Data Collection

The idea here is to gather my own data for classification. I am targeting data of videos available on Youtube. The data is collected for **6 categories**:

- Travel Blogs
- Science and Technology
- Food
- Manufacturing
- History
- Art and Music

To perform the required data collection, I used the **Youtube API v3**. I decided to use the Youtube API since I needed to collected >1700 samples, since it has an option to get data from subsequent pages of the search results.

In [6]:
```
'''
from apiclient.discovery import build
import pandas as pd

# Data to be stored
category = []
no_of_samples = 1700

# Gathering Data using the Youtube API
api_key = "AIzaSyAS9eTgOEnOJ2GlJbbqm_0bR1onuRQjTHE"
youtube_api = build('youtube','v3', developerKey = api_key)

# Travel Data
tvl_titles = []
tvl_descriptions = []
tvl_ids = []
```

```
req = youtube_api.search().list(q='travel vlogs', part='snippet', type
='video', maxResults = 50)
res = req.execute()
while(len(tvl_titles)<no_of_samples):
    for i in range(len(res['items'])):
        tvl_titles.append(res['items'][i]['snippet']['title'])
        tvl_descriptions.append(res['items'][i]['snippet']['descriptio
n'])
        tvl_ids.append(res['items'][i]['id']['videoId'])
        category.append('travel')

    if('nextPageToken' in res):
        next_page_token = res['nextPageToken']
        req = youtube_api.search().list(q='travelling', part='snippet',
 type='video', maxResults = 50, pageToken=next_page_token)
        res = req.execute()
    else:
        break


# Science Data
science_titles = []
science_descriptions = []
science_ids = []

next_page_token = None
req = youtube_api.search().list(q='robotics', part='snippet', type='vid
eo', maxResults = 50)
res = req.execute()
while(len(science_titles)<no_of_samples):
    if(next_page_token is not None):
        req = youtube_api.search().list(q='robotics', part='snippet', t
ype='video', maxResults = 50, pageToken=next_page_token)
        res = req.execute()
    for i in range(len(res['items'])):
        science_titles.append(res['items'][i]['snippet']['title'])
        science_descriptions.append(res['items'][i]['snippet']['descrip
tion'])
```

```
        science_ids.append(res['items'][i]['id']['videoId'])
        category.append('science and technology')

    if('nextPageToken' in res):
        next_page_token = res['nextPageToken']
    else:
        break

# Food Data
food_titles = []
food_descriptions = []
food_ids = []

next_page_token = None
req = youtube_api.search().list(q='delicious food', part='snippet', typ
e='video', maxResults = 50)
res = req.execute()
while(len(food_titles)<no_of_samples):
    if(next_page_token is not None):
        req = youtube_api.search().list(q='delicious food', part='snipp
et', type='video', maxResults = 50, pageToken=next_page_token)
        res = req.execute()
    for i in range(len(res['items'])):
        food_titles.append(res['items'][i]['snippet']['title'])
        food_descriptions.append(res['items'][i]['snippet']['descriptio
n'])
        food_ids.append(res['items'][i]['id']['videoId'])
        category.append('food')

    if('nextPageToken' in res):
        next_page_token = res['nextPageToken']
    else:
        break

# Food Data
manufacturing_titles = []
manufacturing_descriptions = []
manufacturing_ids = []
```

```
next_page_token = None
req = youtube_api.search().list(q='3d printing', part='snippet', type
='video', maxResults = 50)
res = req.execute()
while(len(manufacturing_titles)<no_of_samples):
    if(next_page_token is not None):
        req = youtube_api.search().list(q='3d printing', part='snippe
t', type='video', maxResults = 50, pageToken=next_page_token)
        res = req.execute()
    for i in range(len(res['items'])):
        manufacturing_titles.append(res['items'][i]['snippet']['titl
e'])
        manufacturing_descriptions.append(res['items'][i]['snippet']['d
escription'])
        manufacturing_ids.append(res['items'][i]['id']['videoId'])
        category.append('manufacturing')

    if('nextPageToken' in res):
        next_page_token = res['nextPageToken']
    else:
        break

# History Data
history_titles = []
history_descriptions = []
history_ids = []

next_page_token = None
req = youtube_api.search().list(q='archaeology', part='snippet', type
='video', maxResults = 50)
res = req.execute()
while(len(history_titles)<no_of_samples):
    if(next_page_token is not None):
        req = youtube_api.search().list(q='archaeology', part='snippe
t', type='video', maxResults = 50, pageToken=next_page_token)
        res = req.execute()
    for i in range(len(res['items'])):
        history_titles.append(res['items'][i]['snippet']['title'])
        history_descriptions.append(res['items'][i]['snippet']['descrip
```

```
tion'])
        history_ids.append(res['items'][i]['id']['videoId'])
        category.append('history')

    if('nextPageToken' in res):
        next_page_token = res['nextPageToken']
    else:
        break

# Art and Music Data
art_titles = []
art_descriptions = []
art_ids = []

next_page_token = None
req = youtube_api.search().list(q='painting', part='snippet', type='vid
eo', maxResults = 50)
res = req.execute()
while(len(art_titles)<no_of_samples):
    if(next_page_token is not None):
        req = youtube_api.search().list(q='painting', part='snippet', t
ype='video', maxResults = 50, pageToken=next_page_token)
        res = req.execute()
    for i in range(len(res['items'])):
        art_titles.append(res['items'][i]['snippet']['title'])
        art_descriptions.append(res['items'][i]['snippet']['descriptio
n'])
        art_ids.append(res['items'][i]['id']['videoId'])
        category.append('art and music')

    if('nextPageToken' in res):
        next_page_token = res['nextPageToken']
    else:
        break


# Construct Dataset
final_titles = tvl_titles + science_titles + food_titles + manufacturin
g_titles + history_titles + art_titles
```

```
final_descriptions = tvl_descriptions + science_descriptions + food_des
criptions + manufacturing_descriptions + history_descriptions + art_des
criptions
final_ids = tvl_ids + science_ids + food_ids + manufacturing_ids + hist
ory_ids + art_ids
data = pd.DataFrame({'Video Id': final_ids, 'Title': final_titles, 'Des
cription': final_descriptions, 'Category': category})
data.to_csv('Videos_data.csv')
'''
```

