```
import numpy as np
import pandas as pd
from sklearn.feature_extraction import text
from sklearn.metrics.pairwise import linear_kernel


#Load the data
from google.colab import files
files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving books.csv to books.csv
{'books.csv': b'S.No,Novel Name,Author,year,Rating ,Publisher\n1,The
Pilgrim\'s Progress,John Bunyan,1678,92,Peacock books\n2,Robinson
Crusoe,Danial Defoe,1719,84,William Taylor\n3,Gulliver\'s Travels ,Jonathan
Swift,1726,80,Benjamin Motte\n4,Clarissa ,Samuel
Richordson,1748,72,\xe2\x80\x8ePenguin Classics\n5,Tom Jones ,Henry
Fielding,1749,79,\xe2\x80\x8eWordsworth Editions Ltd (5 May 1992)\n6," The
life and opinions of Tristman Shandy, Gentleman ",Laurence
sterene,1759,71,\xe2\x80\x8eOxford University Press; Revised edition (4
February 2010)\n7,Emma ,Jane Austen,1816,75,John Murray\n8,Frankenstein
,Mary shelliey,1818,83,"Lackington, Hughes, Harding, Mavor &
Jones"\n9,Sybill,Benjamin Disraeli,1845,69,"\xe2\x80\x8eHenry Regnery
Company Chicago; First Edition (January 1, 1973)"\n10,Jane Eyre,Charlotte
Bronte,1847,88,Currer Bell\n'}
```

```
#store the data
df =pd.read_csv('books.csv',encoding='unicode_escape',error_bad_lines=False)
```

```
<ipython-input-6-8cb23f0748d3>:2: FutureWarning: The error_bad_lines argument h
```

```
df =pd.read_csv('books.csv',encoding='unicode_escape',error_bad_lines=False)
```

```
#Show the data
df
```

	S.No	Novel Name	Author	year	Rating	Publisher
0	1	The Pilgrim's Progress	John Bunyan	1678	92	Peacock books
1	2	Robinson Crusoe	Danial Defoe	1719	84	William Taylor
2	3	Gulliver's Travels	Jonathan Swift	1726	80	Benjamin Motte
3	4	Clarissa	Samuel Richordson	1748	72	â□□Penguin Classics
4	5	Tom Jones	Henry Fielding	1749	79	â□□Wordsworth Editions Ltd (5 May 1992)
5	6	The life and opinions of Tristman Shandy, Gen...	Laurence sterene	1759	71	â□□Oxford University Press; Revised edition (4...
6	7	Emma	Jane Austen	1816	75	John Murray
7	8	Frankenstein	Mary shelliey	1818	83	Lackington, Hughes, Harding, Mavor & Jones
8	9	Sybill	Benjamin Disraeli	1845	69	â□□Henry Regnery Company Chicago; First Editio...
9	10	Jane Eyre	Charlotte Bronte	1847	88	Currer Bell

```
#Create a list of columns to keep
Columns =['Novel Name','Author','Publisher']


#Create a function to create the important Columns/features
def Combine_features(data):
  features = []
```

```
  for i in range(0, data.shape[0]):
    features.append(data['Novel Name'][i] + ''+data['Author'][i] + '' + data['Publisher'][


  return features


#Create a column to store the Combined features
df["Combine_features"]=Combine_features(df)

#Show the updated data
print(df)
```

	S.No	Novel Name	Author	year	Rating	Publisher	Combine_features
0	1	The Pilgrim's Progress	John Bunyan	1678	92	Peacock books	The Pilgrim's ProgressJohn BunyanPeacock books
1	2	Robinson Crusoe	Danial Defoe	1719	84	William Taylor	Robinson CrusoeDanial DefoeWilliam Taylor
2	3	Gulliver's Travels	Jonathan Swift	1726	80	Benjamin Motte	Gulliver's Travels Jonathan SwiftBenjamin Motte
3	4	Clarissa	Samuel Richordson	1748	72	â□□Penguin Classics	Clarissa Samuel Richordsonâ□□Penguin Classics
4	5	Tom Jones	Henry Fielding	1749	79	â□□Wordsworth Editions Ltd (5 May 1992)	Tom Jones Henry Fieldingâ□□Wordsworth Editions...
5	6	The life and opinions of Tristman Shandy, Gen...	Laurence sterene	1759	71	â□□Oxford University Press; Revised edition (4...	The life and opinions of Tristman Shandy, Gen...
6	7	Emma	Jane Austen	1816	75	John Murray	Emma Jane AustenJohn Murray
7	8	Frankenstein	Mary shelliey	1818	83	Lackington, Hughes, Harding, Mavor & Jones	Frankenstein Mary shellieyLackington, Hughes, ...
8	9	Sybill	Benjamin Disraeli	1845	69	â□□Henry Regnery Company Chicago; First Editio...	SybillBenjamin Disraeliâ□□Henry Regnery Compan...
9	10	Jane Eyre	Charlotte Bronte	1847	88	Currer Bell	Jane EyreCharlotte BronteCurrer Bell

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer


#convert the text from the new column to a matrix of word counts using CountVectorizer
Vectorizer = CountVectorizer()
CM = CountVectorizer().fit_transform(df['Combine_features'])
```

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity



#Get the cosine_similarity Matrix from the Count matrix
CS = cosine_similarity(CM)
#print the Scores
print(CS)
```

```
      [[1.         0.         0.         0.         0.         0.10846523
        0.         0.         0.         0.        ]
       [0.         1.         0.         0.         0.         0.
        0.         0.         0.         0.        ]
       [0.         0.         1.         0.         0.         0.
        0.         0.         0.         0.        ]
       [0.         0.         0.         1.         0.         0.
        0.         0.         0.         0.        ]
       [0.         0.         0.         0.         1.         0.
        0.         0.12598816 0.10540926 0.        ]
       [0.10846523 0.         0.         0.         0.         1.
        0.         0.         0.0766965  0.        ]
       [0.         0.         0.         0.         0.         0.
        1.         0.         0.         0.25      ]
       [0.         0.         0.         0.         0.12598816 0.
        0.         1.         0.         0.        ]
       [0.         0.         0.         0.         0.10540926 0.0766965
        0.         0.         1.         0.        ]
       [0.         0.         0.         0.         0.         0.
        0.25       0.         0.         1.        ]]
```

```
#Get the Novel Name of the book the reader likes
novel_Name = df['Novel Name'][1]
#Show the Novel Name
print(novel_Name)
```

```
      Robinson Crusoe
```

```
#Find the year of the book that the user likes
book_year = df.get('book_year', 1)

#Show the book year
print(book_year)
```

```
      1
```

```
#Create a list of tuples in the form(book_year, similarity score)
scores = list(enumerate(CS[book_year]))
print(scores)
```

```
      [(0, 0.0), (1, 1.0), (2, 0.0), (3, 0.0), (4, 0.0), (5, 0.0), (6, 0.0), (7, 0.0)
```

```
#Sort the list of similar books in descending order
sorted_scores = sorted(scores, key= lambda x:x[1], reverse= True)
```

```
sorted_scores = sorted_scores[1:]
#show the sorted scores
sorted_scores
```

```
[(0, 0.0),
 (2, 0.0),
 (3, 0.0),
 (4, 0.0),
 (5, 0.0),
 (6, 0.0),
 (7, 0.0),
 (8, 0.0),
 (9, 0.0)]
```

```
#Create a new column called 'book_year'
book_year = [1810, 1811, 1812, 1813, 1814, 1815]
```

```
#Find the year of the book that the user likes
book_year = ('book_year', 1814)

#Show the book year
print(book_year)
```

```
('book_year', 1814)
```

```
#create a loop to print the first 5 books from the sorted list
j = 0
print('The 5 most recommended books are:\n')
for book_year in sorted_scores[:5]:
    print(j + 1, book_year)
    j += 1
    if j >= 5:
        break
print("End of loop")
```

```
The 5 most recommended books are:

1 (0, 0.0)
2 (2, 0.0)
3 (3, 0.0)
4 (4, 0.0)
5 (5, 0.0)
End of loop
```