Text Classification of News Articles

MODEL 1

In [290]:

```
#PACKAGES
import pandas as pd
import warnings
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import RegexpTokenizer
from sklearn.feature_extraction.text import TfidfVectorizer
import numpy as np
from sklearn.feature_selection import chi2
from sklearn.manifold import TSNE
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
warnings.simplefilter('ignore')
```

In [291]:

```
# Load Data
df=pd.read_csv("BBC News.csv")
df.shape
```

Out[291]:

```
(1490, 3)
```

In [292]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1490 entries, 0 to 1489
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   ArticleId  1490 non-null   int64 
 1   Text       1490 non-null   object
 2   Category   1490 non-null   object
dtypes: int64(1), object(2)
memory usage: 35.0+ KB
```

In [293]:

```
df.head()
```

Out[293]:

	ArticleId	Text	Category
0	1833	worldcom ex-boss launches defence lawyers defe...	business
1	154	german business confidence slides german busin...	business
2	1101	bbc poll indicates economic gloom citizens in ...	business
3	1976	lifestyle governs mobile choice faster bett...	tech
4	917	enron bosses in $168m payout eighteen former e...	business

In [294]:

```
#Unique values
print(df['Category'].value_counts())
print(df['Category'].value_counts().sum())
```

```
sport            346
business         336
politics         274
entertainment    273
tech             261
Name: Category, dtype: int64
1490
```

In [295]:

```
#Check any duplicates
df[df.duplicated()]
```

Out[295]:

	ArticleId	Text	Category

In [296]:

```
#Check any nulls
df.isna().sum()
```

Out[296]:

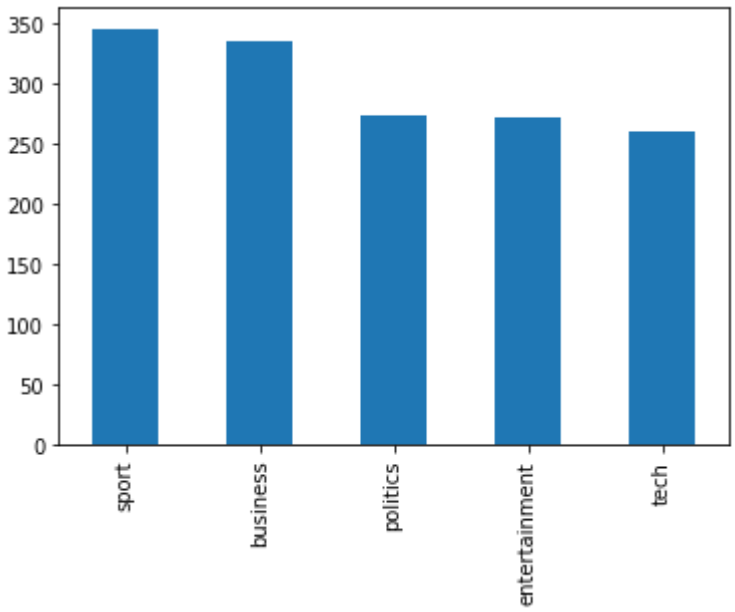
```
ArticleId    0
Text         0
Category     0
dtype: int64
```

In [297]:

```
#Ploting
from matplotlib import pyplot as plot
df['Category'].value_counts().plot(kind='bar')
```

Out[297]:

<AxesSubplot:>



In [298]:

```
#Preprocessing
```

In [299]:

```
df['category_id'] = df['Category'].factorize()[0]
colslist = ['Index', 'Text', 'Category', 'category_id']
df.columns = colslist
```

In [300]:

```
df.head()
```

Out[300]:

	Index	Text	Category	category_id
0	1833	worldcom ex-boss launches defence lawyers defe...	business	0
1	154	german business confidence slides german busin...	business	0
2	1101	bbc poll indicates economic gloom citizens in ...	business	0
3	1976	lifestyle governs mobile choice faster bett...	tech	1
4	917	enron bosses in $168m payout eighteen former e...	business	0

In [301]:

```
#Stop words
swords=stopwords.words('english')
```

In [302]:

```
df['news_without_stopwords'] = df['Text'].apply(lambda x: ' '.join([word for word in x.s
```

In [303]:

```
#Stemming
st=PorterStemmer()
```

In [304]:

```
df['news_porter_stemmed'] = df['news_without_stopwords'].apply(lambda x: ' '.join([st.st
```

In [305]:

```
df['news_porter_stemmed'] = df['news_porter_stemmed'].apply(lambda x: ' '.join(x.lower()
```

In [306]:

```
df['news_porter_stemmed'] = df['news_porter_stemmed'].str.replace('[^\w\s]','')
```

In [307]:

```
#Vectorizing

tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', encoding='latin-1', ngra
features = tfidf.fit_transform(df.news_porter_stemmed).toarray()#
labels = df.category_id
features.shape
```

Out[307]:

```
(1490, 9784)
```

In [308]:

```
freq = pd.Series(' '.join(df['news_porter_stemmed']).split()).value_counts()
freq.head()
```

Out[308]:

```
said     4838
mr       2006
would    1711
year     1701
also     1426
dtype: int64
```

In [309]:

```
freq2 = freq[freq <= 3]
freq2
```

Out[309]:

```
standout    3
alegr       3
quash       3
unearth     3
mileston    3
           ..
earning     1
60p         1
sexes       1
average     1
tail        1
Length: 15120, dtype: int64
```

In [310]:

```
freq3 = list(freq2.index.values)
freq3
```

```
 'deputies',
 'boys',
 'gallas',
 'publicis',
 'flee',
 'stead',
 'diseases',
 'overtur',
 'canni',
 'highrisk',
 'ballerina',
 'thirdquart',
 'rato',
 'resent',
 'radiu',
 'exchief',
 'casinos',
 'difficulties',
 '107bn',
 'raised',
```

In [311]:

```
df['news_porter_stemmed'] = df['news_porter_stemmed'].apply(lambda x: ' '.join([word for
```

In [312]:

```
df = df[['Index', 'Category', 'category_id', 'news_porter_stemmed']]
df
```

Out[312]:

	Index	Category	category_id	news_porter_stemmed
0	1833	business	0	worldcom launch defenc lawyer defend former wo...
1	154	business	0	german busi confid slide german busi confid fe...
2	1101	business	0	bbc poll indic econom citizen major nation sur...
3	1976	tech	1	lifestyl govern mobil choic faster better hard...
4	917	business	0	enron boss 168m payout eighteen former enron d...
...	...	...	...	...
1485	857	entertainment	4	doubl evict big brother model capric citi acto...
1486	325	entertainment	4	dj doubl act revamp chart show dj duo jk joel ...
1487	1590	business	0	weak dollar hit reuter revenu media group reut...
1488	1587	tech	1	appl ipod famili expand market appl expand ipo...
1489	538	tech	1	santi worm make visit thousand websit bulletin...

1490 rows × 4 columns

In [313]:

```
tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', encoding='latin-1', ngra
```

In [314]:

```
features = tfidf.fit_transform(df.news_porter_stemmed).toarray()
labels = df.category_id
features.shape
```

Out[314]:

```
(1490, 9901)
```

In [315]:

```
df.columns = ['Index', 'Category', 'category_id','news_porter_stemmed']
```

In [316]:

```
category_id_df = df[['Category', 'category_id']].drop_duplicates().sort_values('category
category_to_id = dict(category_id_df.values)
id_to_category = dict(category_id_df[['category_id', 'Category']].values)
```

In [317]:

```
N = 3
for newstype, category_id in sorted(category_to_id.items()):
    features_chi2 = chi2(features, labels == category_id)
    indices = np.argsort(features_chi2[0])
    feature_names = np.array(tfidf.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(newstype))
    print("  . Most correlated unigrams:\n       . {}".format('\n       . '.join(unigram
    print("  . Most correlated bigrams:\n       . {}".format('\n       . '.join(bigrams[
```