Out[6]:

```
'\nfrom apiclient.discovery import build\nimport pandas as pd\n\n# Data
to be stored\ncategory = []\nno_of_samples = 1700\n\n# Gathering Data u
sing the Youtube API\napi_key = "AIzaSyAS9eTgOEnOJ2GlJbbqm_0bR1onuRQjTH
E"\nyoutube_api = build(\'youtube\',\'v3\', developerKey = api_key)\n\n
# Travel Data\ntvl_titles = []\ntvl_descriptions = []\ntvl_ids = []\n\n
req = youtube_api.search().list(q=\'travel vlogs\', part=\'snippet\', t
ype=\'video\', maxResults = 50)\nres = req.execute()\nwhile(len(tvl_tit
les)<no_of_samples):\n    for i in range(len(res[\'items\'])):\n
tvl_titles.append(res[\'items\'][i][\'snippet\'][\'title\'])\n        t
vl_descriptions.append(res[\'items\'][i][\'snippet\'][\'description\'])
\n        tvl_ids.append(res[\'items\'][i][\'id\'][\'videoId\'])\n
   category.append(\'travel\')\n            \n    if(\'nextPageToken\'
in res):\n        next_page_token = res[\'nextPageToken\']\n        req
= youtube_api.search().list(q=\'travelling\', part=\'snippet\', type=
\'video\', maxResults = 50, pageToken=next_page_token)\n        res = r
eq.execute()\n    else:\n        break\n\n\n# Science Data\nscience_tit
les = []\nscience_descriptions = []\nscience_ids = []\n\nnext_page_toke
n = None\nreq = youtube_api.search().list(q=\'robotics\', part=\'snippe
t\', type=\'video\', maxResults = 50)\nres = req.execute()\nwhile(len(s
cience_titles)<no_of_samples):\n    if(next_page_token is not None):\n
       req = youtube_api.search().list(q=\'robotics\', part=\'snippet
\', type=\'video\', maxResults = 50, pageToken=next_page_token)\n
  res = req.execute()\n    for i in range(len(res[\'items\'])):\n
  science_titles.append(res[\'items\'][i][\'snippet\'][\'title\'])\n
     science_descriptions.append(res[\'items\'][i][\'snippet\'][\'descr
iption\'])\n        science_ids.append(res[\'items\'][i][\'id\'][\'vide
oId\'])\n        category.append(\'science and technology\')\n
   \n    if(\'nextPageToken\' in res):\n        next_page_token = res
[\'nextPageToken\']\n    else:\n        break\n    \n# Food Data\nfood_
titles = []\nfood_descriptions = []\nfood_ids = []\n\nnext_page_token =
```

```
None\nreq = youtube_api.search().list(q=\'delicious food\', part=\'snip
pet\', type=\'video\', maxResults = 50)\nres = req.execute()\nwhile(len
(food_titles)<no_of_samples):\n    if(next_page_token is not None):\n
      req = youtube_api.search().list(q=\'delicious food\', part=\'snip
pet\', type=\'video\', maxResults = 50, pageToken=next_page_token)\n
     res = req.execute()\n    for i in range(len(res[\'items\'])):\n
     food_titles.append(res[\'items\'][i][\'snippet\'][\'title\'])\n
     food_descriptions.append(res[\'items\'][i][\'snippet\'][\'descript
ion\'])\n        food_ids.append(res[\'items\'][i][\'id\'][\'videoId

\'])\n        category.append(\'food\')\n            \n    if(\'nextPag
eToken\' in res):\n        next_page_token = res[\'nextPageToken\']\n
  else:\n        break\n\n# Food Data\nmanufacturing_titles = []\nmanuf
acturing_descriptions = []\nmanufacturing_ids = []\n\nnext_page_token =
None\nreq = youtube_api.search().list(q=\'3d printing\', part=\'snippet
\', type=\'video\', maxResults = 50)\nres = req.execute()\nwhile(len(ma
nufacturing_titles)<no_of_samples):\n    if(next_page_token is not Non
e):\n        req = youtube_api.search().list(q=\'3d printing\', part=
\'snippet\', type=\'video\', maxResults = 50, pageToken=next_page_toke
n)\n        res = req.execute()\n    for i in range(len(res[\'items
\'])):\n        manufacturing_titles.append(res[\'items\'][i][\'snippet
\'][\'title\'])\n        manufacturing_descriptions.append(res[\'items
\'][i][\'snippet\'][\'description\'])\n        manufacturing_ids.append
(res[\'items\'][i][\'id\'][\'videoId\'])\n        category.append(\'man
ufacturing\')\n            \n    if(\'nextPageToken\' in res):\n
next_page_token = res[\'nextPageToken\']\n    else:\n        break\n
\n# History Data\nhistory_titles = []\nhistory_descriptions = []\nhisto
ry_ids = []\n\nnext_page_token = None\nreq = youtube_api.search().list
(q=\'archaeology\', part=\'snippet\', type=\'video\', maxResults = 50)
\nres = req.execute()\nwhile(len(history_titles)<no_of_samples):\n    i
f(next_page_token is not None):\n        req = youtube_api.search().lis
t(q=\'archaeology\', part=\'snippet\', type=\'video\', maxResults = 50,
pageToken=next_page_token)\n        res = req.execute()\n    for i in r
ange(len(res[\'items\'])):\n        history_titles.append(res[\'items
\'][i][\'snippet\'][\'title\'])\n        history_descriptions.append(re
s[\'items\'][i][\'snippet\'][\'description\'])\n        history_ids.app
end(res[\'items\'][i][\'id\'][\'videoId\'])\n        category.append
(\'history\')\n            \n    if(\'nextPageToken\' in res):\n
next_page_token = res[\'nextPageToken\']\n    else:\n        break\n
\n# Art and Music Data\nart_titles = []\nart_descriptions = []\nart_ids
```

```
= []\n\nnext_page_token = None\nreq = youtube_api.search().list(q=\'pai
nting\', part=\'snippet\', type=\'video\', maxResults = 50)\nres = req.
execute()\nwhile(len(art_titles)<no_of_samples):\n    if(next_page_toke
n is not None):\n        req = youtube_api.search().list(q=\'painting
\', part=\'snippet\', type=\'video\', maxResults = 50, pageToken=next_p
age_token)\n        res = req.execute()\n    for i in range(len(res[\'i
tems\'])):\n        art_titles.append(res[\'items\'][i][\'snippet\']
[\'title\'])\n        art_descriptions.append(res[\'items\'][i][\'snipp
et\'][\'description\'])\n        art_ids.append(res[\'items\'][i][\'id

\'][\'videoId\'])\n        category.append(\'art and music\')\n
    \n    if(\'nextPageToken\' in res):\n        next_page_token = res
[\'nextPageToken\']\n    else:\n        break\n \n    \n# Construct Dat
aset\nfinal_titles = tvl_titles + science_titles + food_titles + manufa
cturing_titles + history_titles + art_titles\nfinal_descriptions = tvl_
descriptions + science_descriptions + food_descriptions + manufacturing
_descriptions + history_descriptions + art_descriptions\nfinal_ids = tv
l_ids + science_ids + food_ids + manufacturing_ids + history_ids + art_
ids\ndata = pd.DataFrame({\'Video Id\': final_ids, \'Title\': final_tit
les, \'Description\': final_descriptions, \'Category\': category}) \nda
ta.to_csv(\'Videos_data.csv\')\n'
```

Text Classification

Importing Libraries

In [3]:
```
import pandas as pd
import nltk
#nltk.download()
from nltk.corpus import stopwords
import re
import string
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem.wordnet import WordNetLemmatizer
```

In [8]:
```
# Import Data
vdata = pd.read_csv('Videos_data.csv')
vdata = data.iloc[:, 1:]     # Remove extra un-named column
vdata.head(10)
```

Out[8]:

	Title	Description	Category
0	Ep 1| Travelling through North East India | Of...	The journey to Arunachal, North East India beg...	travel
1	How do I travel so much ! How do I earn money!!	SUBSCRIBE - https://goo.gl/dEtSMJ ('MountainTr...	travel
2	TRAVEL VLOG · Welcome to Bali | PRISCILLA LEE	I had the chance to fly out to Bali with my wh...	travel
3	GOA TRAVEL DIARY | FOUR DAYS IN GOA | TRAVEL O...	Hope you enjoy MY GOA TRAVEL DIARY this video!...	travel
4	5 Steps to Becoming a Travel Blogger	Travel blogger, Nikki Vargas, of The Pin the M...	travel
5	Backpacking In Meghalaya | NorthEast India Tri...	In this video I explored North East India, sta...	travel
6	Welcome to Peru! | Best Essential Tips & T...	Welcome to Peru! This essential travel guide w...	travel
7	How to Start a Travel Blog [2019] Travel Blogg...	Create a Travel Blog Website for Just $3.95 + ...	travel
8	A Day with KSRTC Bus Fans - Aanavandi Travel B...	ആനവണ്ടി ഭ്രാന്തൻമാരോടൊപ്പം കുമളിയിൽ ഒരു ദിവസം ...	travel
9	What is it like to travel in PAKISTAN?	Subscribe now: https://goo.gl/6zXZGK Watch the...	travel

Data Preprocessing and Cleaning

Missing Values

In [10]:
```
# Missing Values
num_missing_desc = data.isnull().sum()[2]    # No. of values with msisi
ng descriptions
print('Number of missing values: ' + str(num_missing_desc))
vdata = data.dropna()
```

```
Number of missing values: 0
```

Text Cleaning

The cleaning of the text is performed in the following manner:

- Converting to Lowercase
- Removing numerical values, because they do not contribute towards predicting the category
- Removing Punctuation because special characters like $, !, etc. do not hold any useful information
- Removing extra white spaces
- Tokenizing into words - This means to convert a text string into a list of 'tokens', where each token is a word. Eg. The sentence 'My Name is Rishi' becomes ['My', 'Name', 'is', 'Rishi']
- Removing all non-alphabetic words
- Filtering out stop words such as and, the, is, etc. because they do not contain useful information for text classification
- Lemmatizing words - Lemmatizing reduces words to their base meaning, such as words 'fly' and 'flying' are both convert to just 'fly'

In [ ]:
```
```

In [12]:
```
# Change to lowercase
vdata['Title'] = vdata['Title'].map(lambda x: x.lower())
vdata['Description'] = vdata['Description'].map(lambda x: x.lower())

# Remove numbers
vdata['Title'] = vdata['Title'].map(lambda x: re.sub(r'\d+', '', x))
vdata['Description'] = vdata['Description'].map(lambda x: re.sub(r'\d+'
```

```
, '', x))

# Remove Punctuation
vdata['Title']  = vdata['Title'].map(lambda x: x.translate(x.maketrans(
'', '', string.punctuation)))
vdata['Description']  = vdata['Description'].map(lambda x: x.translate(
x.maketrans('', '', string.punctuation)))

# Remove white spaces
vdata['Title'] = vdata['Title'].map(lambda x: x.strip())
vdata['Description'] = vdata['Description'].map(lambda x: x.strip())

# Tokenize into words
vdata['Title'] = vdata['Title'].map(lambda x: word_tokenize(x))
vdata['Description'] = vdata['Description'].map(lambda x: word_tokenize
(x))

# Remove non alphabetic tokens
vdata['Title'] = vdata['Title'].map(lambda x: [word for word in x if wo
rd.isalpha()])
vdata['Description'] = vdata['Description'].map(lambda x: [word for wor
d in x if word.isalpha()])

# filter out stop words
stop_words = set(stopwords.words('english'))
vdata['Title'] = vdata['Title'].map(lambda x: [w for w in x if not w in
 stop_words])
vdata['Description'] = vdata['Description'].map(lambda x: [w for w in x
 if not w in stop_words])

# Word Lemmatization
lem = WordNetLemmatizer()
vdata['Title'] = vdata['Title'].map(lambda x: [lem.lemmatize(word,"v")
for word in x])
vdata['Description'] = vdata['Description'].map(lambda x: [lem.lemmatiz
e(word,"v") for word in x])

# Turn lists back to string
```

```
vdata['Title'] = vdata['Title'].map(lambda x: ' '.join(x))
vdata['Description'] = vdata['Description'].map(lambda x: ' '.join(x))
```

In [14]:
```
vdata.head(10)
```

Out[14]:

	Video Id	Title	Description	Category
0	ehmsJLZlCZ0	ep travel north east india arunachal journey b...	journey arunachal north east india begin train...	travel
1	e2NQE41J5eM	travel much earn money	subscribe httpsgoogldetsmj mountaintrekker gim...	travel
2	i9E_Blai8vk	travel vlog welcome bali priscilla lee	chance fly bali whole family thanksgiving firs...	travel
3	#NAME?	goa travel diary four days goa travel outfit i...	hope enjoy goa travel diary video dont forget ...	travel
4	7ByoBJYXU0k	step become travel blogger	travel blogger nikki vargas pin map project vo...	travel
5	yvn79Rv0F48	backpack meghalaya northeast india trip sohra ...	video explore north east india start guwahati ...	travel
6	SL_YBLWdZb8	welcome peru best essential tip amp travel guide	welcome peru essential travel guide show best ...	travel
7	kiNyRY5s7n8	start travel blog travel blogging fulltime	create travel blog website httpbitlyltstarttra...	travel
8	kY41XgTEEQU	day ksrtc bus fan aanavandi travel blog meet k...	ksrtc	travel
9	7mIzRYh8jGA	like travel pakistan	subscribe httpsgooglzxzgk watch full series ht...	travel

Data Preprocessing

Label Encoding classes

In [19]:
```
# Encode classes
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
le.fit(vdata.Category)
vdata.Category = le.transform(vdata.Category)
vdata.head(10)
```

Out[19]:

	Video Id	Title	Description	Category
0	ehmsJLZlCZ0	ep travel north east india arunachal journey b...	journey arunachal north east india begin train...	5
1	e2NQE41J5eM	travel much earn money	subscribe httpsgoogldetsmj mountaintrekker gim...	5
2	i9E_Blai8vk	travel vlog welcome bali priscilla lee	chance fly bali whole family thanksgiving firs...	5
3	#NAME?	goa travel diary four days goa travel outfit i...	hope enjoy goa travel diary video dont forget ...	5
4	7ByoBJYXU0k	step become travel blogger	travel blogger nikki vargas pin map project vo...	5
5	yvn79Rv0F48	backpack meghalaya northeast india trip sohra ...	video explore north east india start guwahati ...	5
6	SL_YBLWdZb8	welcome peru best essential tip amp travel guide	welcome peru essential travel guide show best ...	5
7	kiNyRY5s7n8	start travel blog travel blogging fulltime	create travel blog website httpbitlyltstarttra...	5
8	kY41XgTEEQU	day ksrtc bus fan aanavandi travel blog meet k...	ksrtc	5
9	7mIzRYh8jGA	like travel pakistan	subscribe httpsgooglzxzgk watch full series ht...	5

Vectorizing text features using TF-IDF

In [20]:
```
# TF-IDF
from sklearn.feature_extraction.text import TfidfVectorizer
```

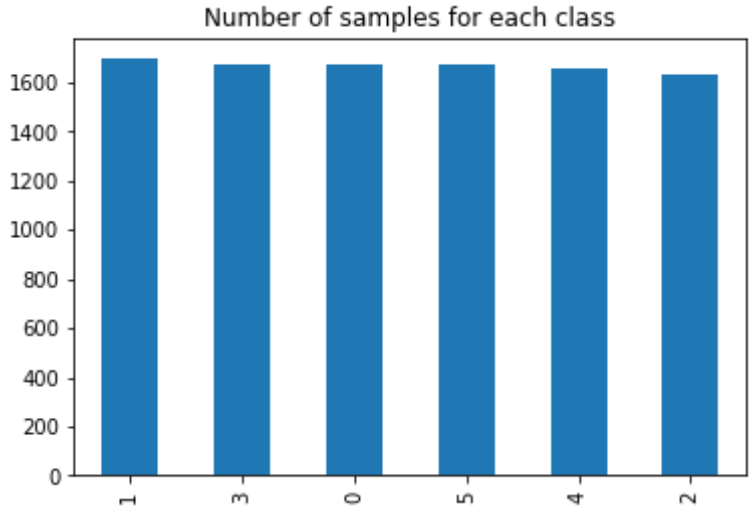
```
tfidf_title = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', e
ncoding='latin-1', ngram_range=(1, 2), stop_words='english')
tfidf_desc = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', en
coding='latin-1', ngram_range=(1, 2), stop_words='english')
labels = vdata.Category
features_title = tfidf_title.fit_transform(vdata.Title).toarray()
features_description = tfidf_desc.fit_transform(vdata.Description).toar
ray()
print('Title Features Shape: ' + str(features_title.shape))
print('Description Features Shape: ' + str(features_description.shape))
```

```
Title Features Shape: (9999, 2637)
Description Features Shape: (9999, 4858)
```

Data Analysis and Feature Exploration

In [22]:
```
# Plotting class distribution
vdata['Category'].value_counts().sort_values(ascending=False).plot(kind
='bar', y='Number of Samples',
                                                                    title=
'Number of samples for each class')
```

Out[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21fb733c108>
```



Now let us see if the features are correctly extracted from the text data by checking the most important features for each class

In [24]:
```
# Best 5 keywords for each class using Title Feaures
from sklearn.feature_selection import chi2
import numpy as np
N = 10
for current_class in list(le.classes_):
    current_class_id = le.transform([current_class])[0]
    features_chi2 = chi2(features_title, labels == current_class_id)
    indices = np.argsort(features_chi2[0])
    feature_names = np.array(tfidf_title.get_feature_names())[indices]
    unigrams = [v for v in feature_names if len(v.split(' ')) == 1]
    bigrams = [v for v in feature_names if len(v.split(' ')) == 2]
    print("# '{}':".format(current_class))
    print("Most correlated unigrams:")
    print('-' *30)
    print('. {}'.format('\n. '.join(unigrams[-N:])))
    print("Most correlated bigrams:")
    print('-' *30)
    print('. {}'.format('\n. '.join(bigrams[-N:])))
    print("\n")
```

```
# 'art and music':
Most correlated unigrams:
------------------------------
. musical
. live
. travel
. arts
. video
. paint
. official
. music
. art
. theatre
Most correlated bigrams:
------------------------------
. art challenge
. avengers endgame
. theatre company
. theatre official
. theatre congolais
. capitol theatre
. musical theatre
. work theatre
. official music
. music video


# 'food':
Most correlated unigrams:
------------------------------
. street
. recipe
. taste
. healthy
. try
. foods
. eat
. snack

  cook
```

```
. cook
. food
Most correlated bigrams:
------------------------------
. cook guy
. sam cook
. try hiho
. eat snack
. emmy eat
. healthy snack
. snack amp
. taste test
. kid try
. street food


# 'history':
Most correlated unigrams:
------------------------------
. archaeologist
. rap
. anthropologist
. anthropological
. archaeologists
. discoveries
. archaeological
. archaeology
. history
. anthropology
Most correlated bigrams:
------------------------------
. cultural anthropology
. history documentary
. concepts anthropology
. world history
. forensic anthropology
. history channel
. rap battle
. epic rap

  battle history
```

```
. battle history
. archaeological discoveries


# 'manufacturing':
Most correlated unigrams:
------------------------------
. additive
. lean
. production
. factory
. manufacturer
. business
. printer
. process
. print
. manufacture
Most correlated bigrams:
------------------------------
. manufacture industry
. manufacture tour
. manufacture engineer
. future manufacture
. advance manufacture
. manufacture plant
. lean manufacture
. additive manufacture
. manufacture business
. manufacture process


# 'science and technology':
Most correlated unigrams:
------------------------------
. robots
. primitive
. technologies
. quantum
. robotics

  compute
```

```
. compute
. computers
. science
. computer
. technology
Most correlated bigrams:
------------------------------
. course computer
. quantum computers
. future technology
. university science
. quantum compute
. science amp
. amp technology
. primitive technology
. computer science
. science technology


# 'travel':
Most correlated unigrams:
------------------------------
. viewfinder
. manufacture
. tip
. expedia
. trip
. blogger
. vlog
. travellers
. blog
. travel
Most correlated bigrams:
------------------------------
. start travel
. travel light
. travel salesman
. travel guide
. expedia viewfinder

  viewfinder travel
```

```
. viewfinder travel
. travel blogger
. tip travel
. travel vlog
. travel blog
```

In [25]:
```
# Best 5 keywords for each class using Description Features
from sklearn.feature_selection import chi2
import numpy as np
N = 10
for current_class in list(le.classes_):
    current_class_id = le.transform([current_class])[0]
    features_chi2 = chi2(features_description, labels == current_class_
id)
    indices = np.argsort(features_chi2[0])
    feature_names = np.array(tfidf_desc.get_feature_names())[indices]
    unigrams = [v for v in feature_names if len(v.split(' ')) == 1]
    bigrams = [v for v in feature_names if len(v.split(' ')) == 2]
    print("# '{}':".format(current_class))
    print("Most correlated unigrams:")
    print('-' *30)
    print('. {}'.format('\n. '.join(unigrams[-N:])))
    print("Most correlated bigrams:")
    print('-' *30)
    print('. {}'.format('\n. '.join(bigrams[-N:])))
    print("\n")
```

```
# 'art and music':
Most correlated unigrams:
------------------------------
. spotify
. album
. draw
. listen
. arts
. official
. paint
. music
```

```
. art
. theatre
Most correlated bigrams:
------------------------------
. work theatre
. official video
. live capitol
. theatre passaic
. passaic nj
. capitol theatre
. click listen
. production connexion
. official music
. music video


# 'food':
Most correlated unigrams:
------------------------------
. delicious
. recipes
. taste
. healthy
. recipe
. foods
. eat
. snack
. cook
. food
Most correlated bigrams:
------------------------------
. httpbitlyznbqjw come
. httpbitlycomhihofans update
. sign httpbitlycomhihofans
. series httpbitlyznbqjw
. update hiho
. special offer
. hiho special
. come play
. sponsor series
```

