We will see that how we can perform Singular Value Decomposition of some book titles we are having in our dataset using TruncatedSVD.

This transformer performs linear dimensionality reduction by means of truncated singular value decomposition (SVD).

Contrary to PCA, this estimator does not center the data before computing the singular value decomposition. This means it can work with sparse matrices efficiently. When we perform SVD (Singular Value Decomposition) on text data it is also called LSA (Latent Semantic Analysis).

Step 1 – Importing libraries required for Singular Value Decomposition.

```
import nltk
from nltk.stem import WordNetLemmatizer
import numpy as np
from sklearn.decomposition import TruncatedSVD
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline

nltk.download('punkt')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

Step 2 – Reading lines from our text file.

```
titles = [line.strip() for line in open('/content/all_book_titles (1).txt')]
titles
```

```
'Flashback: A Brief Film History',
'The Greeks: History, Culture, and Society',
'History of Italian Renaissance Art (Paper cover) (7th Edition) (Mysearchlab Series for Art)',
'Photography: A Cultural History (3rd Edition) (Mysearchlab Series for Art Mysearchlab Series for Art)',
'Latin America : An Interpretive History',
"The World's History (4th Edition)",
'Consider the Source : Documents in Latin American History - An Interpretive History',
'Sociology of Religion: A Reader',
'Anthropology of Religion, Magic, and Witchcraft, The (3rd Edition)',
'Ultimate Questions: Thinking about Philosophy',
'Seeing Ourselves: Classic, Contemporary, and Cross-Cultural Readings in Sociology',
'Anthropology (13th Edition) (MyAnthroLab Series)',
'Art History, Volume 1 (4th Edition) (MyArtsLab Series)',
'Art History, Volume 2 (4th Edition) (MyArtsLab Series)',
'Art History, Combined (4th Edition) (MyArtsLab Series)',
'Women, Politics, and American Society (5th Edition) (Longman Classics in Political Science)',
'Political Science: An Introduction',
'Look! Art History Fundamentals',
'Film: A Critical Introduction (3rd Edition) (MyCommunicationKit Series)',
'THINK Sociology',
'THINK Human Sexuality',
'Other World, The (9th Edition) (Mysearchlab Series for Political Science)',
'Cultural Anthropology in a Globalizing World (3rd Edition)',
'Introduction to Forensic Anthropology (4th Edition)',
'Empirical Political Analysis (8th Edition) (Mysearchlab Series for Political Science)',
'Challenge of Third World Development, The (6th Edition) (Mysearchlab Series for Political Science)',
"Janson's History of Art: The Western Tradition with MyArtsLab and Pearson eText (8th Edition)",
'Social Psychology',
'Statistics for the Behavioral and Social Sciences',
'Sociology of Education, The (7th Edition)',
'Conscious Reader',
'GenderSpeak: Personal Effectiveness in Gender Communication',
"Political Science Student Writer's Manual",
'Thinking about Women : Sociological Perspectives on Sex and Gender',
'Understanding Psychology (10th Edition)',
'Understanding the Political World: A Comparative Introduction to Political Science Plus MyPoliSciLab -- Access Card Package with eText -- Access Card Package (11th Edition)',
'Psychology and Life (20th Edition)',
'Art History (5th Edition)',
'Art History Volume 1 (5th Edition)',
'Art History Portable Book 1 (5th Edition)',
'Art History Portables Book 2 (5th Edition)',
'Art History Portables Book 3 (5th Edition)',
'Art History Portables Book 6 (5th Edition)',
'Art History Volume 2 (5th Edition)',
'Religions of the World Plus NEW MyReligionLab with eText -- Access Card Package (12th Edition)',
'Art Beyond the West (3rd Edition)',
'Essentials of Sociology: A Down-to-Earth Approach Plus NEW MySocLab with eText -- Access Card Package (10th Edition)',
'Essentials of Sociology: A Down-to-Earth Approach (10th Edition)',
'Thinking About Women: Sociological Perspectives on Sex and Gender (10th Edition)',
'Literature and the Writing Process (10th Edition)',
'Exploring Biological Anthropology: The Essentials (3rd Edition)',
'Human Sexuality in a World of Diversity (case) (9th Edition)',
"The World's Religions (4th Edition)",
'Forty Studies that Changed Psychology (7th Edition)',
'Human Evolution and Culture: Highlights of Anthropology Plus NEW MyAnthroLab with Pearson eText -- Access Card Package (7th Edition)',
'Music for Sight Singing (9th Edition)',
'Art History Volume 1, Books a la Carte Edition (5th Edition)',
'Art History volume 1, Books a la Carte Plus NEW MyArtsLab with eText -- Access Card Package (5th Edition)',
```

Step 3 – Creating a Stopwords set.

```
stopwords = set(word.strip() for word in open('/content/stopwords (1).txt'))
stopwords = stopwords.union({
    'introduction', 'edition', 'series', 'application',
    'approach', 'card', 'access', 'package', 'plus', 'etext',
    'brief', 'vol', 'fundamental', 'guide', 'essential', 'printed',
    'third', 'second', 'fourth', })

word_lemmatizer = WordNetLemmatizer()
```

We have our default stopwords like a, is, that, this, etc. in stopwords.txt. So first of all we are loading all those in a variable called stopwords. Secondly, we are adding some more stopwords like edition, introduction, series, etc. manually in the stopwords set. These are some very common words that occur in Book Titles. Lastly we are initializing WordNetLemmatizer().

Step 4 – Creating tokenizer function.

```
def tokenizer(s):
    s = s.lower()
    tokens = nltk.tokenize.word_tokenize(s)
    tokens = [t for t in tokens if len(t)>2]
    tokens = [word_lemmatizer.lemmatize(t) for t in tokens]
    tokens = [t for t in tokens if t not in stopwords]
    tokens = [t for t in tokens if not any(c.isdigit() for c in t)]
    return tokens
```

This will take a string, convert it to lowercase, tokenize it, remove words having a length less than 2, lemmatize words, remove stopwords and finally remove all those words having any digit in them.

Step 5 – Checking tokenizer.

```
tokenizer('my name is abhishek and i am 19 years old!!')
```

```
['name', 'abhishek']
```

Step 6 – Creating word_2_int and int_2_word dictionaries.

```
word_2_int = {}
int_2_words = {}
ind = 0
error_count = 0

for title in titles:
    try:
        title = title.encode('ascii', 'ignore').decode('utf-8') # this will throw exception if bad characters
        tokens = tokenizer(title)
        for token in tokens:
            if token not in word_2_int:
                word_2_int[token] = ind
                int_2_words[ind]=token
                ind += 1
    except Exception as e:
        print(e)
        print(title)
        error_count += 1
```

We have used try-except because there could be some titles that have some special characters. Those will throw exceptions. Then we simply take the title and tokenize it and then simply traverse in those tokens. If the token is not in our vocabulary (word_2_int), append it and give a token number to it. Similarly, create an apposite dictionary int_2_word also for future use. And then simply increment the index.

Step 7 – Creating tokens_2_vectors function.

```
def tokens_2_vectors(tokens):
    X = np.zeros(len(word_2_int))
    for t in tokens:
        try:
            index = word_2_int[t]
            X[index]=1
        except:
            pass
    return X
```

This function simply converts a title in the token form to a vector. It will create an array of our vocabulary size with all elements as 0. It will then replace 0s with 1s for the words that are present in the title whose vector we are creating.

Step 8 – Creating a final matrix and fitting it into our SVD.

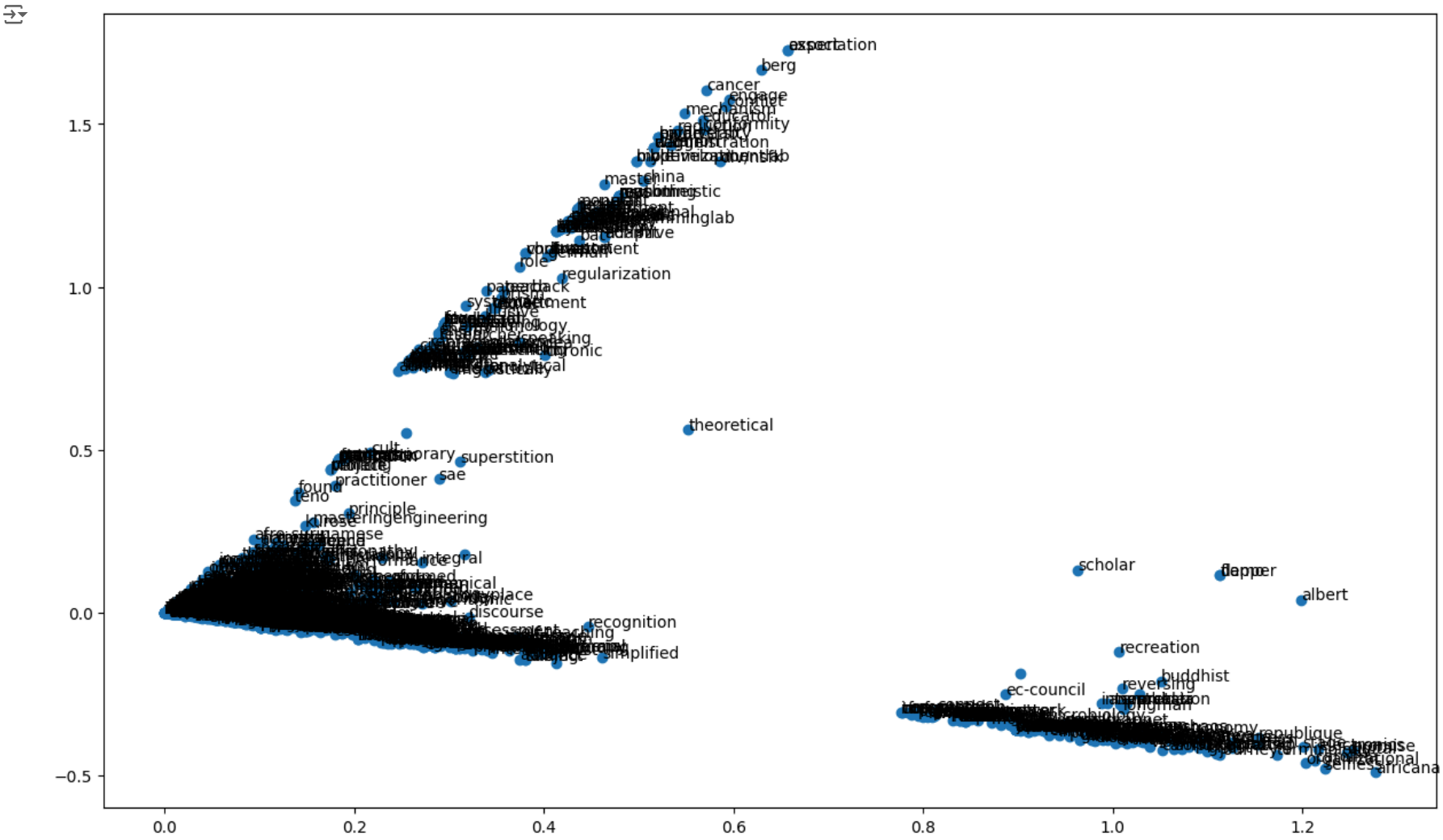
```
final_matrix = np.zeros((len(titles),len(word_2_int)))

for i in range(len(titles)):
    title = titles[i]
    token = tokenizer(title)
    final_matrix[i,:] = tokens_2_vectors(token)

svd = TruncatedSVD()
Z = svd.fit_transform(final_matrix)
Z.shape
```

```
(2373, 2)
```

Here we are creating a final matrix with rows as no of titles (2373) and columns as no of words in our vocabulary. Then we are filling this matrix with vectors of each and every title. Finally fitting our TruncatedSVD with our final matrix. It means that our data is having 2373 titles and 2 represents the position of that token in the plane.

Step 9 – Visualize the results.

```
fig = plt.figure(figsize=(15,9))
plt.scatter(Z[:,0],Z[:,1])
for i in range(len(word_2_int)):
    plt.annotate(int_2_words[i],(Z[i,0],Z[i,1]))
```



You will see that similar words will be closer in this plot.

You will see that similar words will be closer in this plot.