Natural Language Processing

Dr. Karen Mazidi

Part Four: Documents

Topics

- Vector-space model

Quizzes

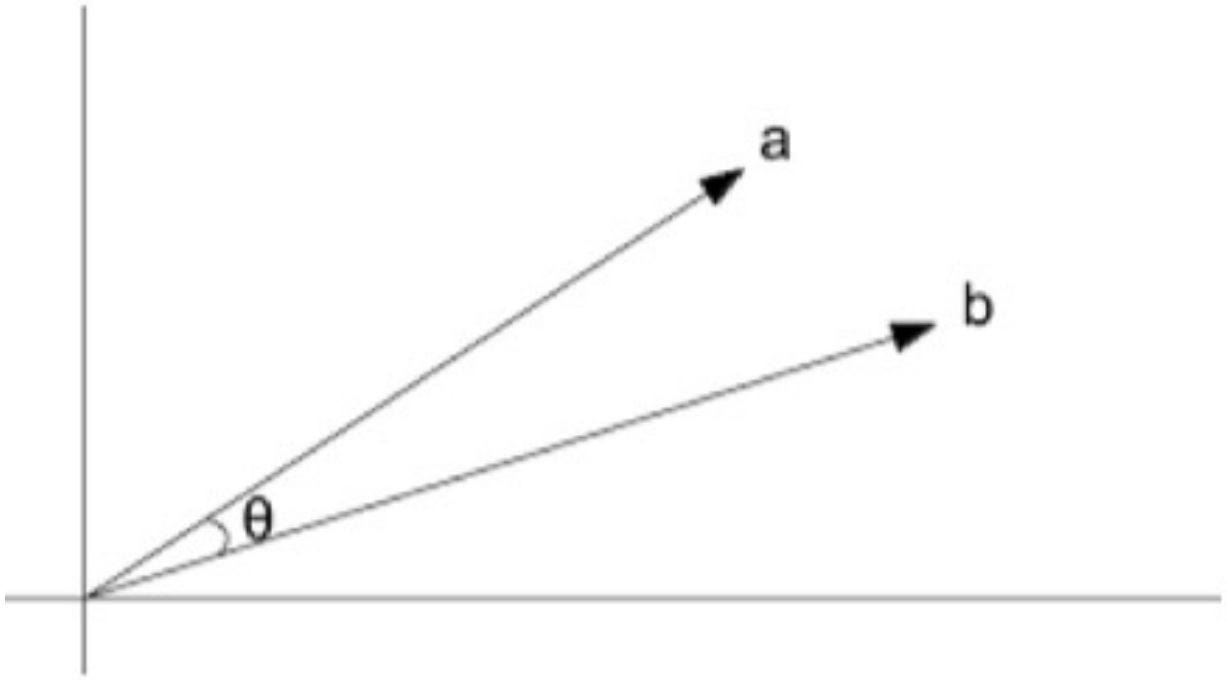
- Q: IE and more

Homework

- no homework on this material

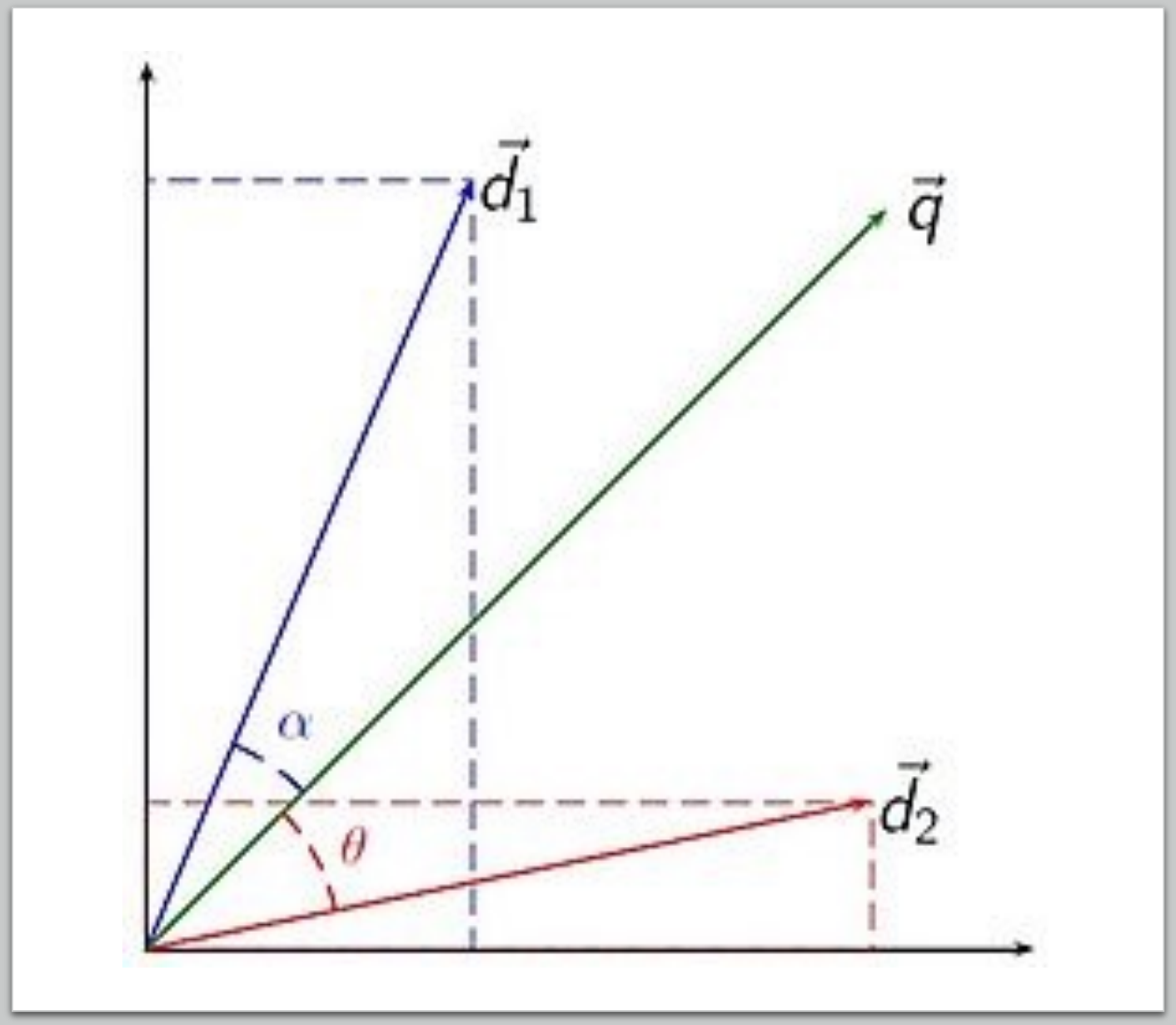
Vector space model

- a vector space model represents documents in a corpus in a common vector space
- the cosine of the angle between vectors is a measure of how similar the documents are
- The closer to 1, the more similar
- The closer to 0, the less similar



Vector space model

- sometimes used to find how close documents are to a query
- although cos ranges [-1, +1], the vector data will be positive, resulting in a cosine similarity [0, 1]



Vector space model

- Each document in a corpus is represented as a vector of numbers in a common vector space, where each element is a tf or tf-idf value for that word
- For the documents below, remove stopwords, punctuation, and lemmatize
- Each document becomes a list of tokens

a: Her favorite hobbies are reading, walking, knitting, and learning new things.
b: His favorite hobbies are reading, discussing politics, and walking the dog.

