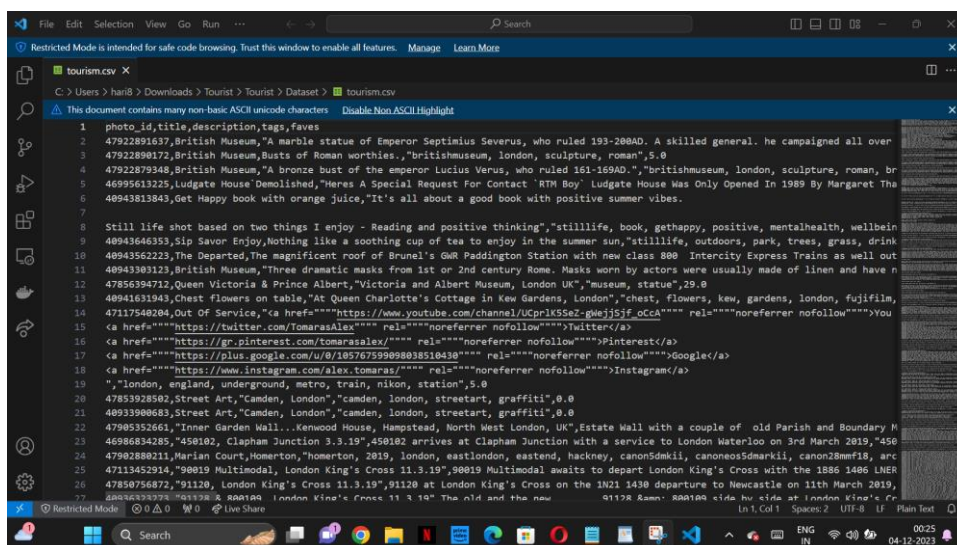


Analysing Tourist Behaviour using Big Data Technology

In this project we are implementing Popular Tourist Place Recommendation using Bigdata framework called SPARK. Spark can process data in distributed manner so it can handle any size of data. In propose work we are using Geo Tagged images dataset to extract places information and then this text data will be cluster to group similar behaviour tourist into same cluster. Whenever new user input any query then clustering algorithm will predict similar cluster as per user query and then suggest top 5 popular visited similar interest places as recommendation.

To cluster user behaviour we have used KMEANS algorithm and then the user features and cluster label will get trained with Random Forest Classifier to explain model and for explanation we have used SHAP framework instead of LIME. This model will explain which features contribute most to predict particular label.

For this project we have used below tourism data which is Geo Tagged using FLICKER

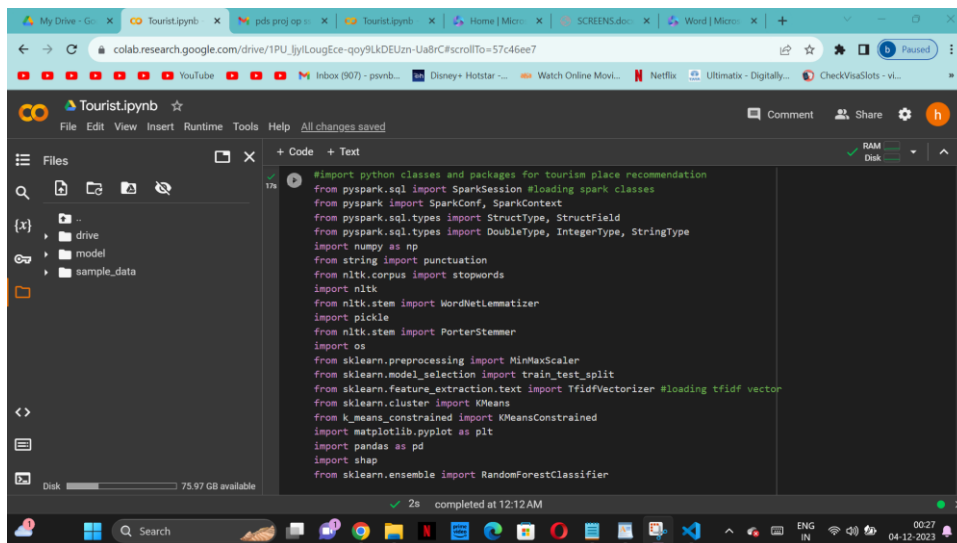


In above dataset first row represents column names and remaining rows represents dataset values and in columns we have names as “Photo ID, Description, tags, favourites as Number of time visited etc.”. So by using above dataset we will cluster and recommend places for new user.

