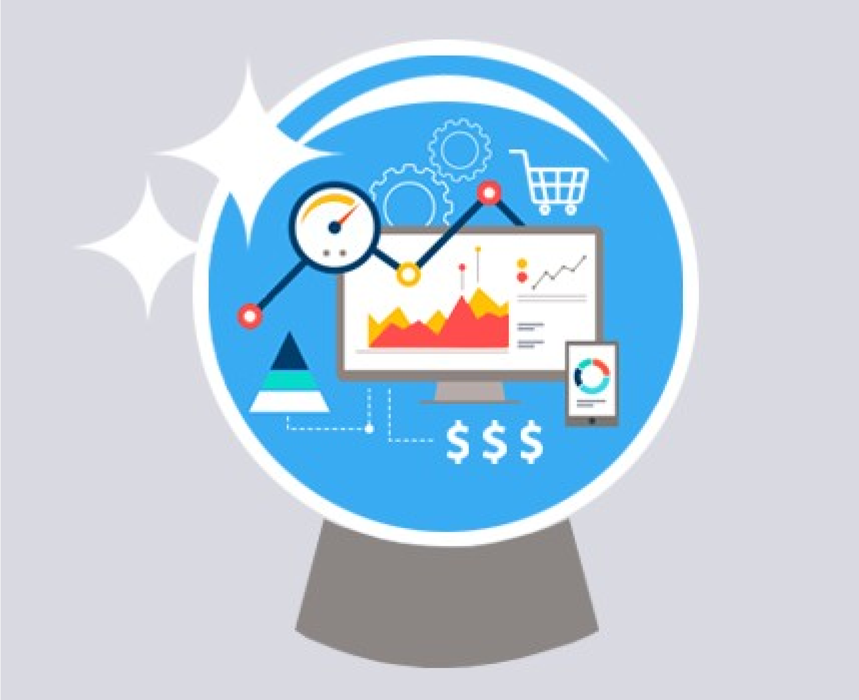
AI *driven exploration and prediction of company registration trends with registrar of companies* (ROC)

TEAM MEMBER

623021106043:RANJITHKUMAR.P

PHASE 2 SUBMISSION DOCUMENT



INTRODUCTION

- **Data exploration definition: Data exploration refers to the initial step in data analysis in which data analysts use data visualization and statistical techniques to describe dataset characterizations, such as size, quantity,** and accuracy, in order to better understand the nature of the data.

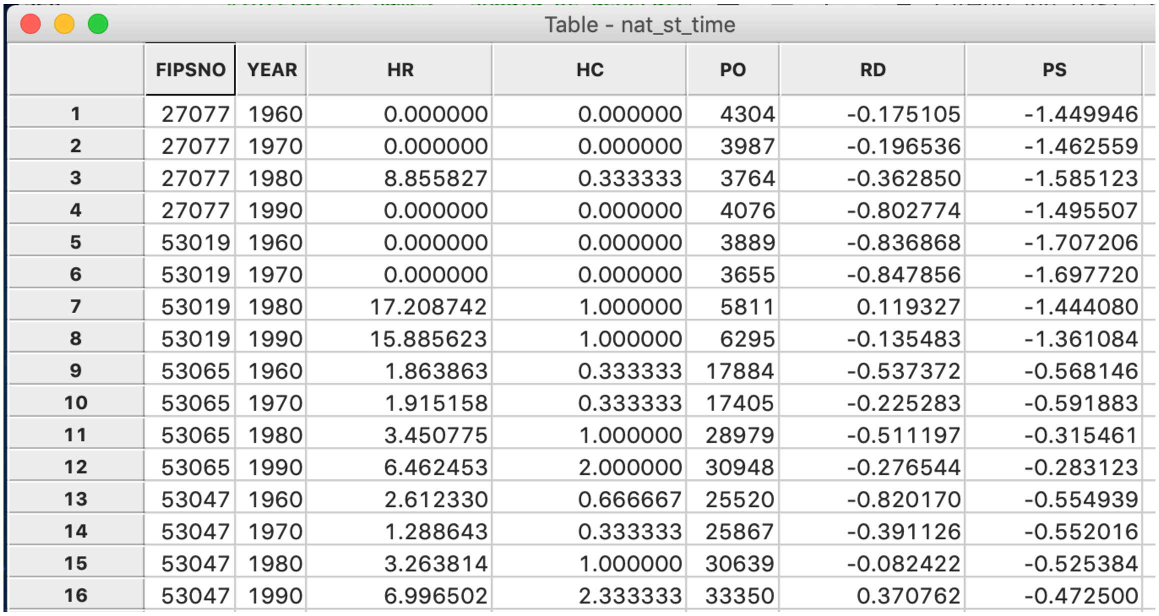
- Data exploration techniques include both manual analysis and automated data exploration software solutions that visually explore and identify relationships between different data variables, the structure of the dataset, the presence of outliers, and the distribution of data values in order to reveal patterns and points of interest, enabling data analysts to gain greater insight into the raw data.
- Data is often gathered in large, unstructured volumes from various sources and data analysts must first understand and develop a comprehensive view of the data before extracting relevant data for further analysis, such as univariate, bivariate, multivariate, and principal components analysis.

