowercasing**

Stemming in NLP is transforming any word to its base stem, which we could define as the set of characters that are used to construct the word and it’s derivatives.

Ex - **tuning GREAT AI model**

tun -> tune | tuned | tuning

After we perform stemming on our corpus, the word tune, tuned, and tuning will be reduced to the stem tun.

GREAT | Great | great

**Matrix X**

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