SCREEN SHOTS

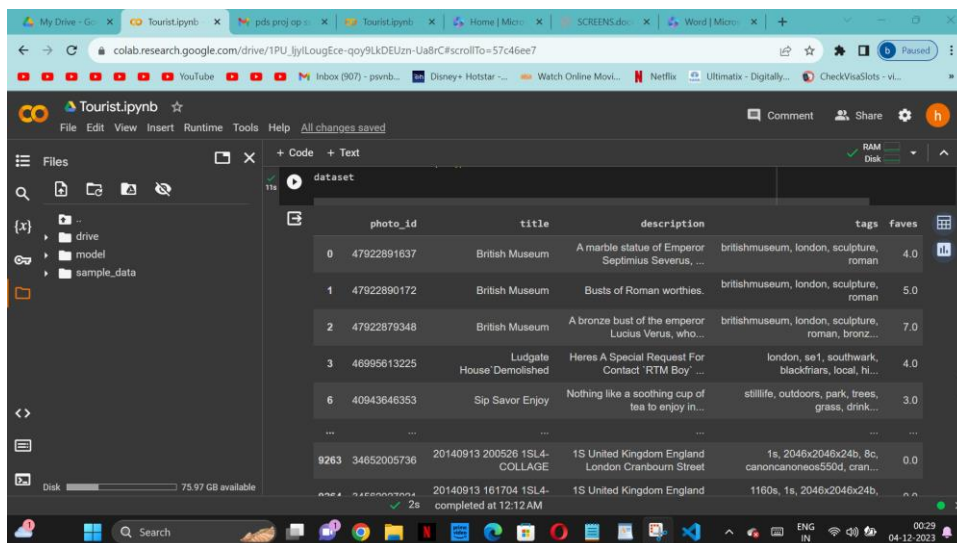
We have coded this project using GOOGLE colab and below are the code and output screens with comments

In below screen importing spark, NLP (natural language processing API to remove stop words, special symbols from geo tag text dataset) and then importing KMEANS and other classes



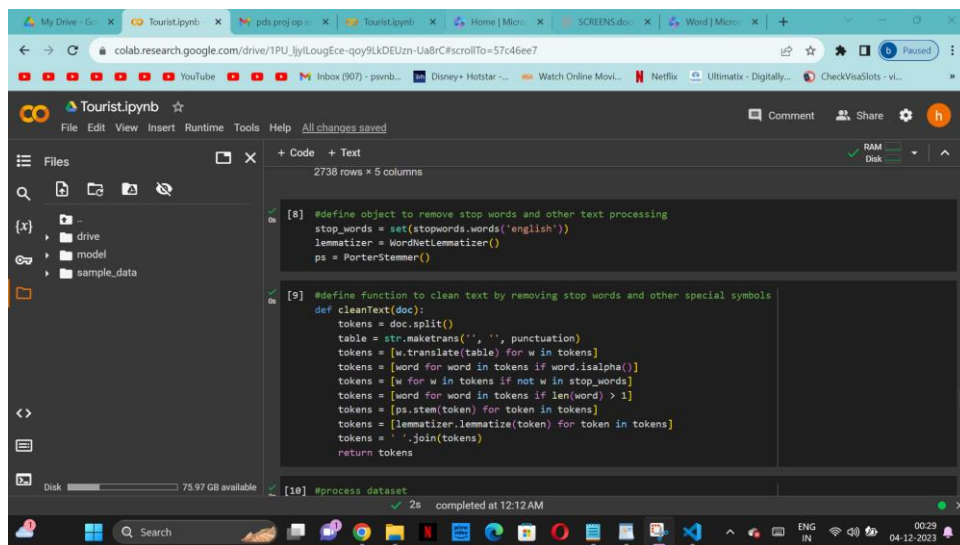
```
#import python classes and packages for tourism place recommendation
from pyspark.sql import SparkSession #loading spark classes
from pyspark import SparkConf, SparkContext
from pyspark.sql.types import StructType, StructField
from pyspark.sql.types import DoubleType, IntegerType, StringType
import numpy as np
from string import punctuation
from nltk.corpus import stopwords
import nltk
from nltk.stem import WordNetLemmatizer
import pickle
from nltk.stem import PorterStemmer
import os
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer #loading tfidf vector
from sklearn.cluster import KMeans
from k_means_constrained import KMeansConstrained
import matplotlib.pyplot as plt
import pandas as pd
import shap
from sklearn.ensemble import RandomForestClassifier
```