CONTENT FOR PROJECT PHASE 2:

consider exploring advanced AI algorithm like time series forecasting or ensemble methods for inproved predictive accuracy

DATA SOURCE:

DATASET LINK:(https://www.kaggle.com/datasets/thedevastator/analysis-of-coronary-artery-disease-risk-factors/data**)**



	FIPSNO	YEAR	HR	HC	PO	RD	PS
1	27077	1960	0.000000	0.000000	4304	-0.175105	-1.449946
2	27077	1970	0.000000	0.000000	3987	-0.196536	-1.462559
3	27077	1980	8.855827	0.333333	3764	-0.362850	-1.585123
4	27077	1990	0.000000	0.000000	4076	-0.802774	-1.495507
5	53019	1960	0.000000	0.000000	3889	-0.836868	-1.707206
6	53019	1970	0.000000	0.000000	3655	-0.847856	-1.697720
7	53019	1980	17.208742	1.000000	5811	0.119327	-1.444080
8	53019	1990	15.885623	1.000000	6295	-0.135483	-1.361084
9	53065	1960	1.863863	0.333333	17884	-0.537372	-0.568146
10	53065	1970	1.915158	0.333333	17405	-0.225283	-0.591883
11	53065	1980	3.450775	1.000000	28979	-0.511197	-0.315461
12	53065	1990	6.462453	2.000000	30948	-0.276544	-0.283123
13	53047	1960	2.612330	0.666667	25520	-0.820170	-0.554939
14	53047	1970	1.288643	0.333333	25867	-0.391126	-0.552016
15	53047	1980	3.263814	1.000000	30639	-0.082422	-0.525384
16	53047	1990	6.996502	2.333333	33350	0.370762	-0.472500

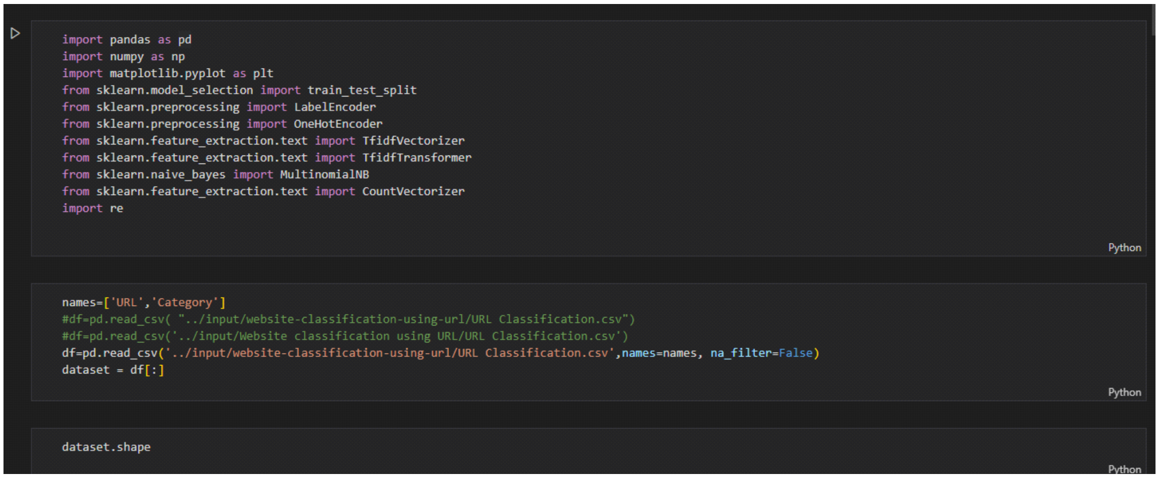
DATA COLLECTION AND PREPROCESSING:

- Collect historical ROC company registration data. This data can be obtained from the ROC website, or from other sources such as commercial data providers.
- Split the data into training and testing sets. The training set will be used to train the model, and the testing set will be used to evaluate the model's performance on unseen data.