```
. street food


# 'history':
Most correlated unigrams:
------------------------------
. rap
. anthropologist
. ancient
. archaeologist
. archaeologists
. discoveries
. archaeological
. history
. archaeology
. anthropology
Most correlated bigrams:
------------------------------
. begin april
. season begin
. decide erb
. erb season
. history decide
. episode epic
. epic rap
. battle history
. rap battle
. archaeological discoveries


# 'manufacturing':
Most correlated unigrams:
------------------------------
. machine
. plant
. manufacturers
. manufacturer
. printers
. factory
```

```
. printer
. process
. print
. manufacture
Most correlated bigrams:
------------------------------
. manufacture facility
. manufacture industry
. manufacture company
. manufacture unit
. advance manufacture
. process make
. lean manufacture
. additive manufacture
. manufacture business
. manufacture process


# 'science and technology':
Most correlated unigrams:
------------------------------
. technologies
. future
. robots
. robotics
. compute
. quantum
. computers
. science
. computer
. technology
Most correlated bigrams:
------------------------------
. new technology
. artificial intelligence
. cloud compute
. future technology
. university science
. quantum computers
```

```
. primitive technology
. quantum compute
. computer science
. science technology


# 'travel':
Most correlated unigrams:
------------------------------
. stay
. expedia
. tip
. adventure
. blogger
. vlog
. travellers
. trip
. blog
. travel
Most correlated bigrams:
------------------------------
. becomingfilipino travel
. instagram taesungsayshi
. travel video
. travel world
. travel vlog
. tip travel
. start travel
. expedia viewfinder
. travel blogger
. travel blog
```

Modeling and Training

Features for both **Title** and **Description** are extracted and then concantenated in order to construct a final feature matrix

In [26]:
```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn import linear_model
from sklearn.ensemble import AdaBoostClassifier

X_train, X_test, y_train, y_test = train_test_split(vdata.iloc[:, 1:3],
 vdata['Category'], random_state = 0)
X_train_title_features = tfidf_title.transform(X_train['Title']).toarra
y()
X_train_desc_features = tfidf_desc.transform(X_train['Description']).to
array()
features = np.concatenate([X_train_title_features, X_train_desc_feature
s], axis=1)
```

In [32]:
```
X_train.head()
```

Out[32]:

	Title	Description
3072	lec mit introduction computer science program ...	lecture goals course computation introduction ...
713	travel cargo ship philippines	travel cargo ship yes something many people th...
3598	store cat food vs homemade	patreon httpswwwpatreoncomjunskitchen thank wa...
1680	indian tourism travel goa hd	create video youtube video editor httpwwwyoutu...
819	travel hong kong	heres hong kong vlog stay overnight airport vi...

In [31]:
```
y_train.head()
```

Out[31]:
```
3072    4
713     5
3598    1
1680    5
```

```
819     5
Name: Category, dtype: int32
```

In [33]:
```
# Naive Bayes
nb = MultinomialNB().fit(features, y_train)
# SVM
svm = linear_model.SGDClassifier(loss='modified_huber',max_iter=1000, t
ol=1e-3).fit(features,y_train)
# AdaBoost
adaboost = AdaBoostClassifier(n_estimators=40,algorithm="SAMME").fit(fe
atures,y_train)
```

In [34]:
```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from keras.utils.np_utils import to_categorical

# The maximum number of words to be used. (most frequent)
MAX_NB_WORDS = 20000
# Max number of words in each complaint.
MAX_SEQUENCE_LENGTH = 50
# This is fixed.
EMBEDDING_DIM = 100

# Combining titles and descriptions into a single sentence
titles = vdata['Title'].values
descriptions = vdata['Description'].values
data_for_lstms = []
for i in range(len(titles)):
    temp_list = [titles[i], descriptions[i]]
    data_for_lstms.append(' '.join(temp_list))

tokenizer = Tokenizer(num_words=MAX_NB_WORDS, filters='!"#$%&()*+,-./:;
<=>?@[\]^_`{|}~', lower=True)
tokenizer.fit_on_texts(data_for_lstms)
word_index = tokenizer.word_index
print('Found %s unique tokens.' % len(word_index))
```

```
# Convert the data to padded sequences
X = tokenizer.texts_to_sequences(data_for_lstms)
X = pad_sequences(X, maxlen=MAX_SEQUENCE_LENGTH)
print('Shape of data tensor:', X.shape)

# One-hot Encode labels
Y = pd.get_dummies(vdata['Category']).values
print('Shape of label tensor:', Y.shape)

# Splitting into training and test set
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, random_state =
 42)
```

```
Using TensorFlow backend.
```

```
Found 26134 unique tokens.
Shape of data tensor: (9999, 50)
Shape of label tensor: (9999, 6)
```

In [35]:

```
# Define LSTM Model
model = Sequential()
model.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM, input_length=X.shape[1
]))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(6, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metric
s=['accuracy'])
print(model.summary())
```

```
WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:74: The name tf.get_defaul
t_graph is deprecated. Please use tf.compat.v1.get_default_graph instea
d.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:517: The name tf.placehold
er is deprecated. Please use tf.compat.v1.placeholder instead.
```

```
WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:4138: The name tf.random_u
niform is deprecated. Please use tf.random.uniform instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:133: The name tf.placehold
er_with_default is deprecated. Please use tf.compat.v1.placeholder_with
_default instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:3445: calling dropout (fro
m tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will b
e removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate =
1 - keep_prob`.
WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\optimizers.py:790: The name tf.train.Optimizer is deprec
ated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:3295: The name tf.log is d
eprecated. Please use tf.math.log instead.


_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 50, 100)           2000000
_________________________________________________________________
spatial_dropout1d_1 (Spatial (None, 50, 100)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 100)               80400
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 606
=================================================================
Total params: 2,081,006
Trainable params: 2,081,006
Non-trainable params: 0
_________________________________________________________________
None
```

In [36]:
```
# Training LSTM Model
epochs = 5
batch_size = 64

history = model.fit(X_train, Y_train, epochs=epochs, batch_size=batch_s
ize,validation_split=0.1)
```

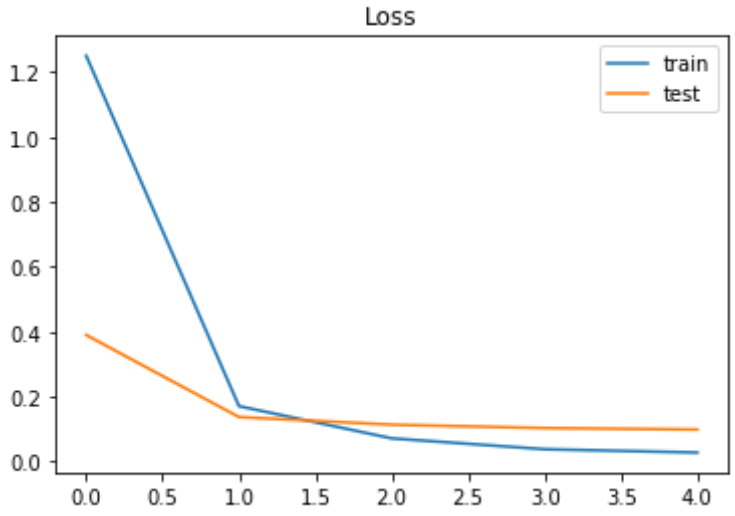
```
WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\tensorflow_core\python\ops\math_grad.py:1424: where (from tens
orflow.python.ops.array_ops) is deprecated and will be removed in a fut
ure version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:986: The name tf.assign_ad
d is deprecated. Please use tf.compat.v1.assign_add instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:973: The name tf.assign is
deprecated. Please use tf.compat.v1.assign instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:2741: The name tf.Session
is deprecated. Please use tf.compat.v1.Session instead.

Train on 6749 samples, validate on 750 samples
Epoch 1/5
WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:174: The name tf.get_defau
lt_session is deprecated. Please use tf.compat.v1.get_default_session i
nstead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:181: The name tf.ConfigPro
to is deprecated. Please use tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:190: The name tf.global_va
```

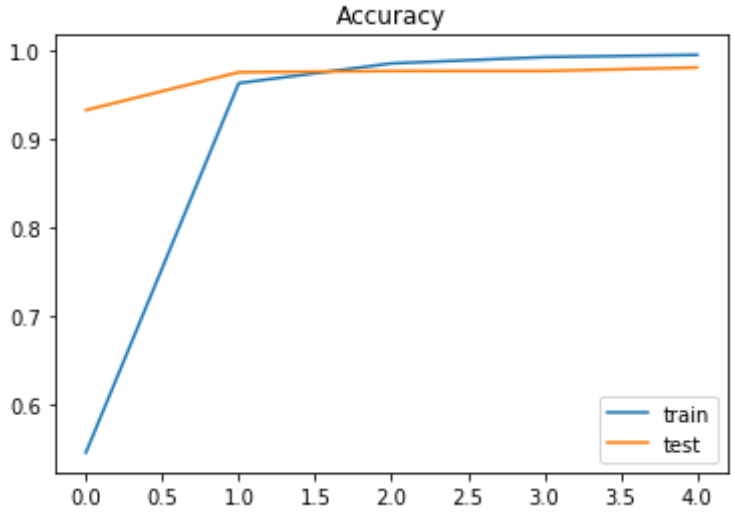
```
riables is deprecated. Please use tf.compat.v1.global_variables instea
d.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:199: The name tf.is_variab
le_initialized is deprecated. Please use tf.compat.v1.is_variable_initi
alized instead.