In below screen using Spark we are loading Tourism dataset and after loading will get below output



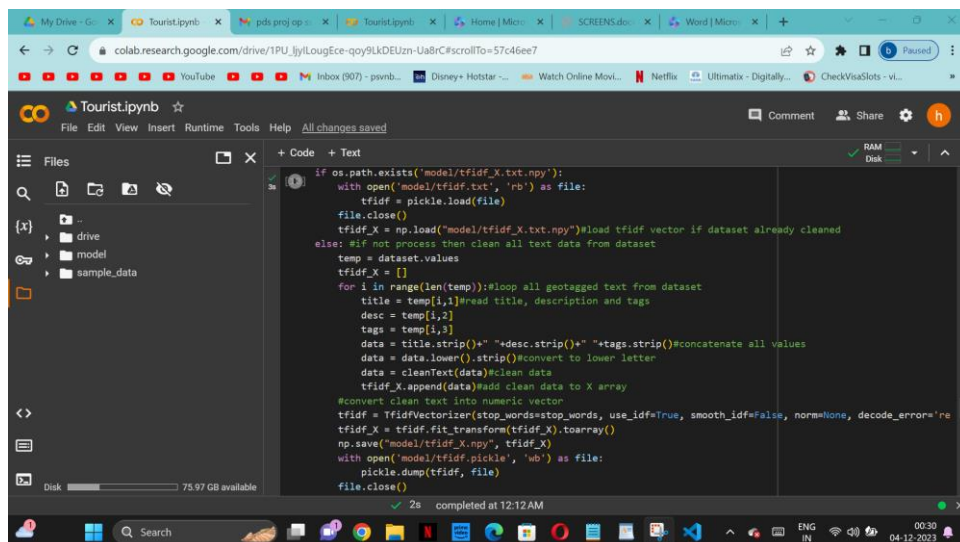
	photo_id	title	description	tags	faves
0	47922891637	British Museum	A marble statue of Emperor Septimius Severus, ...	britishmuseum, london, sculpture, roman	4.0
1	47922890172	British Museum	Busts of Roman worthies.	britishmuseum, london, sculpture, roman	5.0
2	47922879348	British Museum	A bronze bust of the emperor Lucius Verus, who...	britishmuseum, london, sculpture, roman, bronz...	7.0
3	46995613225	Ludgate House Demolished	Here's A Special Request For Contact 'RTM Boy' ...	london, se1, southwark, blackfriars, local, hi...	4.0
6	40943646353	Sip Savor Enjoy	Nothing like a soothing cup of tea to enjoy in...	stillife, outdoors, park, trees, grass, drink...	3.0
...
9263	34652005736	20140913 200526 1SL4-COLLAGE	1S United Kingdom England London Cranbourn Street	1s, 2046x2046x24b, 8c, canoncanoneos550d, cran...	0.0
...
...	20140913 161704 1SL4-	20140913 161704 1SL4-	1S United Kingdom England	1160s, 1s, 2046x2046x24b,	...

