In this lecture, we'll use a technique called Bag of Words to build text analytics models.

Fully understanding text is difficult, but Bag of Words provides a very simple approach.

It just counts the number of times each word appears in the text and uses these counts as the independent variables.

For example, in the sentence, "This course is great.

I would recommend this course my friends," the word this is seen twice, the word course is seen twice, the word great is seen once, et cetera.

In Bag of Words, there's one feature for each word.

This is a very simple approach, but is often very effective, too.

It's used as a baseline in text analytics projects and for natural language processing.

This isn't the whole story, though.

Preprocessing the text can dramatically improve the performance of the Bag of Words method.

One part of preprocessing the text is to clean up irregularities.

Text data often as many inconsistencies that will cause algorithms trouble.

Computers are very literal by default.

Apple with just an uppercase A, APPLE all in uppercase letters, or ApPLe with a mixture of uppercase and lowercase letters will all be counted separately.

We want to change the text so that all three versions of Apple here will be counted as the same word, by either changing all words to uppercase or to lower case.

We'll typically change all the letters to lowercase, so these three versions of Apple will all become Apple with lower case letters and will be counted as the same word.

Punctuation can also cause problems.

The basic approach is to deal with this is to remove everything that isn't a standard number or letter.

However, sometimes punctuation is meaningful.

In the case of Twitter, @Apple denotes a message to Apple, and #Apple is a message about Apple.

For web addresses, the punctuation often defines the web address.

For these reasons, the removal of punctuation should be tailored to the specific problem.

In our case, we will remove all punctuation, so @Apple, Apple with an exclamation point, Apple with dashes will all count as just Apple.

Another preprocessing task we want to do is to remove unhelpful terms.

Many words are frequently used but are only meaningful in a sentence.

These are called stop words.

Examples are the, is, at, and which.

It's unlikely that these words will improve the machine learning prediction quality, so we want to remove them to reduce the size of the data.

There are some potential problems with this approach.

Sometimes, two stop words taken together have a very important meaning.

For example, "The Who"-- which is a combination of two stop words-- is actually the name of the band we see on the right here.

By removing the stop words, we remove both of these words, but The Who might actually have a significant meaning for our prediction task.

Another example is the phrase, "Take That".

If we remove the stop words, we'll remove the word "that," so the phrase would just say, "take." It no longer has the same meaning as before.

So while removing stop words sometimes is not helpful, it generally is a very helpful preprocessing step.

Lastly, an important preprocessing step is called stemming.

This step is motivated by the desire to represent words with different endings as the same word.

We probably do not need to draw a distinction between argue, argued, argues, and arguing.

They could all be represented by a common stem, argue.

The algorithmic process of performing this reduction is called stemming.

There are many ways to approach the problem.

One approach is to build a database of words and their stems.

A pro is that this approach handles exceptions very nicely, since we have to find all of the stems.

However, it won't handle new words at all, since they are not in the database.

This is especially bad for problems where we're using data from the internet, since we have no idea what words will be used.

A different approach is to write a rule-based algorithm.

In this approach, if a word ends in things like ed, ing, or ly, we would remove the ending.

A pro of this approach is that it handles new or unknown words well.

However, there are many exceptions, and this approach would miss all of these.

Words like child and children would be considered different, but it would get other plurals, like dog and dogs.

This second approach is widely popular and is called the Porter Stemmer, designed by Martin Porter in 1980, and it's still used today.

Stemmers like this one have been written for many languages.

Other options for stemming include machine learning, where algorithms are trained to recognize the roots of words and combinations of the approaches explained here.

As a real example from our data set, the phrase "by far the best customer care service I have ever received" has three words that would be stemmed-- customer, service, and received.

The "er" would be removed in customer, the "e" would be removed in service, and the "ed" would be removed in received.

In the next video, we'll see how to run these preprocessing steps in R.