WARNING:tensorflow:From C:\Users\SHREE\.conda\envs\tensorflow\lib\site-
packages\keras\backend\tensorflow_backend.py:206: The name tf.variables
_initializer is deprecated. Please use tf.compat.v1.variables_initializ
er instead.

6749/6749 [==============================] - 18s 3ms/step - loss: 1.250
8 - acc: 0.5467 - val_loss: 0.3901 - val_acc: 0.9320
Epoch 2/5
6749/6749 [==============================] - 15s 2ms/step - loss: 0.170
7 - acc: 0.9624 - val_loss: 0.1371 - val_acc: 0.9747
Epoch 3/5
6749/6749 [==============================] - 15s 2ms/step - loss: 0.071
4 - acc: 0.9846 - val_loss: 0.1137 - val_acc: 0.9760
Epoch 4/5
6749/6749 [==============================] - 15s 2ms/step - loss: 0.038
0 - acc: 0.9917 - val_loss: 0.1032 - val_acc: 0.9760
Epoch 5/5
6749/6749 [==============================] - 16s 2ms/step - loss: 0.027
7 - acc: 0.9942 - val_loss: 0.0987 - val_acc: 0.9800
```

In [37]:

```
import matplotlib.pyplot as plt
plt.title('Loss')
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.legend()
plt.show();

plt.title('Accuracy')
plt.plot(history.history['acc'], label='train')
plt.plot(history.history['val_acc'], label='test')
```

```
plt.legend()
plt.show();
```





Performance Evaluation

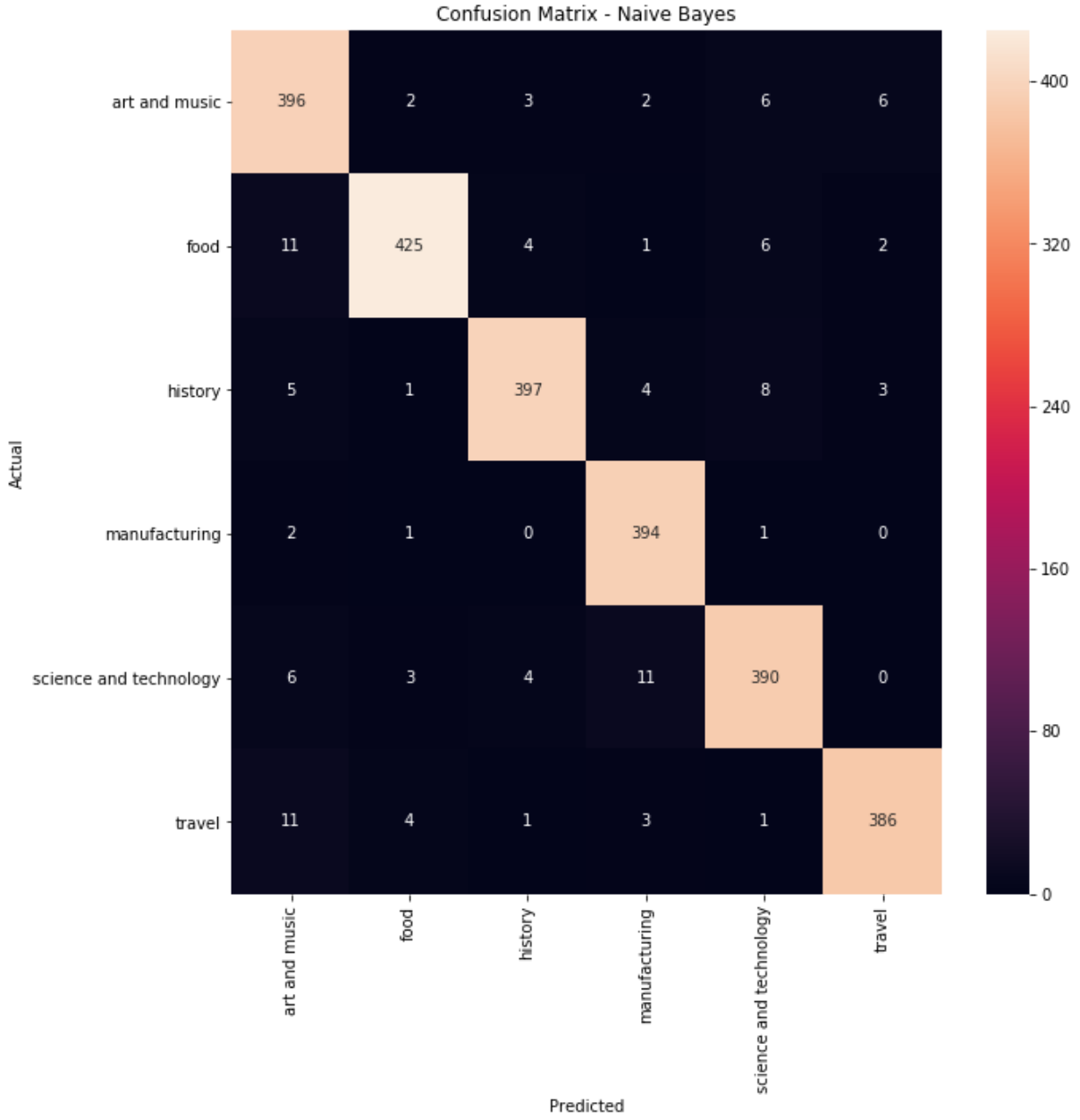
In [38]:
```
X_train, X_test, y_train, y_test = train_test_split(vdata.iloc[:, 1:3],
 vdata['Category'], random_state = 0)
X_test_title_features = tfidf_title.transform(X_test['Title']).toarray
()
X_test_desc_features = tfidf_desc.transform(X_test['Description']).toar
ray()
test_features = np.concatenate([X_test_title_features, X_test_desc_feat
ures], axis=1)
```

Naive Bayes

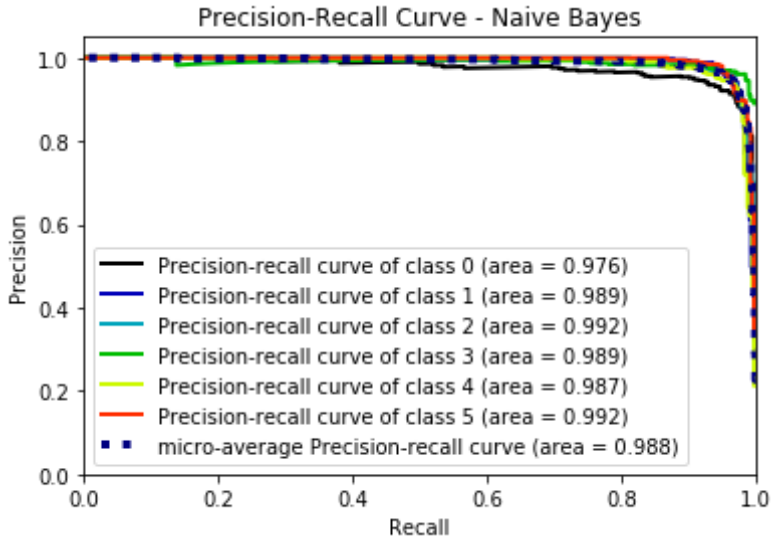
In [39]:
```
from sklearn import metrics
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
import scikitplot as skplt

X_test_title_features = tfidf_title.transform(X_test['Title']).toarray
()
X_test_desc_features = tfidf_desc.transform(X_test['Description']).toar
ray()
test_features = np.concatenate([X_test_title_features, X_test_desc_feat
ures], axis=1)

# Naive Bayes
y_pred = nb.predict(test_features)
y_probas = nb.predict_proba(test_features)

print(metrics.classification_report(y_test, y_pred,
                                    target_names=list(le.classes_)))

conf_mat = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='d', xticklabels=list(le.classes_
), yticklabels=list(le.classes_))
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.title('Confusion Matrix - Naive Bayes')
```

```
plt.show()