ADVANCED AI ALGORITHM:

- Linear Regression.
- Logistic Regression.
- Decision Tree.
- SVM.
- Naive Bayes.
- kNN.
- K-Means.
- Random Forest

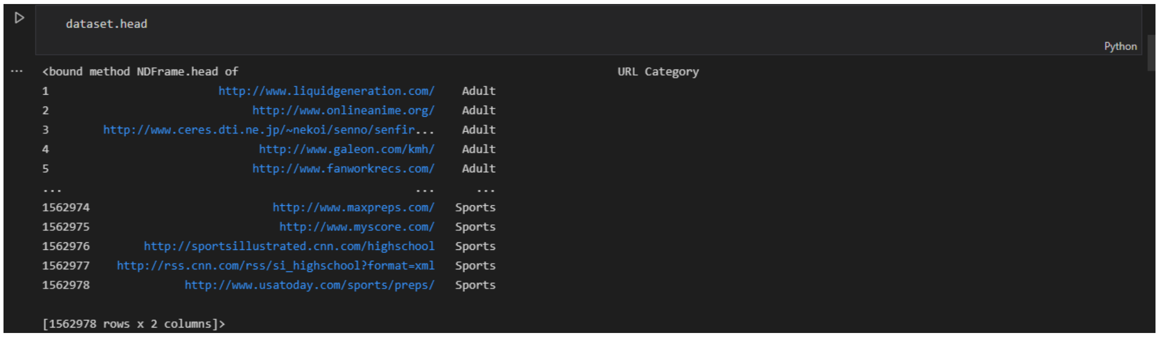
PROGRAM:



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer
import re
```

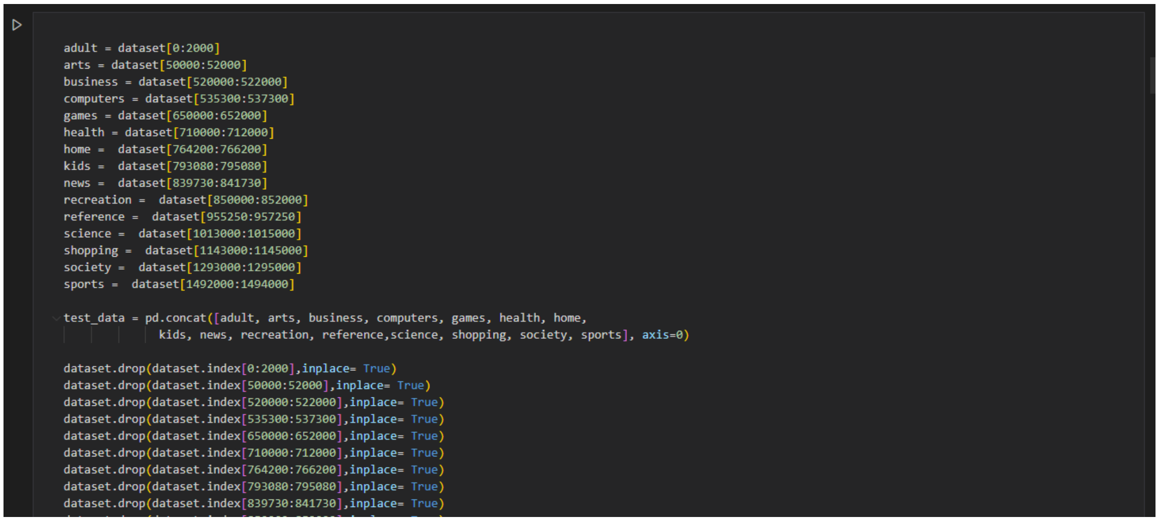
```python
names=['URL','Category']
#df=pd.read_csv( "../input/website-classification-using-url/URL Classification.csv")
#df=pd.read_csv('../input/Website classification using URL/URL Classification.csv')
df=pd.read_csv('../input/website-classification-using-url/URL Classification.csv',names=names, na_filter=False)
dataset = df[:]
```

