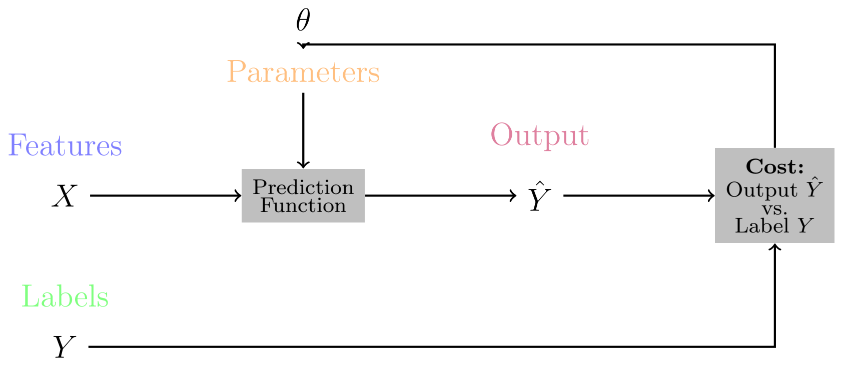
Contents

1 Classification and Vector Spaces

1.1 Supervised ML and Sentiment Analysis

In supervised ML, we have input features X and a set of labels Y. To get the most accurate predictions, we try to minimize our *error rates* or *cost function* as much as possible: to do this, we'll run our prediction function which takes in parameters θ to map you input features to output labels $\hat{Y}$. The best mapping is achieved when the difference between the expected values Y and the predicted values $\hat{Y}$ is minimized, which the cost function does by comparing how closely your output $\hat{Y}$ is to your label Y. You can then update your parameters and repeat the whole process until your cost is minimized.



How about the supervised ML classification task of sentiment analysis? Suppose we're given a tweet that says, "I'm happy because I'm learning NLP": and the objective in the task is to predict whether a tweet has a positive or negative sentiment. We'll do this by starting with a training set where tweets with a positive label have a label of unit value, and tweets with a negative sentiment have a label of zero. To get started building a logistic regression classifier that's capable of predicting sentiments of an arbitrary tweet, we first need to process the raw tweets in our training data set and extract useful features. Then, we will train our logistic regression classifier while minimimizing the cost. Finally, we'll be able to make predictions.

How to represent text as a vector In order to represent text as a vector, we need to first build a vocabulary. We define the vocabulary V as the *set* of unique words from your input data (e.g. your listing of tweets). To get this listing, we quite literally need to comb through all words from all input data and save every new word that appears in our search. To represent a tweet as a vector, we can use a one-hot encoding with our vocabulary: i.e. each tweet will be represented with a length $|V|$ vector where elements are binary-valued - a one indicates the word is in the tweet and a zero indicates the absence of a word in a tweet. We call this a *sparse* representation because the number of non-zero entries is relatively small when compared with the number of zero entries. Realize that if we are running a logistic regression, we would require learning $|V| + 1$ parameters which can be problematic for large vocabularies. If not prohibitive, it would make training models take excessive time and making predictions would be expensive.

Negative and positive frequencies