In below screen we can see loaded dataset values



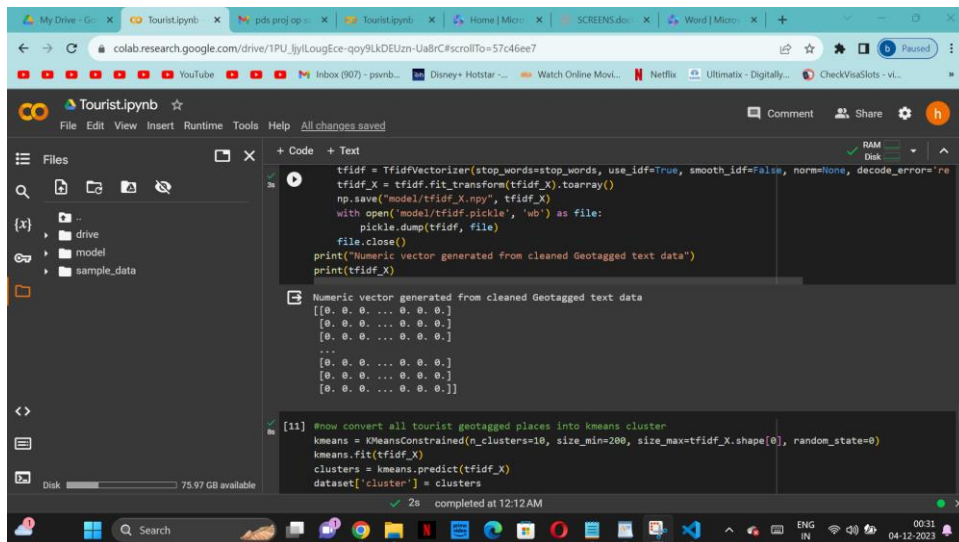
The screenshot shows a Google Colab notebook interface. The top bar indicates the notebook is named 'Tourist.ipynb' and is located in a directory. The left sidebar shows a file explorer with folders 'drive', 'model', and 'sample_data'. The main code area contains two code cells. The first cell, labeled '[8]', defines objects for removing stop words and other text processing: `stop_words = set(stopwords.words('english'))`, `lemmatizer = WordNetLemmatizer()`, and `ps = PorterStemmer()`. The second cell, labeled '[9]', defines a function `cleanText(doc)` that splits text into tokens, removes punctuation, filters out non-alphabetic words, removes stop words, and lemmatizes the remaining words. The output of the second cell shows a dataset with 2738 rows and 5 columns. The bottom status bar indicates the notebook is completed at 12:12 AM.

In below screen defining function to clean geo tagged text data



The screenshot shows a Google Colab notebook interface. The top bar indicates the notebook is named 'Tourist.ipynb' and is located in a directory. The left sidebar shows a file explorer with folders 'drive', 'model', and 'sample_data'. The main code area contains a single code cell. The code defines a function `cleanText(doc)` that splits text into tokens, removes punctuation, filters out non-alphabetic words, removes stop words, and lemmatizes the remaining words. The function is then used to process the dataset. The output of the code cell shows a dataset with 2738 rows and 5 columns. The bottom status bar indicates the notebook is completed at 12:12 AM.

In below screen reading all Geo Tagged text data and then converting all clean text data into numeric TFIDF vector and this vector contains average frequency of each words and if word does not contains then vector will have 0 and by using this vector KMEANS will perform clustering



The screenshot shows a Jupyter Notebook interface with the following code and output:

```
tfidf = TfidfVectorizer(stop_words=stop_words, use_idf=True, smooth_idf=False, norm=None, decode_error='replace')
tfidf_X = tfidf.fit_transform(tfidf_X).toarray()
np.save("model/tfidf_X.npy", tfidf_X)
with open("model/tfidf.pickle", 'wb') as file:
    pickle.dump(tfidf, file)
file.close()
print("Numeric vector generated from cleaned Geotagged text data")
print(tfidf_X)
```

