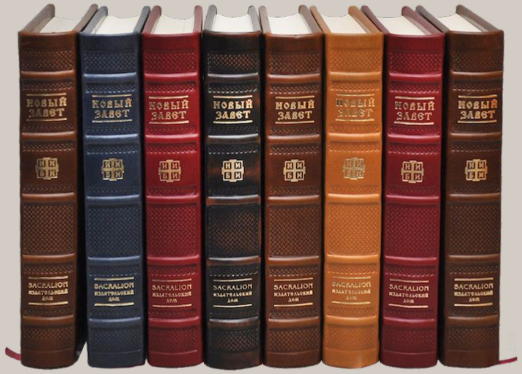
BookEnder

A book recommender

By Veena Kumar

Data

- Original dataset contains plot summaries for 16,559 books (taken from Wikipedia), author, and genre(s)
- Final data: **8,825 books**



Model

1. **Tokenize** words from book summaries
2. Run every word in each document through **word2vec's** model to get a **score** for the word
   - 3,000,000 pre-trained words from Google News dataset
   - Vectors are 300 dimensions (i.e 'features') x # of words
3. Take **mean** of every dimension
   - Vectors become 300 x 1 for each document
4. Find **cosine similarity** between documents
5. Donezo! Almost…

Model

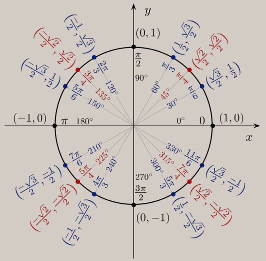
1. **Tokenize** words from book summaries
2. Run every word in each document through Google **word2vec's** model to get a **score** for the word
   - 3,000,000 pre-trained words from Google News dataset
   - Vectors are 300 dimensions (i.e 'features') x # of words
3. Take **mean** of every dimension
   - Vectors become 300 x 1 for each document
4. Find **cosine similarity** between documents
5. Donezo! Almost…

Model

1. **Tokenize** words from book summaries
2. Run every word in each document through Google **word2vec's** model to get a **score** for the word
   - 3,000,000 pre-trained words from Google News dataset
   - Vectors are 300 dimensions (i.e 'features') x # of words
3. Take **mean** of every dimension
   - Vectors become 300 x 1 for each document
4. Find **cosine similarity** between documents
5. Donezo! Almost…

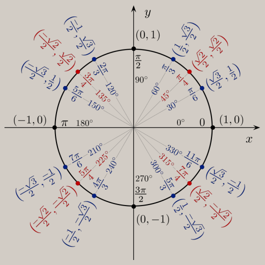
Model

1. **Tokenize** words from book summaries
2. Run every word in each document through **word2vec's** model to get a **score** for the word
    - 3,000,000 pre-trained words from Google News dataset
    - Vectors are 300 dimensions (i.e 'features') x # of words
3. Take **mean** of every dimension
    - Vectors become 300 x 1 for each document
4. Find **cosine similarity** between documents
5. Donezo! Almost...

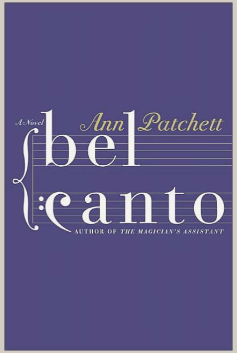


Model

1. **Tokenize** words from book summaries
2. Run every word in each document through **word2vec's** model to get a **score** for the word
   - 3,000,000 pre-trained words from Google News dataset
   - Vectors are 300 dimensions (i.e 'features') x # of words
3. Take **mean** of every dimension
   - Vectors become 300 x 1 for each document
4. Find **cosine similarity** between documents
5. Donezo! Almost…

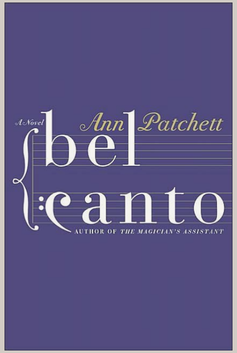


Example



- ❖ **Book:** Bel Canto
- ❖ **Genres:** Thriller, Fiction, Romance
- ❖ **Summary of the summary:** A terrorist organization attempts to assassinate the President during a party thrown by the Vice President, however the President was not in attendance. The terrorists decide instead to take important guests hostage in order to leverage a ransom. The rest of the story is about the romances that develop during the hostage situation.

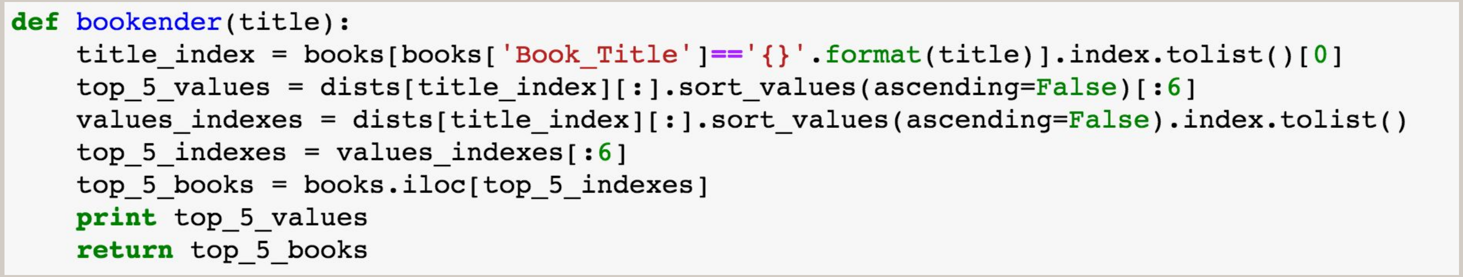
Example



- ❖ Enter book title

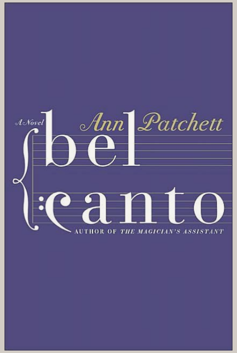
bookender(‘Bel Canto’)

- ❖ Function



```
def bookender(title):
    title_index = books[books['Book_Title']=='{}'.format(title)].index.tolist()[0]
    top_5_values = dists[title_index][:].sort_values(ascending=False)[:6]
    values_indexes = dists[title_index][:].sort_values(ascending=False).index.tolist()
    top_5_indexes = values_indexes[:6]
    top_5_books = books.iloc[top_5_indexes]
    print top_5_values
    return top_5_books
```

Example



Results

Scores
1.000000
0.887298
0.883713
0.883078
0.882075
0.881482

Book Title	Author	Genre
Bel Canto	Ann Patchett	Thriller, Fiction, Romance
We the Living	Ayn Rand	Autobiography, Historical
Omerta	Mario Puzo	Thriller, Crime Fiction
Farnham’s Freehold	Robert A. Heinlein	Science Fiction
Twilight at the Well of Souls	Jack L. Chalker	Science Fiction,
Becoming Madame Mao	Anchee Min	Historical

Future Enhancements

- Add extra layer to match by genre
- Sentiment analysis
- Website!
  - Links to purchase book
  - Reviews
  - Scrape Wikipedia to obtain summary, if book is not in database

References

Data:

- David Bamman and Noah Smith (2013), "New Alignment Methods for Discriminative Book Summarization"

Google’s word2vec:

- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS, 2013.
- Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. In Proceedings of NAACL HLT, 2013.

Questions?