Our Progress So Far

- We've made a lot of progress:
  - Representing unstructured text data in a format amenable to analytics and machine learning.
  - Building a standard text analytics data pre-processing pipeline.
  - Improving the bag-of-words model (BOW) with the use of the mighty TF-IDF.
  - Extending BOW to incorporate word ordering via n-grams.

- However, we've encountered some notable problems as well:
  - Document-term matrices explode to be very wide (i.e., lots of columns).
  - The features of document-term matrices don't contain a lot of signal (i.e., they're sparse).
  - We're running into scalability issues like RAM and huge amounts of computation.
  - The curse of dimensionality.

- The vector space model helps address many of the problems above!

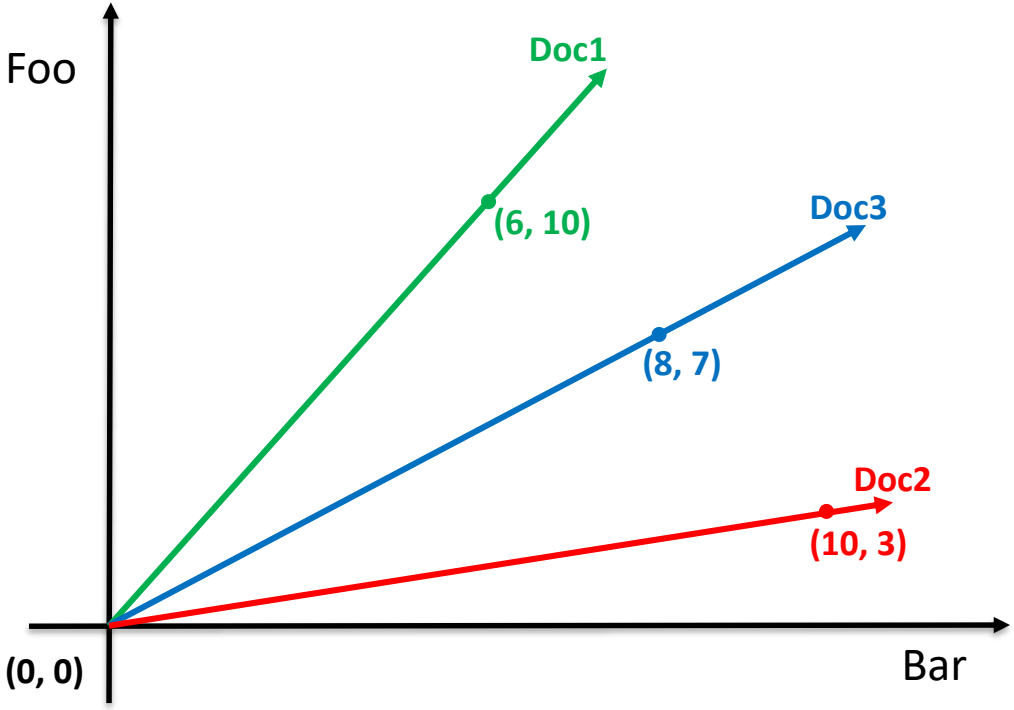
The Vector Space Model

- Core intuition – we represent documents as vectors of numbers.
- Our representation allows us to work with document geometrically.
- Take the hypothetical document-term frequency matrix:

bar	foo
6	10
10	3
8	7

- Three documents
- Two terms

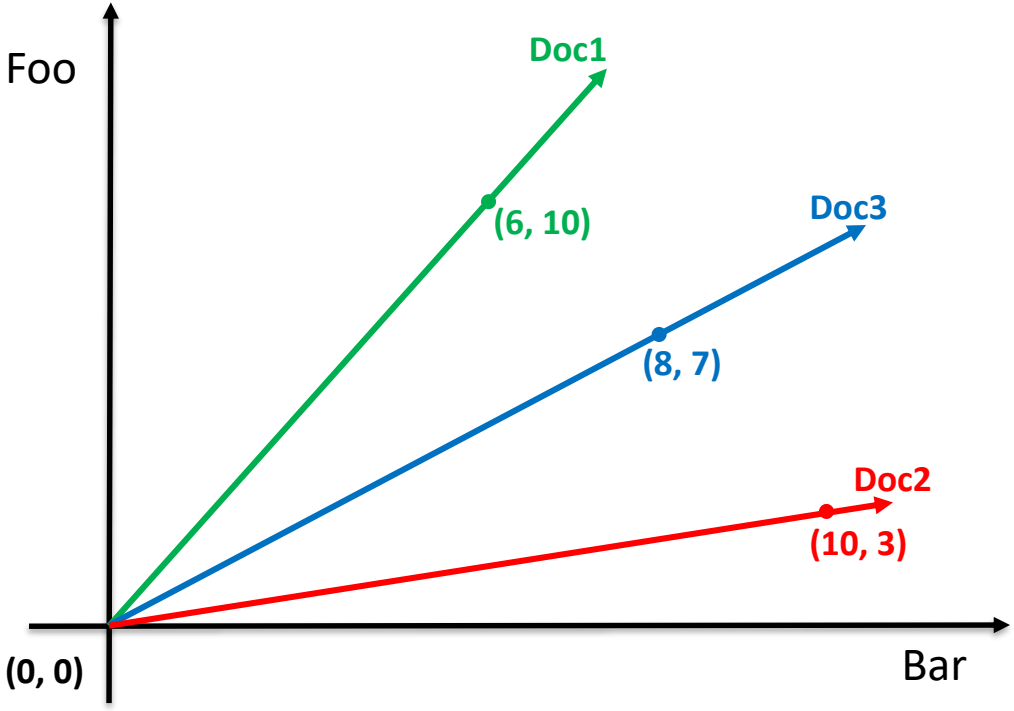
Visualizing Vector Space



Given that we have only two terms, we can visualize our documents using a 2D plane.

If we assume that all document vectors originate from the origin (0, 0) we can plot each document on the plane.

Thinking Geometrically



Thinking about documents geometrically provides us with many advantages.

First, we can intuitively see that certain documents are more alike (e.g., Doc 1 and Doc 3) than others.

We can use mathematics to analyze and understand document relationships!

Dot Product of Two Vectors

Intuition – We can think of the dot product of two document vectors as a proxy for correlation.

$$\text{Dot Product of A, B} = \text{A} \cdot B = \sum_{i=1}^{n} A_i B_i$$

Given the document-term frequency matrix

bar	foo
6	10
10	3
8	7

$$Doc1 \cdot Doc2 = (6 \times 10) + (10 \times 3) = 90$$

$$Doc1 \cdot Doc3 = (6 \times 8) + (10 \times 7) = 118$$

$$Doc2 \cdot Doc3 = (10 \times 8) + (3 \times 7) = 101$$

The dot products align to our geometric understanding (e.g., Doc1 and Doc3 are most alike)

Dot Products of Documents

As dot products of document vectors are useful, we can leverage matrix multiplication to calculate all of them all at once!

$$\text{Dot Product of all Docs} = XX^T$$

Given the document-term frequency matrix

bar	foo
6	10
10	3
8	7

$$\begin{bmatrix} 6 & 10 \\ 10 & 3 \\ 8 & 7 \end{bmatrix} \begin{bmatrix} 6 & 10 & 8 \\ 10 & 3 & 7 \end{bmatrix} = \begin{bmatrix} 136 & 90 & 118 \\ 90 & 109 & 101 \\ 118 & 101 & 113 \end{bmatrix}$$

Intuition – The dot product of the documents is indicative of document correlation given the set of matrix terms.

Dot Products of Terms

We can also take the perspective of taking the dot products of the terms in the document-term frequency matrix!

$$\text{Dot Product of all Terms} = X^T X$$

Given the document-term frequency matrix

bar	foo
6	10
10	3
8	7

$$\begin{bmatrix} 6 & 10 & 8 \\ 10 & 3 & 7 \end{bmatrix} \begin{bmatrix} 6 & 10 \\ 10 & 3 \\ 8 & 7 \end{bmatrix} = \begin{bmatrix} 200 & 146 \\ 146 & 158 \end{bmatrix}$$

Intuition – The dot product of the terms is indicative of term correlation given the set of matrix documents.

Latent Semantic Analysis

Intuition – Extract relationships between the documents and terms assuming that terms that are close in meaning will appear in similar (i.e., correlated) pieces of text.

Implementation – LSA leverages a singular value decomposition (SVD) factorization of a term-document matrix to extract these relationships.

$$SVD\ of\ X = X = U\sum V^T$$

NOTE – We'll need to transpose our matrix!

Where:

U contains the eigenvectors of the term correlations, XX^T

V contains the eigenvectors of the document correlations, X^TX

$\sum$ contains the singular values of the factorization

LSA to the Rescue!

- Latent Semantic Analysis (LSA) often remediates the curse of dimensionality problem in text analytics:
  - The matrix factorization has the effect of combining columns, potentially enriching signal in the data.
  - By selecting a fraction of the most important singular values, LSA can dramatically reduce dimensionality.

- However, there's no free lunch:
  - Performing the SVD factorization is computationally intensive.
  - The reduced factorized matrices (i.e., the "semantic spaces") are approximations.
  - We will need to project new data into the semantic space.

- SVD is effective and is a staple of text analytics pipelines!

Projecting New Data

- As with TF-IDF the use of SVD will require that new data be transformed/projected before predictions can be made!

- The following represents the high-level process for projection:
  - Normalize the document vector (i.e., row) using the term.frequency() function.
  - Complete the TF-IDF projection using the tf.idf() function.
  - Apply the SVD projection on the document vector.

- Mathematically, the SVD projection for document d is:

$$\hat{d} = \sum^{-1} U^T d$$