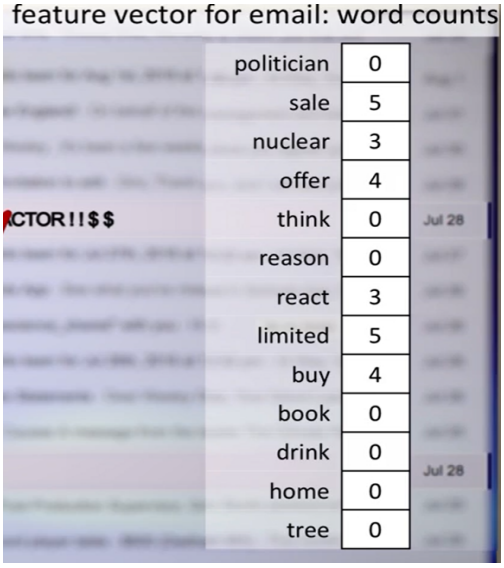
K-Means Clustering- Recap

We have already discussed many interesting and useful methods to find grouping structure in the data.
For example: The K-Means Algorithm.

Let's recap some of the things that we have discussed so far:
For clustering and many other tasks, our data comes in the form of data points. Each data point is a feature vector. We can think of a feature vector as a string of numbers and each number describes a feature.

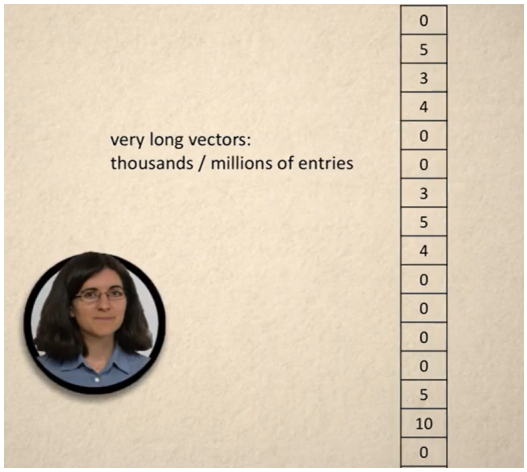


For example:
In the email (Spam or Not Spam) example that we have discussed at the beginning, each email is a data point. It is described by the words it contains. So, we have a huge vector and each entry in this vector corresponds to a word. The number for each word means how often this word occurs in the email. With such representation we can use K-Means.

But we don't get such vectors that easily all the time. Sometimes the feature vector can contain too much useless information or noise. We may not be able to construct such vectors at all. In such cases, we may be able to construct new features from whatever we had.
We will discuss how to do it later.

Now look at some examples:

Data can contain a lot of measurements i.e each data point is a huge feature vector with many entries.



1. Suppose a person is a data point and the description is the variation in the person's genome. This can be huge.
2. Think of a collection of images that are face portraits. Each image is a data point and is described by a few hundred thousand pixels. But, some pixels are more meaningful than others.

For example, the upper right corner of face images, that part is most likely going to be not so relevant. It's a constant value or varies randomly across images since it's background. So there is not a whole lot of information.

The important information is how the images deviate in a major way from some average image. For example, there will be variation in hair, glasses, beard, etc.

These major access variations are what captures that data in a meaningful way.
And maybe there are actually much fewer axes and pixels. As some groups of pixels tend to vary together, because they belong to some subregion like eyebrows and chin. These few trends can help us compress the data.

3. Similarly user ratings can be much more easily understood with patterns. My rating for a movie can be understood more easily if you know which genre of movies I like, for example: comedy or adventure movies.

Finding and recognizing such patterns can help us to reduce the complexity in the data and also to reduce noise and bring out important trends and also to compress it.
We will discuss some important methods for finding such patterns later on.

Sometimes we do not even have a feature vector. Let's take an example from article 1.2, which is about the social network of monks that Dr. Simpson recorded back in the 60's.
The monks are our data points and we want to find groups of friends among them.

Can we use K-Means to find groups here?
The answer is, it's going to be hard. As we don't have any feature vectors. All we know here is who talks to whom.

So, we have data points as the monks and we have a relationship of who talks to whom.
This will give us a graph.
We can draw all the monks on paper and draw lines between those who talk. The structure of points and lines is a graph. The lines are called edges.

This graph tells us a whole lot about the patterns in our data. Groups of friends will have lots of edges between them and between such groups there are not many edges.

We will discuss how a little bit of math will magically find us the groups in the graph.
For all the problems of high level which are mentioned here, the same approach will work. We are going to create new features that represent our data points. These new features will reveal a lot about the hidden structures in the data. The features are constructed by looking at the entire data.