```
# 'business':
  . Most correlated unigrams:
       . profit
       . growth
       . bank
  . Most correlated bigrams:
       . econom growth
       . interest rate
       . analyst said
# 'entertainment':
  . Most correlated unigrams:
       . actor
       . star
       . film
  . Most correlated bigrams:
       . lo angel
       . best film
       . box offic
# 'politics':
  . Most correlated unigrams:
       . blair
       . tori
       . labour
  . Most correlated bigrams:
       . lib dem
       . toni blair
       . mr blair
# 'sport':
  . Most correlated unigrams:
       . coach
       . cup
       . champion
  . Most correlated bigrams:
       . grand slam
       . australian open
       . six nation
# 'tech':
  . Most correlated unigrams:
       . softwar
       . comput
       . user
  . Most correlated bigrams:
       . peopl use
       . let peopl
       . mobil phone
```

In [318]:

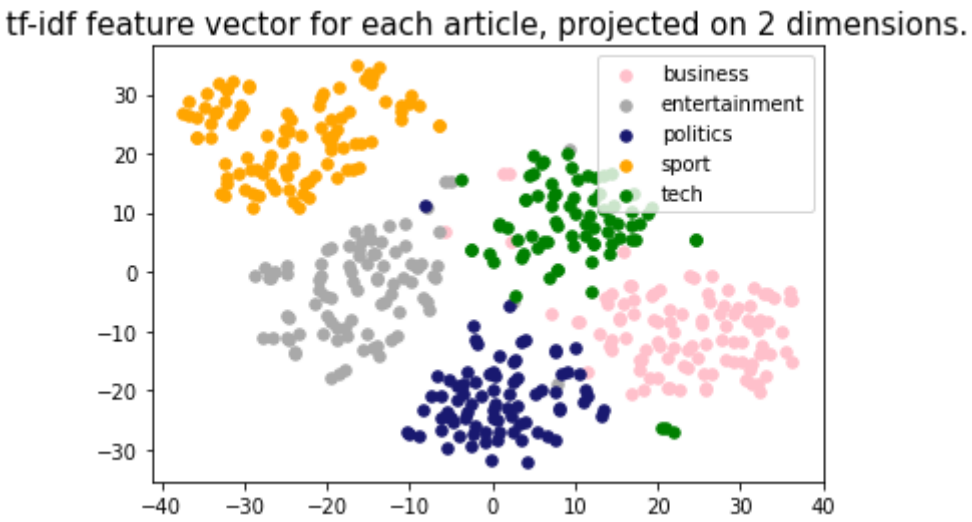
```
# Sampling a subset of our dataset because t-SNE is computationally expensive
SAMPLE_SIZE = int(len(features) * 0.3)
np.random.seed(0)
indices = np.random.choice(range(len(features)), size=SAMPLE_SIZE, replace=False)
projected_features = TSNE(n_components=2, random_state=0).fit_transform(features[indices
colors = ['pink', 'green', 'midnightblue', 'orange', 'darkgrey']
```

In [319]:

```
for category, category_id in sorted(category_to_id.items()):
    points = projected_features[(labels[indices] == category_id).values]
    plot.scatter(points[:, 0], points[:, 1], s=30, c=colors[category_id], label=category
plot.title("tf-idf feature vector for each article, projected on 2 dimensions.",
          fontdict=dict(fontsize=15))
plot.legend()
```

Out[319]:

```
<matplotlib.legend.Legend at 0x274b12ae5f8>
```



In [320]:

```
#MODEL BUILDING
```

In [321]:

```
model = LogisticRegression(random_state=0)

X_train, X_test, y_train, y_test, indices_train, indices_test = train_test_split(feature
model.fit(X_train, y_train)
y_pred_proba = model.predict_proba(X_test)
y_pred = model.predict(X_test)
print('LogisticRegression Accuracy:',metrics.accuracy_score(y_test,y_pred))
```

```
LogisticRegression Accuracy: 0.975609756097561
```

In [322]:

```
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
# Model Generation Using Multinomial Naive Bayes
clf = MultinomialNB().fit(X_train, y_train)
predicted= clf.predict(X_test)
print("MultinomialNB Accuracy:",metrics.accuracy_score(y_test, predicted))
```

```
MultinomialNB Accuracy: 0.9695121951219512
```

In [328]:

```
#TEST
texts = ["Hooli stock price soared after a dip in PiedPiper revenue growth.",
         "Captain Tsubasa scores a magnificent goal for the Japanese team.",
         "Merryweather mercenaries are sent on another mission, as government oversight
         "Beyoncé releases a new album, tops the charts in all of south-east Asia!",
         "You won't guess what the latest trend in data analysis is!"]



text_features = tfidf.transform(texts)
print(text_features.shape)
predictions = model.predict(text_features)

for text, predicted in zip(texts, predictions):
    print('"{}"'.format(text))
    print("  - Predicted as: '{}'".format(id_to_category[predicted]))
    print("")
```

```
(5, 9901)
"Hooli stock price soared after a dip in PiedPiper revenue growth."
  - Predicted as: 'business'

"Captain Tsubasa scores a magnificent goal for the Japanese team."
  - Predicted as: 'sport'

"Merryweather mercenaries are sent on another mission, as government over
sight groups call for new sanctions."
  - Predicted as: 'business'

"Beyoncé releases a new album, tops the charts in all of south-east Asi
a!"
  - Predicted as: 'entertainment'

"You won't guess what the latest trend in data analysis is!"
  - Predicted as: 'sport'
```

MODEL 2

In [359]:

```
df2=pd.read_csv("BBC News.csv")
token=RegexpTokenizer(r'[a-zA-Z0-9]+')
cv=CountVectorizer(stop_words='english',ngram_range=(1,1),tokenizer=token.tokenize,max_f
# text_counts=cv.fit_transform(df2['Text'])
y = np.array(df.Category.values)
# cv = CountVectorizer(max_features = 5000)
x = cv.fit_transform(df2.Text).toarray()

x_train2,x_test2,y_train2,y_test2=train_test_split(x,y,test_size=0.25,random_state=5)
from sklearn.naive_bayes import MultinomialNB
MNB=MultinomialNB()
MNB.fit(x_train2,y_train2)
pred2=MNB.predict(x_test2)
accs2=metrics.accuracy_score(pred2,y_test2)
print("Accuracy:",accs2,"\n")
texts = ["Hooli stock price soared after a dip in PiedPiper revenue growth.",
         "Captain Tsubasa scores a magnificent goal for the Japanese team.",
         "Merryweather mercenaries are sent on another mission, as government oversight
         "Beyoncé releases a new album, tops the charts in all of south-east Asia!",
         "You won't guess what the latest trend in data analysis is!"]

y_pred1 = cv.transform(texts)

yy = MNB.predict(y_pred1)
for text, pred in zip(texts, yy):
    print('"{}"'.format(text))
    print("  - Predicted as: '{}'".format(pred))
    print("")
```

```
Accuracy: 0.9812332439678284

"Hooli stock price soared after a dip in PiedPiper revenue growth."
  - Predicted as: 'business'

"Captain Tsubasa scores a magnificent goal for the Japanese team."
  - Predicted as: 'sport'

"Merryweather mercenaries are sent on another mission, as government over
sight groups call for new sanctions."
  - Predicted as: 'politics'

"Beyoncé releases a new album, tops the charts in all of south-east Asi
a!"
  - Predicted as: 'entertainment'

"You won't guess what the latest trend in data analysis is!"
  - Predicted as: 'tech'
```