```python
dataset.shape
```

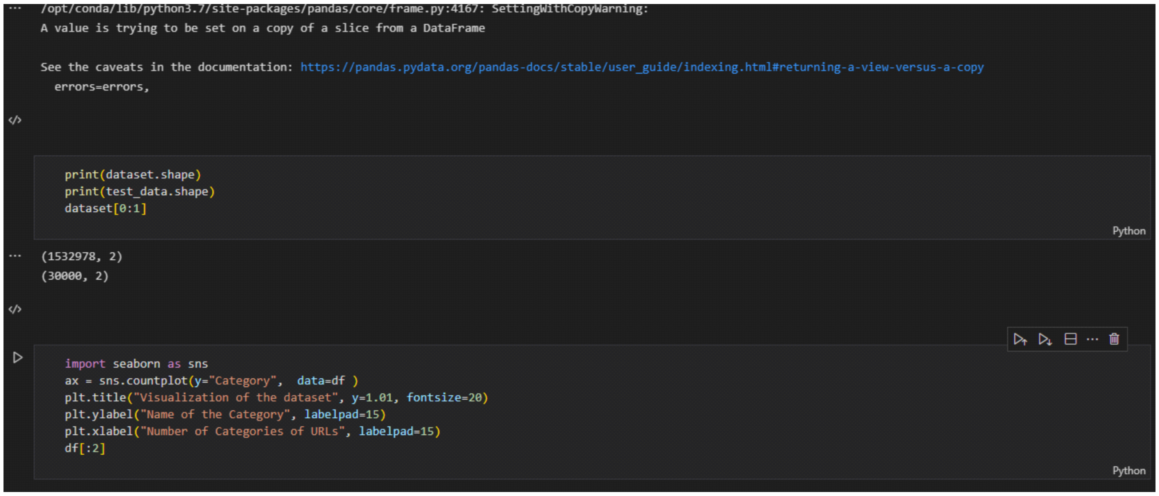


```python
dataset.head
```

```
<bound method NDFrame.head of                                                        URL Category
1                        http://www.liquidgeneration.com/    Adult
2                             http://www.onlineanime.org/    Adult
3         http://www.ceres.dti.ne.jp/~nekoi/senno/senfir...    Adult
4                              http://www.galeon.com/kmh/    Adult
5                             http://www.fanworkrecs.com/    Adult
...                                                     ...      ...
1562974                          http://www.maxpreps.com/   Sports
1562975                           http://www.myscore.com/   Sports
1562976        http://sportsillustrated.cnn.com/highschool   Sports
1562977    http://rss.cnn.com/rss/si_highschool?format=xml   Sports
1562978              http://www.usatoday.com/sports/preps/   Sports

[1562978 rows x 2 columns]>
```



```python
adult = dataset[0:2000]
arts = dataset[50000:52000]
business = dataset[520000:522000]
computers = dataset[535300:537300]
games = dataset[650000:652000]
health = dataset[710000:712000]
home =  dataset[764200:766200]
kids =  dataset[793080:795080]
news =  dataset[839730:841730]
recreation =  dataset[850000:852000]
reference =  dataset[955250:957250]
science =  dataset[1013000:1015000]
shopping =  dataset[1143000:1145000]
society =  dataset[1293000:1295000]
sports =  dataset[1492000:1494000]

test_data = pd.concat([adult, arts, business, computers, games, health, home,
             kids, news, recreation, reference,science, shopping, society, sports], axis=0)