skplt.metrics.plot_precision_recall_curve(y_test, y_probas)
plt.title('Precision-Recall Curve - Naive Bayes')
plt.show()
```

	precision	recall	f1-score	support
art and music	0.92	0.95	0.94	415
food	0.97	0.95	0.96	449
history	0.97	0.95	0.96	418
manufacturing	0.95	0.99	0.97	398
science and technology	0.95	0.94	0.94	414
travel	0.97	0.95	0.96	406
accuracy			0.96	2500
macro avg	0.96	0.96	0.96	2500
weighted avg	0.96	0.96	0.96	2500



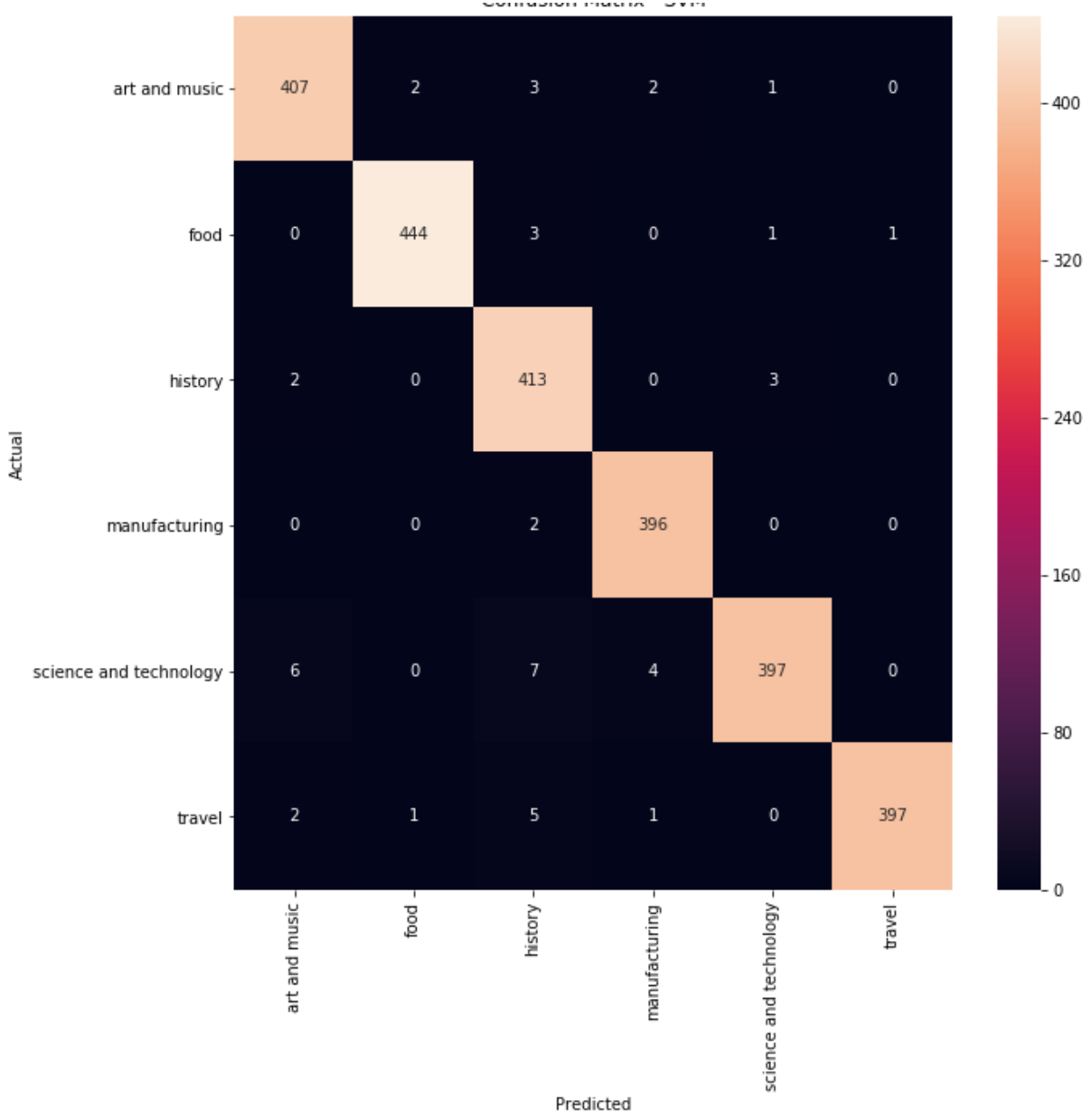
```
C:\Users\SHREE\.conda\envs\tensorflow\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning: Function plot_precision_recall_curve is deprecated; This will be removed in v0.5.0. Please use scikitplot.metrics.plot_precision_recall instead.
  warnings.warn(msg, category=FutureWarning)
```



SVM

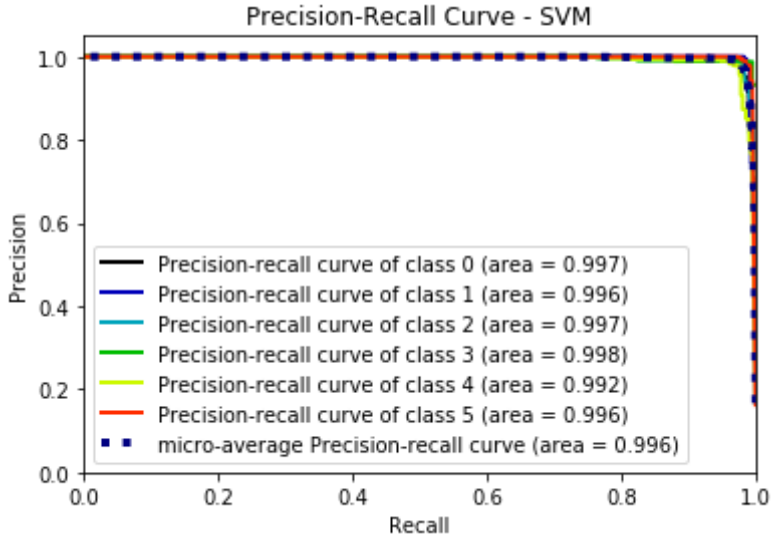
In [40]:
```
# SVM
y_pred = svm.predict(test_features)
y_probas = svm.predict_proba(test_features)

print(metrics.classification_report(y_test, y_pred,
                                    target_names=list(le.classes_)))

conf_mat = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='d', xticklabels=list(le.classes_
), yticklabels=list(le.classes_))
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.title('Confusion Matrix - SVM')
plt.show()

skplt.metrics.plot_precision_recall_curve(y_test, y_probas)
plt.title('Precision-Recall Curve - SVM')
plt.show()
```

```
                         precision    recall  f1-score   support

          art and music       0.98      0.98      0.98       415
                   food       0.99      0.99      0.99       449
                history       0.95      0.99      0.97       418
          manufacturing       0.98      0.99      0.99       398
science and technology       0.99      0.96      0.97       414
                 travel       1.00      0.98      0.99       406

               accuracy                           0.98      2500
              macro avg       0.98      0.98      0.98      2500
           weighted avg       0.98      0.98      0.98      2500
```




```
C:\Users\SHREE\.conda\envs\tensorflow\lib\site-packages\sklearn\utils\d
eprecation.py:87: FutureWarning: Function plot_precision_recall_curve i
s deprecated; This will be removed in v0.5.0. Please use scikitplot.met
```

```
rics.plot_precision_recall instead.
  warnings.warn(msg, category=FutureWarning)
```

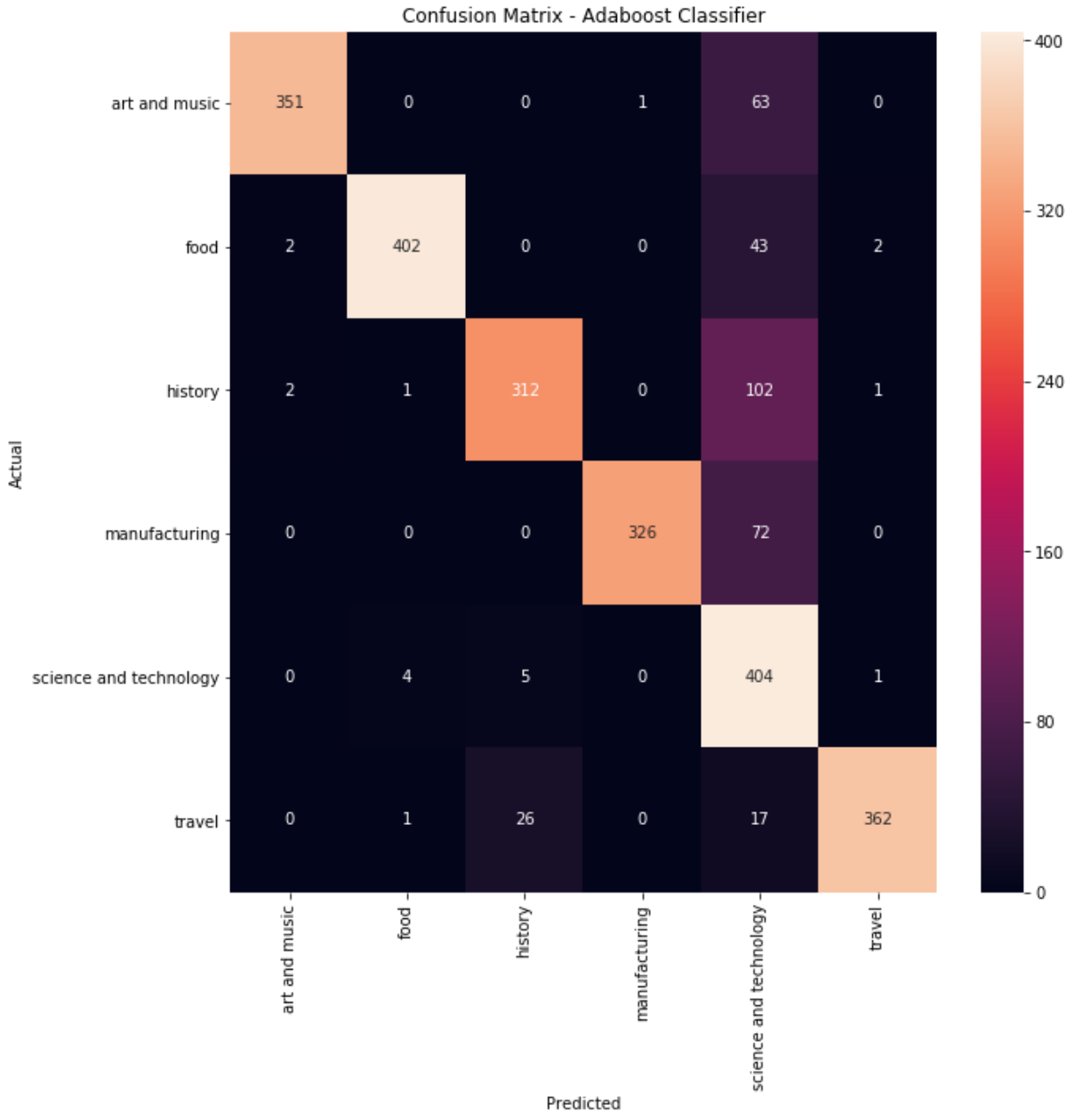


Adaboost Classifier

In [41]:
```
# Adaboost Classifier
y_pred = adaboost.predict(test_features)
y_probas = adaboost.predict_proba(test_features)