A = [favorite, hobby, read, walk, knit, learn, thing]
B = [favorite, hobby, read, discuss, politics, walk, dog]

Vector space model

- Next, find the vocabulary of the corpus, put in alpha order
- Each document will have a count in the position for each word in the vocabulary

Vocab = [discuss, dog, favorite, hobby, knit, learn, politics, read, thing, walk]

A = [favorite, hobby, read, walk, knit, learn, thing]
B = [favorite, hobby, read, discuss, politics, walk, dog]

```
A = [0, 0, 1, 1, 1, 1, 0, 1, 1, 1]
B = [1, 1, 1, 1, 0, 0, 1, 1, 0, 1]
```

Cosine similarity

- The dot product of the two vectors, normalized by the vector norms (sqrt(7))

```
A = [0, 0, 1, 1, 1, 1, 0, 1, 1, 1]
B = [1, 1, 1, 1, 0, 0, 1, 1, 0, 1]
```

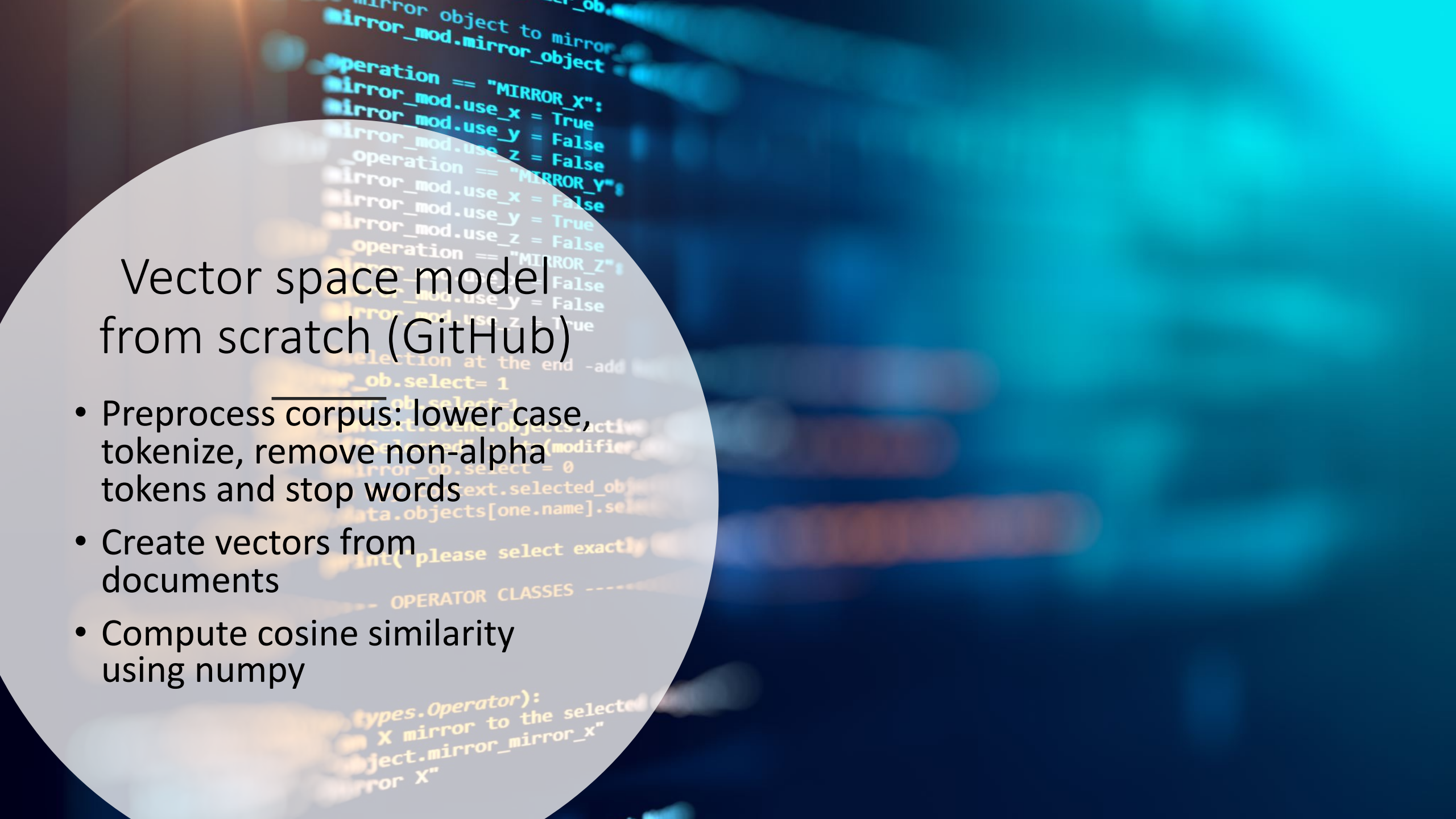
$$cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

$$\frac{4}{2.65 * 2.65} = .57$$

- The closer to 1, the more similar the documents

Cosine similarity

- Decide a cut-off point for similarity, perhaps the average over all the corpus
- Values greater than this will be “similar”
- Or select a threshold, like 0.75



