Natural Language Processing (COMM061)

Part 2

GROUP 21

TEAM MEMBERS: KRISHNAKUMAR KANNAN, ASHITHA REJITH, ARAVIND RAJU

Research Article Classification

In the world of advanced science and technology where research driven scientific knowledge is prioritized over traditional concepts, there is a healthy competition among scientists, and academics interested in the educational sector to publish their findings and works in scientific journals. With more funds being dedicated to research and related studies there are a number of articles about all subjects published in various journals at an alarming rate.

For a layman and other academicians who aren't experts in each of these sectors there is a necessity to classify these article to its respective genre based on its general summary. On this behalf our objective is to develop a NLP model which can classify an article given its abstract and title to its most similar and identical genre. The data-set used in the model is downloaded from kaggle.com. It consists of the title and abstract of articles from 6 various fields namely Computer Science, Physics, Mathematics, Statistics, Quantitative Biology and Quantitative Finance.

Our aim is to classify the text input given by the user and to predict the most similar genre among the 6 to which the work is comprised of.

Data-set

The dataset comprises around 30000 research articles which fall under a wide variety of topics namely Physics, Statistics, Mathematics, Quantitative Biology & Quantitative Finance. The aim of the model is to develop a prototype that classifies an unseen article into one or more of the mentioned topics.

In [1]:
```
import pandas as pd
import numpy as np
import os
import string
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import nltk
import matplotlib.pyplot as plt
nltk.download('stopwords')
nltk.download('wordnet')
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
import matplotlib.pyplot as plt
```

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

Loading Data

In [2]:

```
dataset=pd.read_csv('Research_Article_train.csv')
#dataset.head(15)

dataset.head(5)
```

Out[2]:

	ID	TITLE	ABSTRACT	Computer Science	Physics	Mathematics	Statistics	Quantitative Biology
0	1	Reconstructing Subject-Specific Effect Maps	Predictive models allow subject-specific inf...	1	0	0	0	0
1	2	Rotation Invariance Neural Network	Rotation invariance and translation invarian...	1	0	0	0	0
2	3	Spherical polyharmonics and Poisson kernels fo...	We introduce and develop the notion of spher...	0	0	1	0	0
3	4	A finite element approximation for the stochas...	The stochastic Landau--Lifshitz--Gilbert (LL...	0	0	1	0	0
4	5	Comparative study of Discrete Wavelet Transfor...	Fourier-transform infra-red (FTIR) spectra o...	1	0	0	1	0

In [3]:
```
dataset['ID']=dataset['ID'].astype(float)
dataset['Computer Science']=dataset['Computer Science'].astype(
dataset['Physics']=dataset['Physics'].astype(float)
dataset['Mathematics']=dataset['Mathematics'].astype(float)
dataset['Statistics']=dataset['Statistics'].astype(float)
dataset['Quantitative Biology']=dataset['Quantitative Biology']
dataset['Quantitative Finance']=dataset['Quantitative Finance']
dataset.dtypes
```

Out[3]:
```
ID                       float64
TITLE                     object
ABSTRACT                  object
Computer Science         float64
Physics                  float64
Mathematics              float64
Statistics               float64
Quantitative Biology     float64
Quantitative Finance     float64
dtype: object
```

In [4]:
```
y=dataset[['Computer Science', 'Physics', 'Mathematics',
       'Statistics', 'Quantitative Biology', 'Quantitative Fina
```

In [5]:
```
#combining 2 text columns title and abstract into one and drop
dataset['Text']=dataset['TITLE']+' '+dataset['ABSTRACT']
dataset.drop(columns=['TITLE','ABSTRACT'], inplace=True)
#dataset.head(5)
```

Data Pre-processing

Input dataset will be undergoing some prerprocessing to get a perfect model with maximum performance.

Following steps will be done to make preprocess the dependent variable (comment_text)

🔷 Replace newline,punctuation, tabs and digits with white spaces

🔷 Convert all string to lower case

🔷 Split the text into words

🔷 Apply stemming Lemmatization to each words and remove stop words from the sentence.

🔷 After applying this filters, this words are joined and attached to the same data frame

In [6]:

```
remove_punc = string.punctuation
def remove_punctuation(text):
    return text.translate(str.maketrans('', '', remove_punc))
```

In [7]:

```
stopword = set(stopwords.words('english'))
def remove_stopwords(text):
    """custom function to remove the stopwords"""
    return " ".join([word for word in str(text).split() if word
```

In [8]:

```
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
def stem_words(text):
    return " ".join([stemmer.stem(word) for word in text.split(
```

In [9]:

```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
def lemmatize_words(text):
    return " ".join([lemmatizer.lemmatize(word) for word in tex
```

Defining a function for preprocessing

In [10]:

```
def preprocessing(dataset):
    #convert to string type
    dataset['Text'] = dataset['Text'].astype(str)
    #convert to the lowercase
    dataset["Text"] = dataset["Text"].str.lower()
    #remove punctuations
    dataset["Text"] = dataset["Text"].apply(lambda text: remove_
    #stopwords removal
    dataset["Text"] = dataset["Text"].apply(lambda text: remove_
    #Remove Numbers
    dataset['Text'] =dataset["Text"].str.replace('\d+', '')
    #stemming
    #dataset["Text"] = dataset["Text"].apply(lambda text: stem_
    #lemmatisation
    dataset["Text"] = dataset["Text"].apply(lambda text: lemmat
    return dataset
```

In [11]:

```
import warnings
warnings.filterwarnings('ignore')
processed_data=preprocessing(dataset)
```

In [33]:
```
clean_data=processed_data[['Text','Computer Science','Physics','
clean_data.head(5)
```

Out[33]:

	Text	Computer Science	Physics	Mathematics	Statistics	Quantitative Biology	Quantitative Finance
0	reconstructing subjectspecific effect map pred...	1.0	0.0	0.0	0.0	0.0	0.0
1	rotation invariance neural network rotation in...	1.0	0.0	0.0	0.0	0.0	0.0
2	spherical polyharmonics poisson kernel polyhar...	0.0	0.0	1.0	0.0	0.0	0.0
3	finite element approximation stochastic maxwel...	0.0	0.0	1.0	0.0	0.0	0.0
4	comparative study discrete wavelet transforms ...	1.0	0.0	0.0	1.0	0.0	0.0

Manipulating Dataset

We will balance the dataset by maintaining a ratio of atleat 20-80 percentage between true label and false label so that we will overcome the class imbalance issue.

Steps involved are:

🔷 Split the whole dataset into 6 categories of one label each.

🔷 Balance each of lables in the data set based on 0 and 1 values.

🔷 Pickle and CI CD pipe line is established

In [13]:
```
df_cs=processed_data[['Text','Computer Science']]
df_p=processed_data[['Text','Physics']]
df_m=processed_data[['Text','Mathematics']]
df_s=processed_data[['Text','Statistics']]
df_qb=processed_data[['Text','Quantitative Biology']]
df_qf=processed_data[['Text','Quantitative Finance']]
df_s
```

Out[13]:

	Text	Statistics
0	reconstructing subjectspecific effect map pred...	0.0
1	rotation invariance neural network rotation in...	0.0
2	spherical polyharmonics poisson kernel polyhar...	0.0
3	finite element approximation stochastic maxwel...	0.0
4	comparative study discrete wavelet transforms ...	1.0
...	...	...
20967	contemporary machine learning guide practition...	0.0
20968	uniform diamond coating wcco hard alloy cuttin...	0.0
20969	analysing soccer game clustering conceptors pr...	0.0
20970	efficient simulation lefttail sum correlated l...	1.0
20971	optional stopping problem bayesians recently o...	1.0

20972 rows × 2 columns

Computer Science

For Computer Science data we have 10000+ data for 1 so we take 6000 data each for 0 and 1

In [14]:
```
df_cs_1 = df_cs[df_cs['Computer Science'] == 1].iloc[0:6000,:]
df_cs_1.shape
```

Out[14]: (6000, 2)

In [15]:
```
df_cs_0 = df_cs[df_cs['Computer Science'] == 0].iloc[0:6000,:]
df_cs_done = pd.concat([df_cs_1, df_cs_0], axis=0)
df_cs_done.shape
```

Out[15]: (12000, 2)

Physics

For physics we took 6000 data each for 0 and 1

In [16]:
```
df_p[df_p['Physics'] == 1].count()

```

Out[16]:
```
Text       6013
Physics    6013
dtype: int64
```

In [17]:
```
df_phy_1 = df_p[df_p['Physics'] == 1].iloc[0:6000,:]
df_phy_0 = df_p[df_p['Physics'] == 0].iloc[0:6000,:]
df_phy_done = pd.concat([df_phy_1, df_phy_0], axis=0)
df_phy_done.shape
df_phy_done
```

Out[17]:

	Text	Physics
6	rotation period shape hyperbolic asteroid ioum...	1.0
7	adverse effect polymer coating heat transport ...	1.0
8	sph calculation marsscale collision role equat...	1.0
11	roleseparating ordering social dilemma control...	1.0
12	dynamic exciton magnetic polarons cdmnsecdmgse...	1.0
...	...	...
8359	committee machine computational statistical ga...	0.0
8361	exponentially small splitting separatrix near ...	0.0
8362	learning dynamic coevolution competing sexual ...	0.0
8363	deep neural network multiple speaker detection...	0.0
8364	active tolerant testing work give first algori...	0.0

12000 rows × 2 columns

Mathematics

For Mathematics we took 5618 data each for 0 and 1

In [18]:
```
df_m[df_m['Mathematics'] == 1].count()
```

Out[18]:
```
Text           5618
Mathematics    5618
dtype: int64
```

In [19]:
```
df_m_1 = df_m[df_m['Mathematics'] == 1].iloc[0:5618,:]
df_m_0 = df_m[df_m['Mathematics'] == 0].iloc[0:5618,:]
df_m_done = pd.concat([df_m_1, df_m_0], axis=0)
df_m_done.shape
df_m_done
```

Out[19]:

	Text	Mathematics
2	spherical polyharmonics poisson kernel polyhar...	1.0
3	finite element approximation stochastic maxwel...	1.0
5	maximizing fundamental frequency complement ob...	1.0
15	rank waring decomposition smlangle rangle symm...	1.0
17	higher structure unstable adam spectral sequen...	1.0
...	...	...
7724	asymptotic distribution simultaneous confidenc...	0.0
7726	projected variational integrator degenerate la...	0.0
7727	boosted generative model propose novel approac...	0.0
7729	overlapping community detection using superior...	0.0
7730	debugging transaction tracking provenance reen...	0.0

11236 rows × 2 columns

Statistics

For Statistics we took 5206 data each for 0 and 1

In [20]:
```
df_s[df_s['Statistics'] == 1].count()
```

Out[20]:
```
Text          5206
Statistics    5206
dtype: int64
```

In [21]:
```
df_s_1 = df_s[df_s['Statistics'] == 1].iloc[0:5206,:]
df_s_0 = df_s[df_s['Statistics'] == 0].iloc[0:5206,:]
df_s_done = pd.concat([df_s_1, df_s_0], axis=0)
df_s_done.shape
df_s_done
```

Out[21]:

	Text	Statistics
4	comparative study discrete wavelet transforms ...	1.0
18	comparing covariate prioritization via matchin...	1.0
28	minimax estimation l distance consider problem...	1.0
30	mixup beyond empirical risk minimization large...	1.0
40	covariance robustness variational bayes meanfi...	1.0
...	...	...
6971	nonequilibrium work hamiltonian connection mic...	0.0
6972	thick subcategories stable category module ext...	0.0
6973	planetdriven spiral arm protoplanetary disk ii...	0.0
6974	ideal structure pure infiniteness ample groupo...	0.0
6975	nonparametric mean curvature type flow graph c...	0.0

10412 rows × 2 columns

Quantitative Biology

For Quantitative Biology we took 587 data each for 0 and 1

In [22]:
```
df_qb[df_qb['Quantitative Biology'] == 1].count()
```

Out[22]:
```
Text                    587
Quantitative Biology    587
dtype: int64
```

In [23]:
```
df_qb_1 = df_qb[df_qb['Quantitative Biology'] == 1].iloc[0:587,
df_qb_0 = df_qb[df_qb['Quantitative Biology'] == 0].iloc[0:2348
df_qb_done = pd.concat([df_qb_1, df_qb_0], axis=0)
df_qb_done.shape
df_qb_done
```

Out[23]:

	Text	Quantitative Biology
9	mathcalr fails predict outbreak potential pres...	1.0
20	deciphering noise amplification reduction open...	1.0
33	unsupervised homogenization pipeline clusterin...	1.0
55	competing evolutionary path growing population...	1.0
115	gene regulatory network inference introductory...	1.0
...	...	...
2418	streaming kernel pca tildeosqrtn random featur...	0.0
2419	universal protocol information dissemination u...	0.0
2420	note specie realization nondegeneracy potentia...	0.0
2421	unified stochastic formulation dissipative qua...	0.0
2422	vortex state spin texture rotating spinorbitco...	0.0

2935 rows × 2 columns

Quantitative Finance

For Quantitative Finance we took 249 data each for 0 and 1

In [24]:
```
df_qf[df_qf['Quantitative Finance'] == 1].count()
```

Out[24]:
```
Text                    249
Quantitative Finance    249
dtype: int64
```

