```
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
import nltk
import sklearn

from tensorflow import keras.
from keras.preprocessing.text import text_to_word_sequence
from nltk.corpus import stopwords.
from nltk.stem import porterstemmer
from nltk.stem import wordnetlemmatizer
from nltk.corpus import wordnet
from sklearn.feature_extraction.text import Tfidfvectorizer
from sklearn.feature_extraction.text import TfidTransformer
from sklearn.feature_extraction.text import count vectorizer
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.naive_bayes import multinomial NB
from sklearn import svm
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import roc_acsc_score
from sklearn.pipeline import pipeline
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.tree import decision Tree classifier
from sklearn.ensemble import RandomForest Classifier
dataset = pd.read_csv('/content/BBC News.csv')
```

```
test_set = pd.read_csv("/content/BBC News.csv")
```

```
dataset.head()
```

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

```
target_category = dataset['category'].unique()
print(target_category)
```

```
['business' 'tech' 'politics' 'sport' 'entertainment']
```

```
dataset['categoryId'] = dataset['Category'].factorize()[0]
```

```
dataset.head()
```

	Article ID	Text	Category	Category ID
0	1833	worldcom ex-boss launches defence lawyers def..	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	Business	0

```
Category = dataset[["category", 'categoryId']].drop_duplicates().sort_values('categoryId')
Category
```

	Category	CategoryId
0	business	0

dataset.groupby('category').categoryId.count()

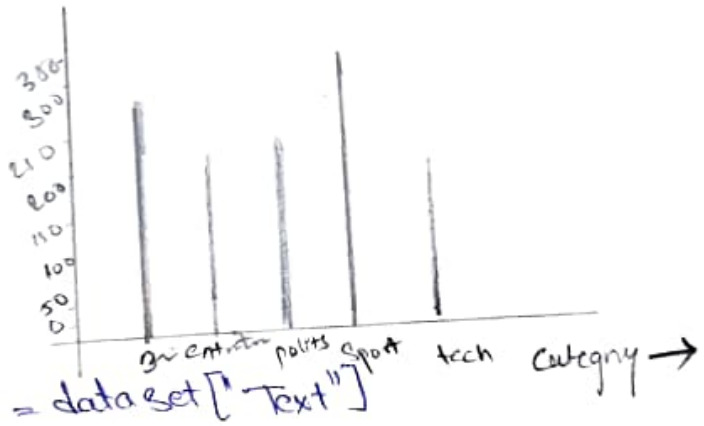
Category

Business	336
entertainment	273
politics	274
sport	346
tech	261

Name: categoryId, dtype: int64

DATA VISUALIZATION

dataset.groupby('category').categoryId.count().plot.bar(ylim=0)



text = dataset["Text"]

text.head()

0 worldcom ex-boss launches defence lawyer defe...
1 german business confidence slides german busin...
2 bbc poll indicates economic gloom citizen in ...
3 lifestyle governs mobile choice faster bett...
4 enron bosses in $168m payout eighteen former...

Name: Text, dtype: object

```
Category = dataset["Category"]
Category.head()
```

```
0    busines
1    Business
2    busines
3      tech
4    busines.
Name: Category, dtype: object
```

Data preprocessing

```
def preprocessDataset(train_text):
    # word tokenization using text_to_word_sequence.
    train_text = str(train_text)
    tokenized_trainset = text_to_word_sequence(train_text, filters='!"#$%&()*+,
                                                -./:;<=>?@[\]^_`{|}~\t\n', lower=True, split=" ")

    # Stop word removal
    stop_words = set(stopwords.words('english'))
    stopwordremove = [i for i in tokenized_train_set if not i in stop_words]
    #join words into sentence
    stopwordremove_text = ' '.join(stopwordremove)

    # remove numbers
    numberremove_text = ''.join(c for c stopwordremove_text if not c.isdigit())

    # --- Stemming ---
    stemmer = porterStemmer()
    steminput = nltk.word_tokenize(numberremove_text)
    stem_text = ' '.join(stemmer.stem(word) for word in steminput)
    lemmatizer = wordnet lemmatizer()
    def get_wordnet_pos(word)
```

```
""" map POS tag to first character lemmatize() accepts"""
    tag = nltk.pos_tag([word])[0][1][0].upper()
    tag_dict = {"J": wordnet.ADJ,
                "N": wordnet.NOUN,
                "V": wordnet.VERB,
                "R": wordnet.ADV}
    return tag_dict.get(tag, wordnet.NOUN)

lem_input = nltk.word_tokenize(stem_text)
lem_text = ' '.join([lemmatizer.lemmatize(w, get_wordnet_pos(w)) for w in lem_input])

return lem_text
```

SPLIT TRAIN SET

```
X_train, X_test, Y_train, Y_test = train_test_split(text, category, test_size=0.3,
                                   random_state=60, shuffle=True, stratify=category)


print(len(X_train))
print(len(X_test))
```

1043

447

MULTINOMIAL NAIVE BAYES

```
nb = Pipeline([('tfidf', TfidfVectorizer()),
               ('clf', MultinomialNB())])

nb.fit(X_train, Y_train)
test_predict = nb.predict(X_test)
train_accuracy = round(nb.score(X_train, Y_train)*100)
test_accuracy = round(accuracy_score(test_predict, Y_test)*100)
print("Naive Bayes Train Accuracy Score: {}%".format(train_accuracy))
print("Naive Bayes Test Accuracy Score: {}%".format(test_accuracy))
print()
```

print(classification_report(test_predict, Y_test, target_names = target_category))

Naive Bayes Train Accuracy Score : 98%

Naive Bayes Test Accuracy score : 96%.

	precision	recall	f1-score	support
business	0.98	0.94	0.96	105
tech	0.89	1.00	0.94	73
politics	0.94	0.92	0.93	84
sport	1.00	0.98	0.99	106
entertainment	0.95	0.94	0.94	79
accuracy			0.96	447
macro avg	0.95	0.96	0.95	447
weighted avg	0.96	0.96	0.96	447

Decision Tree

```
dt = pipeline([('tfidf', TfidfVectorizer()), ('dt', DecisionTreeClassification())])
dt.fit(X_train, Y_train)
test_predict = dt.predict(X_test)
train_accuracy = round(dt.score(X_train, Y_train)*100)
test_accuracy = round(accuracy_score(test_predict, Y_test)*100)
print("Decision Tree Train Accuracy Score : {}%".format(train_accuracy))
print("Decision Tree Test Accuracy Score : {}%".format(test_accuracy))
print()
print(classification_report(test_predict, Y_test, target_name = target_category))
```

Decision Tree Train Accuracy Score : 100%

Decision Tree test Accuracy Score : 76%.

	Precision	recall	f1-score	support
business	0.68	0.73	0.71	94
tech	0.68	0.74	0.71	76
politics	0.78	0.76	0.77	84
sport	0.91	0.79	0.84	121
entertainment	0.73	0.79	0.76	72
accuracy			0.76	447
macro avg	0.76	0.76	0.76	447
weighted avg	0.77	0.76	0.77	447

RANDOM FOREST CLASSIFIER

```
rfc = pipeline([('tfidf', Tfidfvectorizer()),('rfc', RandomForestClassifier(n_estimators=100))])
rfc.fit(x_train, y_train)
test_predict = rfc.predict(x_test)
train_accuracy = round(rfc.score(x_train, y_train)*100)
test_accuracy = round(accuracy_score(test_predict, y_test)*100)
print("K-Nearest Neighbour Train Accuracy Score :{} %".format(train_accuracy))
print("K-Nearest Neighbour Test Accuracy Score :{}%".format(test_accuracy))
print()
print(classification_report(test_predict, y_test, target_names=target_category))
```

K-Nearest Neighbour Train Accuracy Score :100%

K-Nearest Neighbour Test Accuracy Score :93%

	precision	recall	f1-score	support
business	0.97	0.88	0.92	112
tech	0.88	1.00	0.94	72
politics	0.85	0.93	0.89	75

Sport	1.00	0.93	0.96	112
entertainment	0.92	0.95	0.94	76
accuracy			0.93	447
macro avg	0.93	0.94	0.93	447
Weighted avg	0.94	0.93	0.93	447

TEST SET

test_set.head()

	Article ID	Text	category
0	1833	worldcom ex-boss launches defen...	business
1	154	german business confidence ...	business
2	1101	bbc poll indicates ...	business
3	1976	lifestyle governs mobile choice...	tech
4	917	enron bosses in $168m payout...	business