Vector space model from scratch (GitHub)

- Preprocess corpus: lower case, tokenize, remove non-alpha tokens and stop words
- Create vectors from documents
- Compute cosine similarity using numpy

Preprocess

- The input text is 4 texts on different subjects
- Each document is split in half to get 8 documents

```
# preprocess: remove non-alpha, remove stopwords, lemmatize
docs_preprocessed = [[wnl.lemmatize(w) for w in doc
                      if w not in stopwords and w.isalpha()] for doc in docs]
```

Vocabulary

- Find the set of all words in the corpus

Code 14.2.1 — Building a vocabulary. Using sets and lists

```
vocab = set()
for doc in docs_preprocessed:
    doc_set = set(doc)
    vocab = vocab.union(doc_set)

vocab = sorted(list(vocab))
print('vocab length:', len(vocab))
vocab[:5]
```

```
vocab length: 3601
['abandoned', 'abdominal', 'abide', 'ability', 'able']
```

Convert docs to vecors

- Each element is the count of that vocab token in the doc
- Each vector represents the words in the same order

Code 14.2.2 — Convert documents to vectors. 'vectors' is a list of lists

```
vectors = []
for doc in docs_preprocessed:
    vec = [doc.count(t) for t in vocab]
    vectors.append(vec)
print(vectors[0][:10])
```

```
[0, 4, 0, 0, 0, 0, 0, 0, 0, 1]
```

Cosine similarity function

- Use NumPy linear algebra functions

Code 14.2.3 — Cosine similarity. Using NumPy

```
from numpy import dot
from numpy.linalg import norm

# function to compute cosine similarity
def cos_sim(v1, v2):
    return float(dot(v1, v2)) / (norm(v1) * norm(v2))
```

Calculate cosine similarity

- Results:

```
# compute cosine similarity for the first doc, paired with all docs
for i, vec in enumerate(vectors):
    print('cosine similarity anat1 and vector', i+1, '=',
              cformat(cos_sim(vectors[0], vec), '.2f'))
```

```
cosine similarity anat1 and vector 1 = 1.00
cosine similarity anat1 and vector 2 = 0.72
cosine similarity anat1 and vector 3 = 0.05
cosine similarity anat1 and vector 4 = 0.05
cosine similarity anat1 and vector 5 = 0.06
cosine similarity anat1 and vector 6 = 0.06
cosine similarity anat1 and vector 7 = 0.06
cosine similarity anat1 and vector 8 = 0.10
```

Vector space model with sklearn

- Using the same docs, vectorize the docs
- sklearn's fit() and transform() methods are combined

Code 14.3.1 — Vectorizing docs. Using sklearn

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()

docs2 = [' '.join(docs_preprocessed[i]) for i in range(len(docs))]

tfidf_docs = tfidf_vectorizer.fit_transform(docs2)
tfidf_docs.shape
```

Compute cosine similarity

- ‘tfidf_docs.shape’ is 8 x 3593
- 8 docs; 3593 elements in each vector
- Similar results to “from scratch” code that used tf

Code 14.3.2 — Cosine similarity. Using tf-idf vectors

```
from sklearn.metrics.pairwise import cosine_similarity

cosine_similarity(tfidf_docs[0], tfidf_docs)
```

```
array([[1.        , 0.70338879, 0.02614208, 0.0192754 , 0.02797854,
        0.02504257, 0.0273924 , 0.05087871]])
```

Essential points to note

Applications of the vector space model:

- Document clustering
- Information retrieval
- Sentiment analysis
- Chatbot dialog: extract most relevant response from a corpus
- Plagiarism detection
- Author attribution

To Do

- Quiz IE and more
- No homework on this material