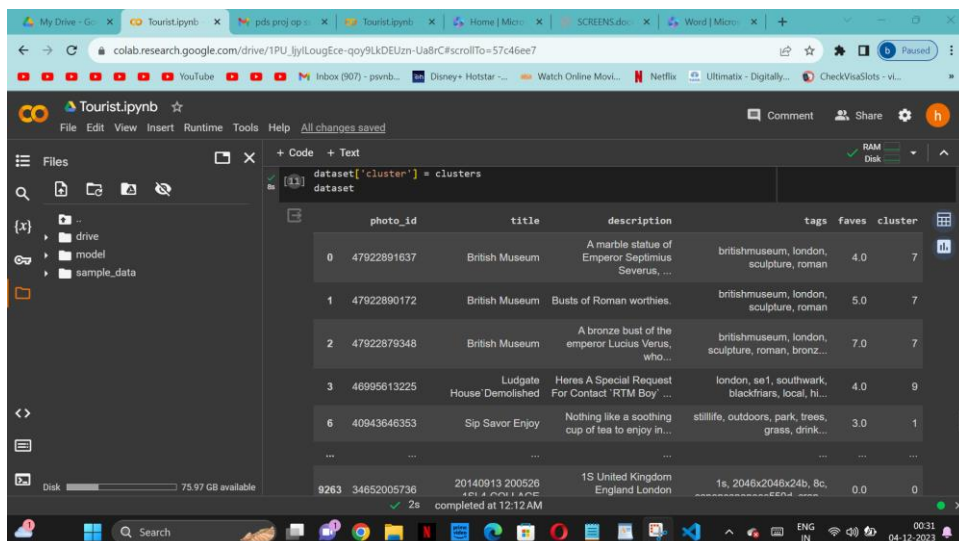
Output:

```
[[0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]]
```

```
[11] #now convert all tourist geotagged places into kmeans cluster
kmeans = KMeansConstrained(n_clusters=10, size_min=200, size_max=tfidf_X.shape[0], random_state=0)
kmeans.fit(tfidf_X)
clusters = kmeans.predict(tfidf_X)
dataset['cluster'] = clusters
```

2s completed at 12:12 AM

In below screen displaying TFIDF vector for few rows from dataset



The screenshot shows a Jupyter Notebook interface displaying a dataset table. The code cell shows:

```
dataset['cluster'] = clusters
dataset
```

The output is a table with the following columns: photo_id, title, description, tags, faves, and cluster.

	photo_id	title	description	tags	faves	cluster
0	47922891637	British Museum	A marble statue of Emperor Septimius Severus, ...	britishmuseum, london, sculpture, roman	4.0	7
1	47922890172	British Museum	Busts of Roman worthies.	britishmuseum, london, sculpture, roman	5.0	7
2	47922879348	British Museum	A bronze bust of the emperor Lucius Venus, who...	britishmuseum, london, sculpture, roman, bronz...	7.0	7
3	46995613225	Ludgate House 'Demolished	Heres A Special Request For Contact 'RTM Boy ...	london, se1, southwark, blackfriars, local, hi...	4.0	9
6	40943646353	Sip Savor Enjoy	Nothing like a soothing cup of tea to enjoy in...	stillife, outdoors, park, trees, grass, drink...	3.0	1
...
9263	34652005736	20140913 200526	1S United Kingdom England London	1s, 2046x2046x24b, 8c, ...	0.0	0

2s completed at 12:12 AM

The screenshot displays a Google Colab notebook interface. The top bar shows the file explorer with folders like 'drive', 'model', and 'sample_data'. The main area contains a scatter plot titled 'Similar Interest Tourist Places Graph'. The y-axis is labeled 'Number of Visits' and ranges from 0 to 100. The x-axis lists 25 tourist locations. The plot shows a strong positive correlation between visits to 'The House Of Toby' and 'A Long Time Ago', with other locations showing lower visit counts.

Tourist Place	Number of Visits (approx.)
The House Of Toby	5
A Long Time Ago	5
Number 19	2
the London Marathon	1
Street art	1
Marion Court	10
Widstria and Pergula	2
Elisa	2
Palm Tree and the Vines	2
Piccadilly Circus	12
on Kings Cross 11.3.19	1
on Kings Cross 11.3.19	1
on Kings Cross 11.3.19	1
Last Man Off	100
St Durstan in the East	1
Docklands Golden Hour	15
pon House Demolished	15
erwer Bridge Glasoden	1
North West London, UK	1
king williams temple	8
Hols & Seck	8
silver daisy bush	15
The White Tower	35
Claudia Rose	1
the middle of no where	1
b x Candid Portraits Ltd	1
British Museum	5
Victoria & Prince Albert	28
Hats & Coats	25
Charterhouse Chapel	1
ower bridge Reflection	1
ne Dome and the River	1
City Sunset	1
the Man And The ocean	40

The screenshot shows a Google Colab notebook titled "TouristIpybn". The interface includes a top toolbar with icons for file operations, a left sidebar with a file explorer showing a directory structure with "drive", "model", and "sample_data", and a main code editor area. The code in the editor is as follows:

```

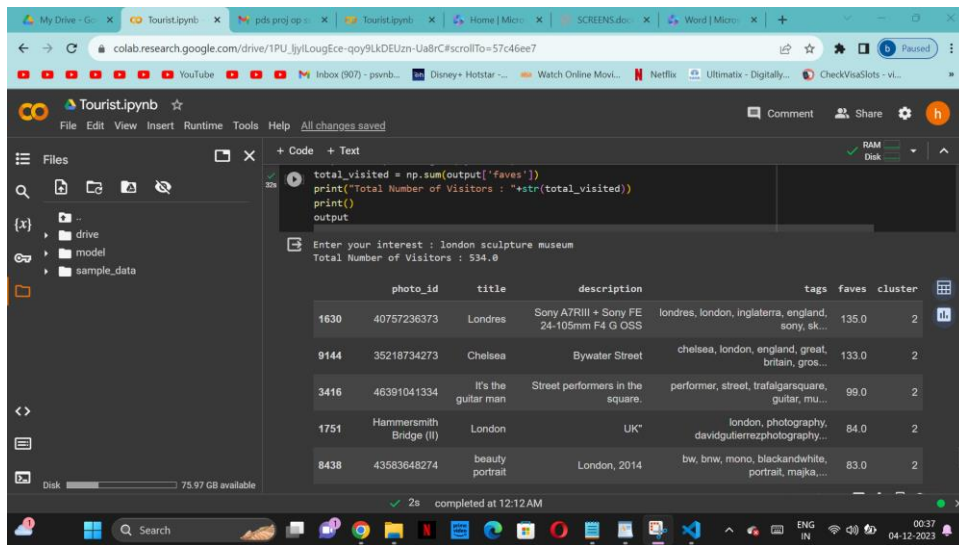
#possible tourist places to search from geo tagged
...
britishmuseum london sculpture roman buckinghampalace themall hydepark parkline unitedkingdom towerhill united
...

query = input("Enter your interest : ")#take user interest query as input
data = query.lower()
data = cleanText(data)#clean the query
temp = []
temp.append(data)
temp = tfidf.transform(temp).toarray()#convert query to vector
XX = []
for i in range(0,tfidf.X.shape[0]):
    XX.append(temp[0])
temp = np.asarray(XX)

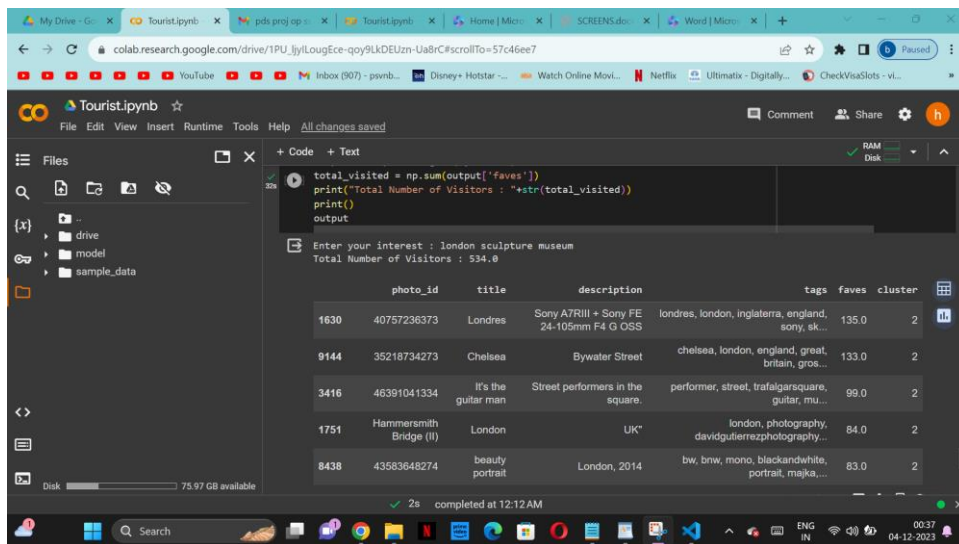
#get user interest places from kmeans cluster
predict = kmeans.predict(temp)
predict = predict[0]#display predicted user interest places from clusters
output = dataset[clusters == predict]
output = output.nlargest(5,"faves")
total_visited = np.sum(output['faves'])
print("Total Number of Visitors : "+str(total_visited))
print()
output
  
```

The code is executed, and the status bar at the bottom indicates "2s completed at 12:12 AM". The right sidebar shows a "RAM Disk" indicator.