print(metrics.classification_report(y_test, y_pred,
                                    target_names=list(le.classes_)))

conf_mat = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='d', xticklabels=list(le.classes_
), yticklabels=list(le.classes_))
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.title('Confusion Matrix - Adaboost Classifier')
plt.show()
```

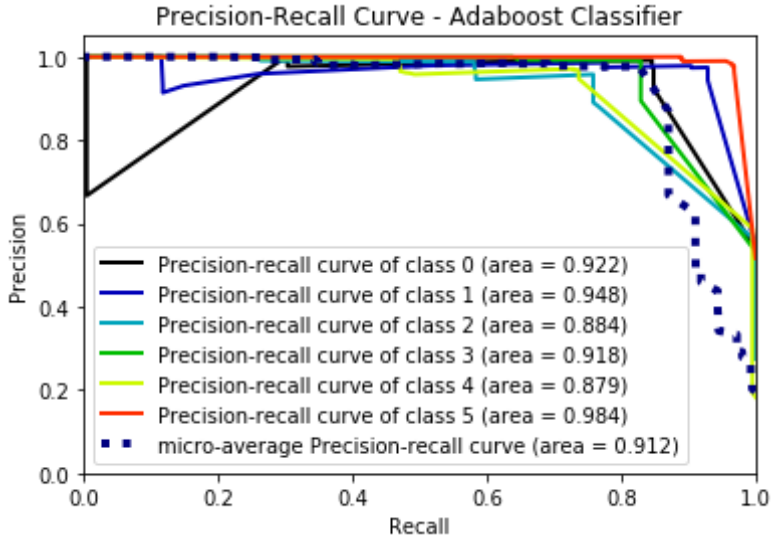
```
skplt.metrics.plot_precision_recall_curve(y_test, y_probas)
plt.title('Precision-Recall Curve - Adaboost Classifier')
plt.show()
```

	precision	recall	f1-score	support
art and music	0.99	0.85	0.91	415
food	0.99	0.90	0.94	449
history	0.91	0.75	0.82	418
manufacturing	1.00	0.82	0.90	398
science and technology	0.58	0.98	0.72	414
travel	0.99	0.89	0.94	406
accuracy			0.86	2500
macro avg	0.91	0.86	0.87	2500
weighted avg	0.91	0.86	0.87	2500




```
C:\Users\SHREE\.conda\envs\tensorflow\lib\site-packages\sklearn\utils\d
eprecation.py:87: FutureWarning: Function plot_precision_recall_curve i
```

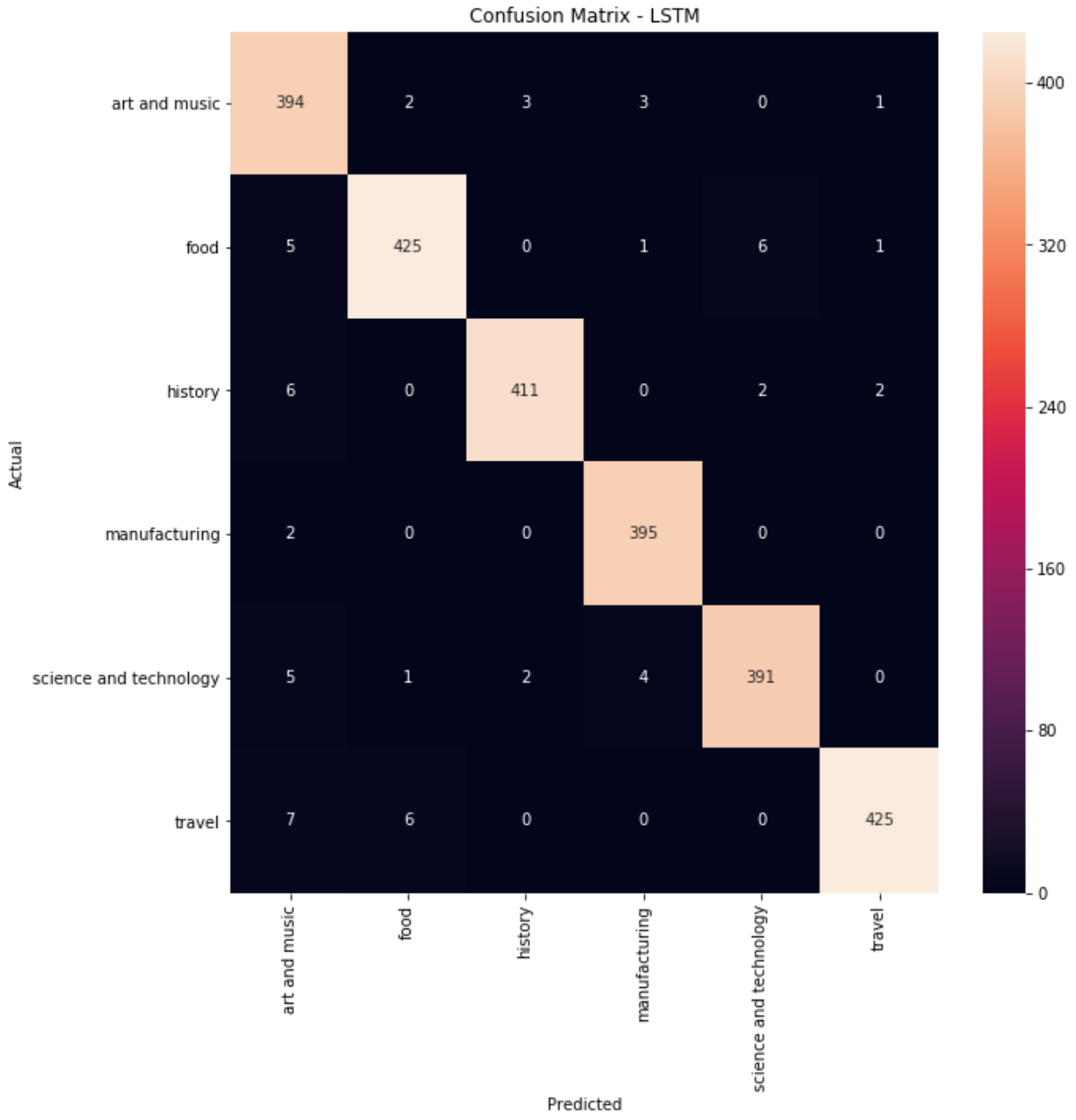
```
s deprecated; This will be removed in v0.5.0. Please use scikitplot.met
rics.plot_precision_recall instead.
  warnings.warn(msg, category=FutureWarning)
```



LSTM

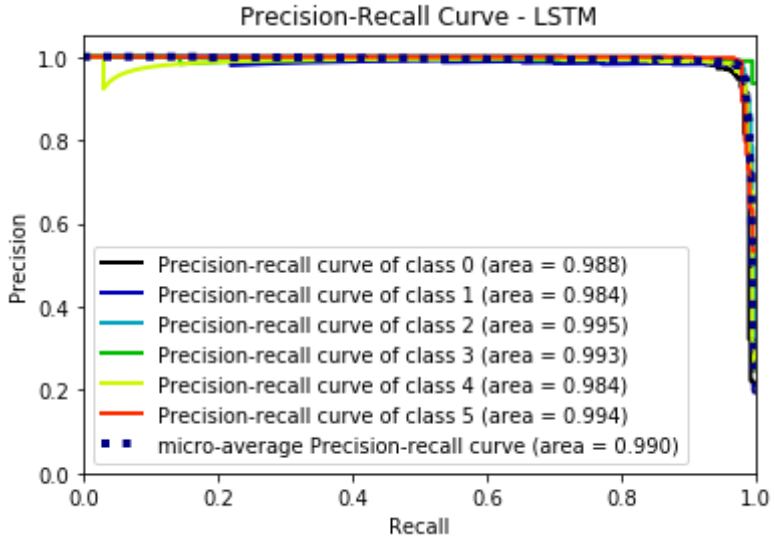
In [42]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, random_state =
 42)
y_probas = model.predict(X_test)
y_pred = np.argmax(y_probas, axis=1)
y_test = np.argmax(Y_test, axis=1)

print(metrics.classification_report(y_test, y_pred,
                                    target_names=list(le.classes_)))

conf_mat = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='d', xticklabels=list(le.classes_
), yticklabels=list(le.classes_))
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.title('Confusion Matrix - LSTM')
plt.show()

skplt.metrics.plot_precision_recall_curve(y_test, y_probas)
plt.title('Precision-Recall Curve - LSTM')
plt.show()
```

```
                         precision    recall  f1-score   support

          art and music       0.94      0.98      0.96       403
                   food       0.98      0.97      0.97       438
                history       0.99      0.98      0.98       421
          manufacturing       0.98      0.99      0.99       397
science and technology       0.98      0.97      0.98       403
                 travel       0.99      0.97      0.98       438

               accuracy                           0.98      2500
              macro avg       0.98      0.98      0.98      2500
           weighted avg       0.98      0.98      0.98      2500
```




```
C:\Users\SHREE\.conda\envs\tensorflow\lib\site-packages\sklearn\utils\d
```

```
eprecation.py:87: FutureWarning: Function plot_precision_recall_curve i
s deprecated; This will be removed in v0.5.0. Please use scikitplot.met
rics.plot_precision_recall instead.
  warnings.warn(msg, category=FutureWarning)
```