In [25]:
```
df_qf_1 = df_qf[df_qf['Quantitative Finance'] == 1].iloc[0:249,
df_qf_0 = df_qf[df_qf['Quantitative Finance'] == 0].iloc[0:996,
df_qf_done = pd.concat([df_qf_1, df_qf_0], axis=0)
df_qf_done.shape
df_qf_done
```

Out[25]:

	Text	Quantitative Finance
41	multifactor gaussian term structure model stil...	1.0
266	high dimensional estimation multifactor model ...	1.0
268	expanded local variance gamma model paper prop...	1.0
492	psychological model investor manager behavior ...	1.0
622	failure smooth pasting principle nonexistence ...	1.0
...	...	...
1003	high temperature thermodynamics honeycomblatti...	0.0
1004	laplace beltrami operator baran metric pluripo...	0.0
1005	magnetic polarons nonequilibrium polariton con...	0.0
1006	inference sparse graph pairwise measurement si...	0.0
1007	oracle importance sampling stochastic simulati...	0.0

1245 rows × 2 columns

Performance of the Classifier

Example :

Input Text

"In machine learning, the task of classification means to use the available data to learn a function which can assign a category to a data point. For example, assign a genre to a movie, like "Romantic Comedy", "Action", "Thriller". Another example could be automatically assigning a category to news articles, like "Sports" and "Politics"."

OutPut Predictions percentage:

🔷 Computer Science, 0.53

🔷 Physics 0.07,

🔷 Mathematics 0.22,

🔷 Statistics 0.77,

🔷 Quantitative Biology 0.15,

🔷 Quantitative Finance' 0.11

Pickling

Pickle renders Python object structures in serial and de-serialized formats. You can pickle any object in Python to save it on disk. Pickle first "serializes" the object before writing it to file.Python pickling is the process of converting a python object (list, dict, etc.) into a character stream. This character stream contains all the information needed to reconstruct the object in another python script.

In [26]:
```
import pickle

from sklearn.linear_model import LogisticRegression
from sklearn.multiclass import OneVsRestClassifier
def pickle_model(df, label):

    X = df.Text
    y = df[label]

    # Initiate a Tfidf vectorizer
    tfv = TfidfVectorizer(ngram_range=(1,1), stop_words='englis

    # Convert the X data into a document term matrix dataframe
    X_vect = tfv.fit_transform(X)

    # saves the column labels (ie. the vocabulary)
    # wb means Writing to the file in Binary mode, written in b
    with open(r"{}.pkl".format('_pickles/'+label + '_vect'), "w
        pickle.dump(tfv, f)

    #randomforest = RandomForestClassifier(n_estimators=100, ra
    #randomforest.fit(X_vect, y)

    # define model
    model = LogisticRegression()
    # define the ovr strategy
    logR = OneVsRestClassifier(model)
    # fit model
    logR.fit(X_vect,y)


    # Create a new pickle file based on random forest
    with open(r"{}.pkl".format('_pickles/'+label + '_model'), "
        pickle.dump(logR, f)
```

In [28]:
```
datalist = [df_cs_done, df_phy_done, df_m_done, df_s_done, df_q
label = ['Computer Science','Physics','Mathematics','Statistics

for i,j in zip(datalist,label):
    pickle_model(i, j)
```

In [ ]:
```

```

CI CD Pipeline

The CI CD stands for Continuous Integration and Continuous Delivery.A practical application of Continuous Integration is to implement small changes, and have the code check in to repositories frequently. This ensures that the code developed across different platforms is integrated. Continuous Delivery involves automation of delivery to selected infrastructure environments.Thus, the code is pushed automatically.

In [29]:
```
from sklearn.pipeline import Pipeline
from joblib import dump
```

In [30]:
```

pipeline = Pipeline(steps= [('tfidf', TfidfVectorizer(min_df=5, max_
                            ('model', LogisticRegression())])
```

In [31]:
```
def create_pipeline(df, label,pipeline):

    X = df.Text
    y = df[label]
    filename=r"{}.joblib".format('_Pipelines/'+label + '_pipeli
    dump(pipeline, filename=filename)
```

In [32]:
```
for i,j in zip(datalist,label):
    create_pipeline(i, j,pipeline)
```

Conclusion

The deployed model has a 64% accuracy which is not the best that can be achieved improve its performance the data set can be processed using better vectorization techniques like word embodding,word2vec,doc2vec and better training models like neural network can be used.

The training data seta can be improved by including more data for all the labels to avoid the imbalance issue also memorizing the data saved from user inputs including more features with higher engrams

Improve the prediction ability in which the model can recognize the subject context and evaluate the content probability of the whole sentence rather than by specific words

In [ ]:
```

```