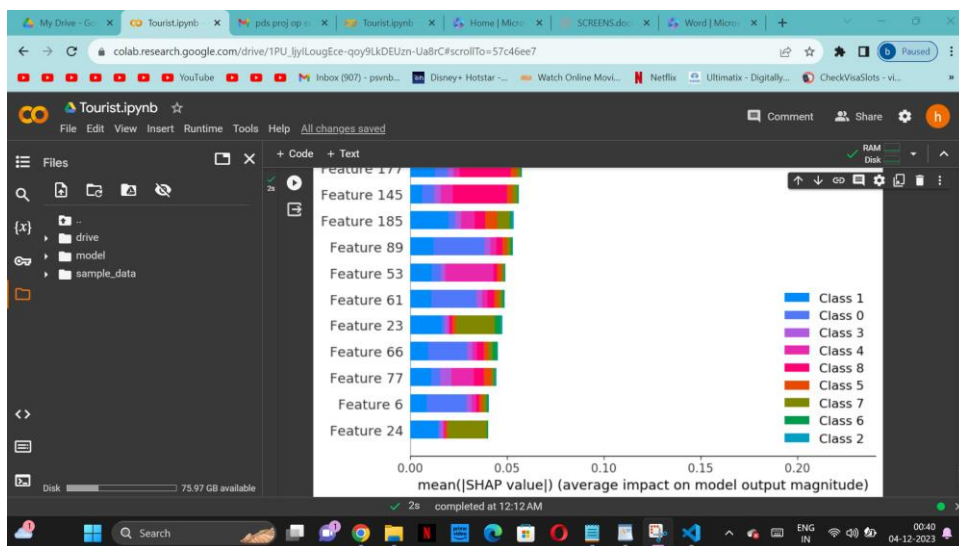
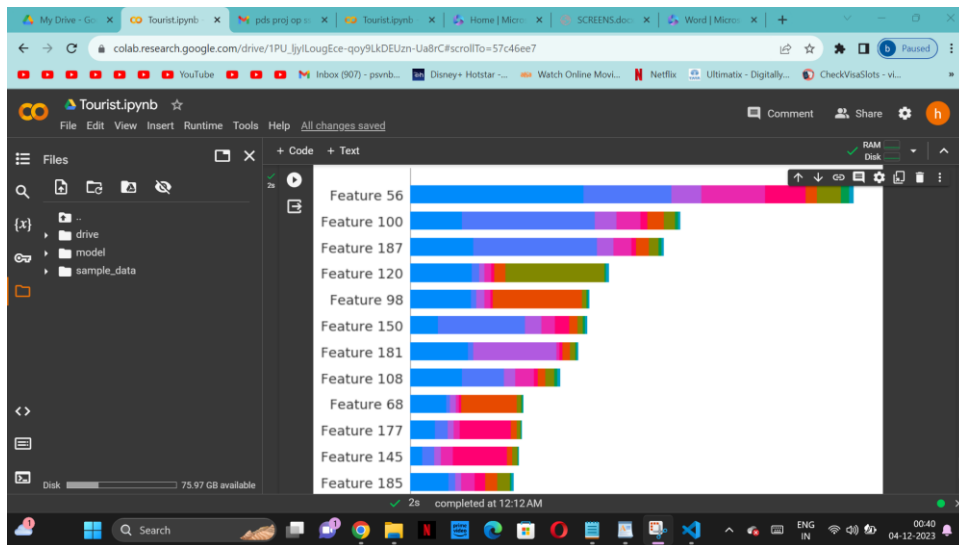
In above screen we define function which will read user query and then recommend similar places from cluster based on user interest query



In above screen in Text Box we gave query as ‘london sculpture museum’ and press enter key to get top 5 recommended places like below screen



In above table we got top 5 recommended places from cluster 2 and in blue colour text we can see number of users visited those places. Similarly you can enter query and get popular tourist places from cluster. Above output can be consider as future tourism places which will be in demand



In above screen we have added SHAP modelling tool to explain about the model who is using which features most for prediction.