Importing Advertisement Dataset

Advertisement data is collected via web scrapping. The Data iscollected from www.adforum.com
Using brower extention and and the extractor tool Screaming Frog SEO spyder

In [52]:
```
#import data
adata = pd.read_csv('collected_sports_data.csv' )
```

In [53]:
```
adata.head(10)
```

Out[53]:

	Original Url	Title 1	Meta Description 1	H1-1	Unnamed: 4	Unnamed: 5	Un
0	https://www.adforum.com/creative-work/ad/playe...	Leos - "Inner Beauty"	Television Inner Beauty for Leos by Xynias, We...	Leos - "Inner Beauty"	NaN	NaN	
1	https://www.adforum.com/creative-work/ad/playe...	Ninemsn - "Indian Restaurant"	Television Indian Restaurant for Ninemsn by Ha...	Ninemsn - "Indian Restaurant"	NaN	NaN	
2	https://www.adforum.com/creative-work/ad/playe...	Social Democratic Party - "Social-Democratic P...	Television Social-Democratic Party for Social ...	Social Democratic Party - "Social-Democratic P...	NaN	NaN	
3	https://www.adforum.com/creative-work/ad/playe...	The Observer - "Sport The 1st"	Television Sport The 1st for The Observer by O...	The Observer - "Sport The 1st"	NaN	NaN	
4	https://www.adforum.com/creative-work/ad/playe...	Wind - "History Has Changed"	Television History Has Changed for Wind by TBW...	Wind - "History Has Changed"	NaN	NaN	
5	https://www.adforum.com/creative-work/ad/playe...	Nestlé - "Swimming"	Television Swimming for Nestlé by Publicis Ita...	Nestlé - "Swimming"	NaN	NaN	
6	https://www.adforum.com/creative-work/ad/playe...	Nestlé - "Sprint"	Television Sprint for Nestlé by Publicis Italy...	Nestlé - "Sprint"	NaN	NaN	

	Original Url	Title 1	Meta Description 1	H1-1	Unnamed: 4	Unnamed: 5	Un
7	https://www.adforum.com/creative-work/ad/playe...	Nestlé - "Volley"	Television Volley for Nestlé by Publicis Italy...	Nestlé - "Volley"	NaN	NaN	
8	https://www.adforum.com/creative-work/ad/playe...	Hertz - "Tennis"	Television Tennis for Hertz by Publicis Mojo	Hertz - "Tennis"	NaN	NaN	
9	https://www.adforum.com/creative-work/ad/playe...	"Natural High"	Television Natural High for by McCann Brisbane	"Natural High"	NaN	NaN	

10 rows × 42 columns

In [57]:
```
#delete columns which are not required
adata.drop(adata.iloc[:, 4:42], inplace = True, axis = 1)
```

In [58]:
```
adata.head(10)
```

Out[58]:

	Original Url	Title 1	Meta Description 1	H1-1
0	https://www.adforum.com/creative-work/ad/playe...	Leos - "Inner Beauty"	Television Inner Beauty for Leos by Xynias, We...	Leos - "Inner Beauty"
1	https://www.adforum.com/creative-work/ad/playe...	Ninemsn - "Indian Restaurant"	Television Indian Restaurant for Ninemsn by Ha...	Ninemsn - "Indian Restaurant"
2	https://www.adforum.com/creative-work/ad/playe...	Social Democratic Party - "Social-Democratic P...	Television Social-Democratic Party for Social ...	Social Democratic Party - "Social-Democratic P...

	Original Url	Title 1	Meta Description 1	H1-1
3	https://www.adforum.com/creative-work/ad/playe...	The Observer - "Sport The 1st"	Television Sport The 1st for The Observer by O...	The Observer - "Sport The 1st"
4	https://www.adforum.com/creative-work/ad/playe...	Wind - "History Has Changed"	Television History Has Changed for Wind by TBW...	Wind - "History Has Changed"
5	https://www.adforum.com/creative-work/ad/playe...	Nestlé - "Swimming"	Television Swimming for Nestlé by Publicis Ita...	Nestlé - "Swimming"
6	https://www.adforum.com/creative-work/ad/playe...	Nestlé - "Sprint"	Television Sprint for Nestlé by Publicis Italy...	Nestlé - "Sprint"
7	https://www.adforum.com/creative-work/ad/playe...	Nestlé - "Volley"	Television Volley for Nestlé by Publicis Italy...	Nestlé - "Volley"
8	https://www.adforum.com/creative-work/ad/playe...	Hertz - "Tennis"	Television Tennis for Hertz by Publicis Mojo	Hertz - "Tennis"
9	https://www.adforum.com/creative-work/ad/playe...	"Natural High"	Television Natural High for by McCann Brisbane	"Natural High"

Data Prepocessing and cleaning

In [59]:
```
# Change to lowercase
adata['Title 1'] = adata['Title 1'].map(lambda x: x.lower())
adata['Meta Description 1'] = adata['Meta Description 1'].map(lambda x:
 x.lower())
adata['H1-1'] = adata['H1-1'].map(lambda x: x.lower())

# Remove Punctuation
adata['Title 1'] = adata['Title 1'].map(lambda x: x.translate(x.maketra
ns('', '', string.punctuation)))
adata['Meta Description 1'] = adata['Meta Description 1'].map(lambda x:
 x.translate(x.maketrans('', '', string.punctuation)))
```

```
adata['H1-1'] = adata['H1-1'].map(lambda x: x.translate(x.maketrans('',
 '', string.punctuation)))

# Remove white spaces
adata['Title 1'] = adata['Title 1'].map(lambda x: x.strip())
adata['Meta Description 1'] = adata['Meta Description 1'].map(lambda x:
 x.strip())
adata['H1-1'] = adata['H1-1'].map(lambda x: x.strip())
```

In [60]:
```
adata.head(10)
```

Out[60]:

	Original Url	Title 1	Meta Description 1	H1-1
0	https://www.adforum.com/creative-work/ad/playe...	leos inner beauty	television inner beauty for leos by xynias wet...	leos inner beauty
1	https://www.adforum.com/creative-work/ad/playe...	ninemsn indian restaurant	television indian restaurant for ninemsn by ha...	ninemsn indian restaurant
2	https://www.adforum.com/creative-work/ad/playe...	social democratic party socialdemocratic party	television socialdemocratic party for social d...	social democratic party socialdemocratic party
3	https://www.adforum.com/creative-work/ad/playe...	the observer sport the 1st	television sport the 1st for the observer by o...	the observer sport the 1st
4	https://www.adforum.com/creative-work/ad/playe...	wind history has changed	television history has changed for wind by tbw...	wind history has changed
5	https://www.adforum.com/creative-work/ad/playe...	nestlé swimming	television swimming for nestlé by publicis ita...	nestlé swimming
6	https://www.adforum.com/creative-work/ad/playe...	nestlé sprint	television sprint for nestlé by publicis italy...	nestlé sprint
7	https://www.adforum.com/creative-work/ad/playe...	nestlé volley	television volley for nestlé by publicis italy...	nestlé volley

	Original Url	Title 1	Meta Description 1	H1-1
8	https://www.adforum.com/creative-work/ad/playe...	hertz tennis	television tennis for hertz by publicis mojo	hertz tennis
9	https://www.adforum.com/creative-work/ad/playe...	natural high	television natural high for by mccann brisbane	natural high

This function below mateches the unigram with advertisement data. And gives the output the url link of the advertisement related to the given keyword.

In [61]:
```
import pandas as pd
def find(dec,k):
    r=[]
    for i in dec.index:
        if k in dec['Meta Description 1'][i]:
            r.append(dec['Original Url'][i])
    return r


# Import Data
#adata = pd.read_csv('collected_sports_data.csv' )
adata=adata[['Original Url', 'Meta Description 1']]

#Search unigram keyword which is extracted from videos data.

result=find(adata, "travel")
for i in result:
    print(" Url Link ",i)
```

```
 Url Link  https://www.adforum.com/creative-work/ad/player/34458334/time-travel/directv
 Url Link  https://www.adforum.com/creative-work/ad/player/34465481/rugby-world/gullivers-sports-travel
 Url Link  https://www.adforum.com/creative-work/ad/player/34485545/travel/ole
 Url Link  https://www.adforum.com/creative-work/ad/player/34488240/new-zealand/expedia-com
```

```
 Url Link  https://www.adforum.com/creative-work/ad/player/34485545/tra
vel/ole
 Url Link  https://www.adforum.com/creative-work/ad/player/34488240/new
-zealand/expedia-com
 Url Link  https://www.adforum.com/creative-work/ad/player/34517107/lea
ve-your-mark/under-armour
 Url Link  https://www.adforum.com/creative-work/ad/player/34518055/tim
e-traveller/sky
 Url Link  https://www.adforum.com/creative-work/ad/player/34519534/do-
it-for-mom/spies
 Url Link  https://www.adforum.com/creative-work/ad/player/34526061/tra
vel-alberta-case-study-thrill/travel-alberta
 Url Link  https://www.adforum.com/creative-work/ad/player/34529381/rel
ax-travel/airbnb
 Url Link  https://www.adforum.com/creative-work/ad/player/34590024/hel
ping-disappointed-u-s-soccer-fans-find-a-new-fandom/kayak
 Url Link  https://www.adforum.com/creative-work/ad/player/6698342/gras
s/travelers
```

The above links in the output re-directs to the advertisement video.

END

In [ ]: