```
!pip install scikit-learn
```

```
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.23.5)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.2.0)
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
import pandas as pd
import gzip
import nltk
```

```
import math
import sklearn
```

```
nltk.download('stopwords')
from nltk import RegexpTokenizer
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
tokenizer = RegexpTokenizer(r"\w+")
stop = stopwords.words('english')
```

```
with gzip.open('Software.json.gz') as rf:
  data = pd.read_json(rf, lines=True, dtype=str)
```

```
data.head()
```

	overall	verified	reviewTime	reviewerID	asin	style	reviewerName	reviewText	summary	unixReviewTime	vote	image
0	4.0	True	03 11, 2014	A240ORQ2LF9LUI	0077613252	{'Format:': ' Loose Leaf'}	Michelle W	The materials arrived early and were in excell...	Material Great	1394496000	nan	nar
								I am really				

```
data_stripped = data[['overall', 'reviewText']]
data_stripped.head()
```

	overall	reviewText
0	4.0	The materials arrived early and were in excell...
1	4.0	I am really enjoying this book with the worksh...
2	1.0	IF YOU ARE TAKING THIS CLASS DON"T WASTE YOUR ...
3	3.0	This book was missing pages!!! Important pages...
4	5.0	I have used LearnSmart and can officially say ...

```
data_stripped['reviewText'].fillna('was nan')
data_stripped.isna().any()
```

```
overall       False
reviewText    False
dtype: bool
```

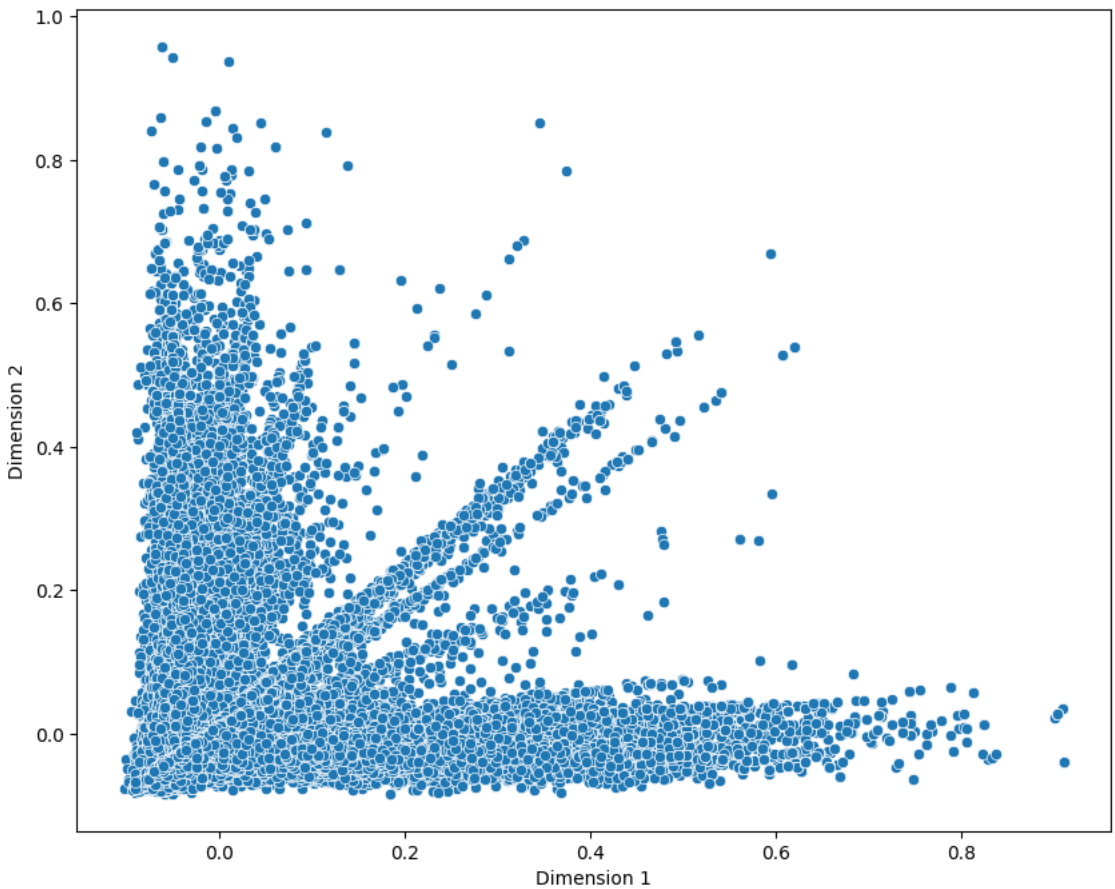
```
reviews = data_stripped['reviewText'].to_numpy()
print(reviews[:2])
```

```
["The materials arrived early and were in excellent condition.  However for the money spent they really should've come with a binder an
 'I am really enjoying this book with the worksheets that make you review your goals, what to do when you do not make it, it reminds me
```

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer

vectorizer = CountVectorizer(max_df=0.70, stop_words='english')
vectorized = vectorizer.fit_transform(reviews)
tfidf = TfidfTransformer()
tfidf_mat = tfidf.fit_transform(vectorized)
```

```
from sklearn.decomposition import TruncatedSVD
lsa = TruncatedSVD(n_components=100)
features = lsa.fit_transform(tfidf_mat)
```

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2).fit_transform(features)

# Create a DataFrame for visualization
plot_df = pd.DataFrame(data={'Dimension 1': pca[:, 0], 'Dimension 2': pca[:, 1]})

# Visualize the data
plt.figure(figsize=(10, 8))
sns.scatterplot(x='Dimension 1', y='Dimension 2', data=plot_df)
plt.show()
```

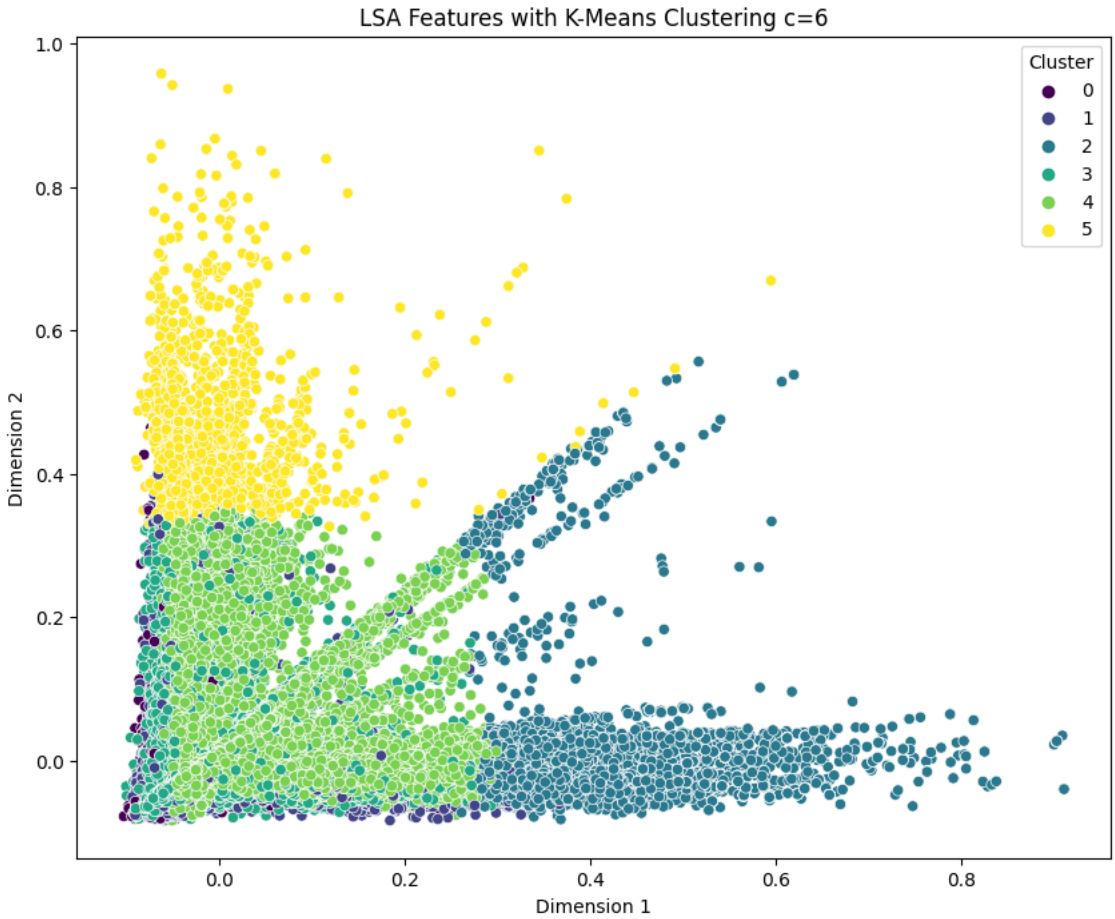



```
from sklearn.cluster import KMeans

# Clustering using K-Means
num_clusters = 6
kmeans = KMeans(n_clusters=num_clusters, random_state=42)
cluster_labels = kmeans.fit_predict(features)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 1
  warnings.warn(
```

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
```

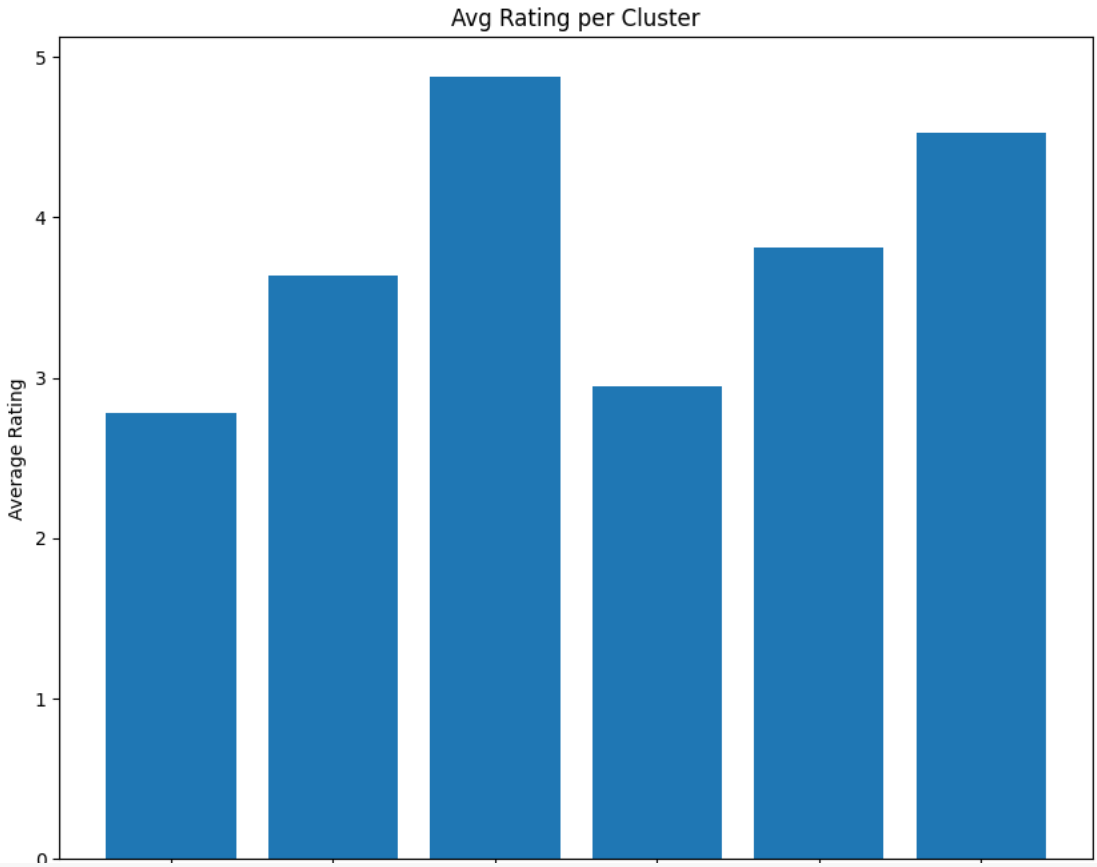
```
# Create a DataFrame for visualization
plot_df = pd.DataFrame(data={'Dimension 1': pca[:, 0], 'Dimension 2': pca[:, 1], 'Cluster': cluster_labels})

# Visualize the data
plt.figure(figsize=(10, 8))
sns.scatterplot(x='Dimension 1', y='Dimension 2', hue='Cluster', data=plot_df, palette='viridis')
plt.title(f'LSA Features with K-Means Clustering c={num_clusters}')
plt.show()
```




```
cluster0 = [reviews[i] for i in range(len(cluster_labels)) if cluster_labels[i] == 0]
cluster1 = [reviews[i] for i in range(len(cluster_labels)) if cluster_labels[i] == 1]
cluster2 = [reviews[i] for i in range(len(cluster_labels)) if cluster_labels[i] == 2]
```

```
cluster_ratings = [0] * num_clusters
cluster_counts = [0] * num_clusters
ratings = data_stripped['overall'].to_numpy()

for i, x in enumerate(cluster_labels):
  cluster_ratings[x] += float(ratings[i])
  cluster_counts[x] += 1

for i in range(num_clusters):
  cluster_ratings[i] /= cluster_counts[i]
```

```
plt.figure(figsize=(10, 8))
plt.bar(['Cluster 1', 'Cluster 2', 'Cluster 3', 'Cluster 4', 'Cluster 5', 'Cluster 6'], cluster_ratings)
plt.title('Avg Rating per Cluster')
plt.ylabel('Average Rating')
plt.show()
```




```
my_review = "This product was absolutely horrible. I never want to buy it again!"

cluster = kmeans.predict(lsa.transform(tfidf.transform(vectorizer.transform([my_review]))))
```

```
cluster
```

```
array([4], dtype=int32)
```