dataset.drop(dataset.index[0:2000],inplace= True)
dataset.drop(dataset.index[50000:52000],inplace= True)
dataset.drop(dataset.index[520000:522000],inplace= True)
dataset.drop(dataset.index[535300:537300],inplace= True)
dataset.drop(dataset.index[650000:652000],inplace= True)
dataset.drop(dataset.index[710000:712000],inplace= True)
dataset.drop(dataset.index[764200:766200],inplace= True)
dataset.drop(dataset.index[793080:795080],inplace= True)
dataset.drop(dataset.index[839730:841730],inplace= True)
```

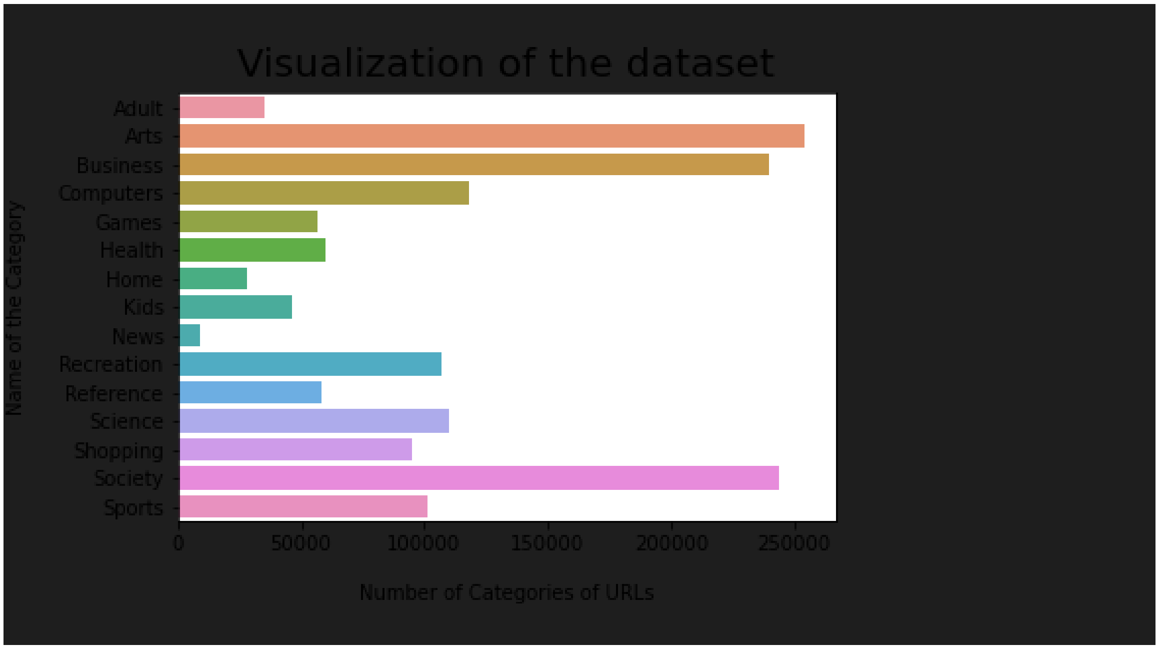


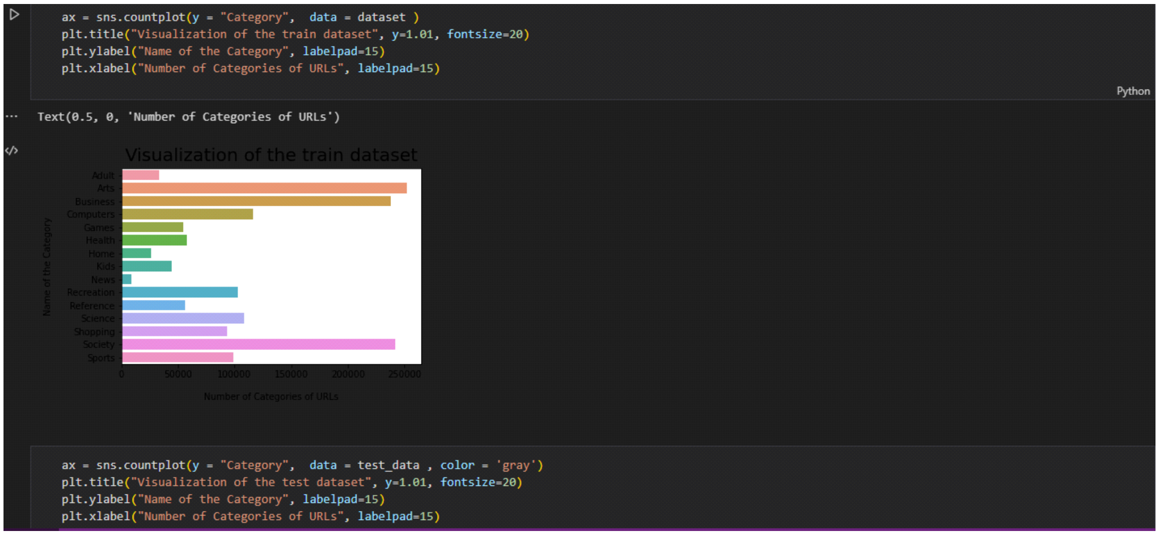
```
/opt/conda/lib/python3.7/site-packages/pandas/core/frame.py:4167: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  errors=errors,
```

```python
print(dataset.shape)
print(test_data.shape)
dataset[0:1]
```

```
(1532978, 2)
(30000, 2)
```

```python
import seaborn as sns
ax = sns.countplot(y="Category",  data=df )
plt.title("Visualization of the dataset", y=1.01, fontsize=20)
plt.ylabel("Name of the Category", labelpad=15)
plt.xlabel("Number of Categories of URLs", labelpad=15)
df[:2]
```

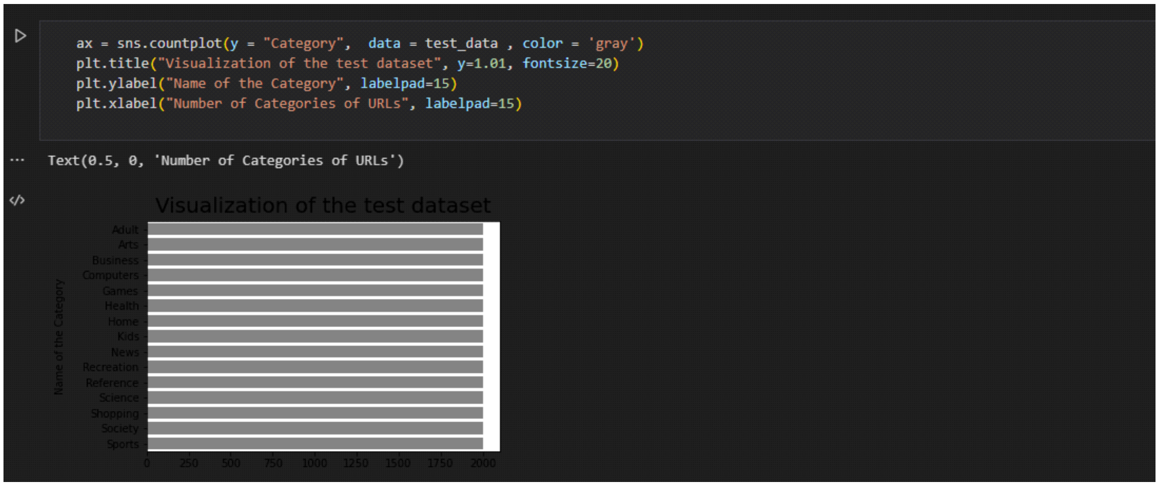




```python
ax = sns.countplot(y = "Category",  data = dataset )
plt.title("Visualization of the train dataset", y=1.01, fontsize=20)
plt.ylabel("Name of the Category", labelpad=15)
plt.xlabel("Number of Categories of URLs", labelpad=15)
```

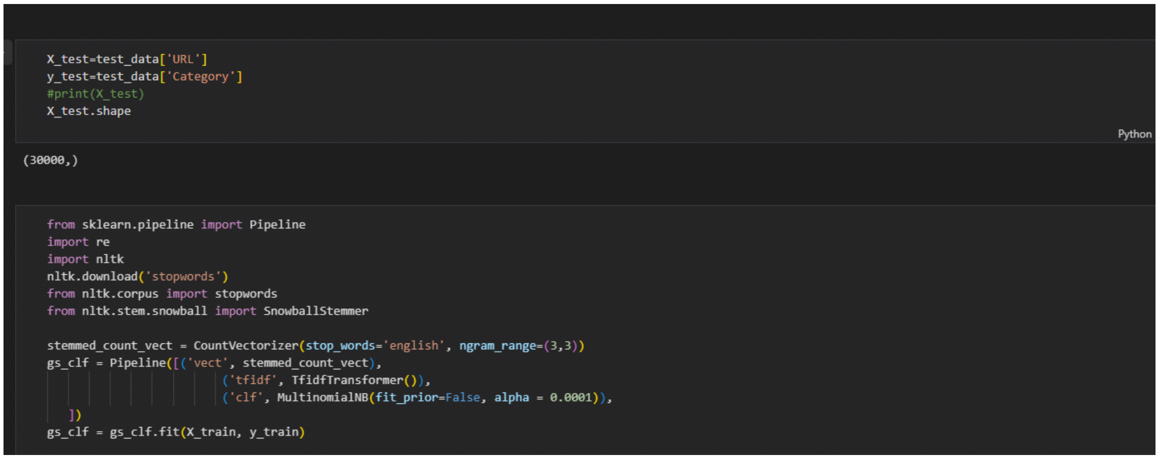
```
Text(0.5, 0, 'Number of Categories of URLs')
```

```python
ax = sns.countplot(y = "Category",  data = test_data , color = 'gray')
plt.title("Visualization of the test dataset", y=1.01, fontsize=20)
plt.ylabel("Name of the Category", labelpad=15)
plt.xlabel("Number of Categories of URLs", labelpad=15)
```



```
ax = sns.countplot(y = "Category",  data = test_data , color = 'gray')
plt.title("Visualization of the test dataset", y=1.01, fontsize=20)
plt.ylabel("Name of the Category", labelpad=15)
plt.xlabel("Number of Categories of URLs", labelpad=15)
```

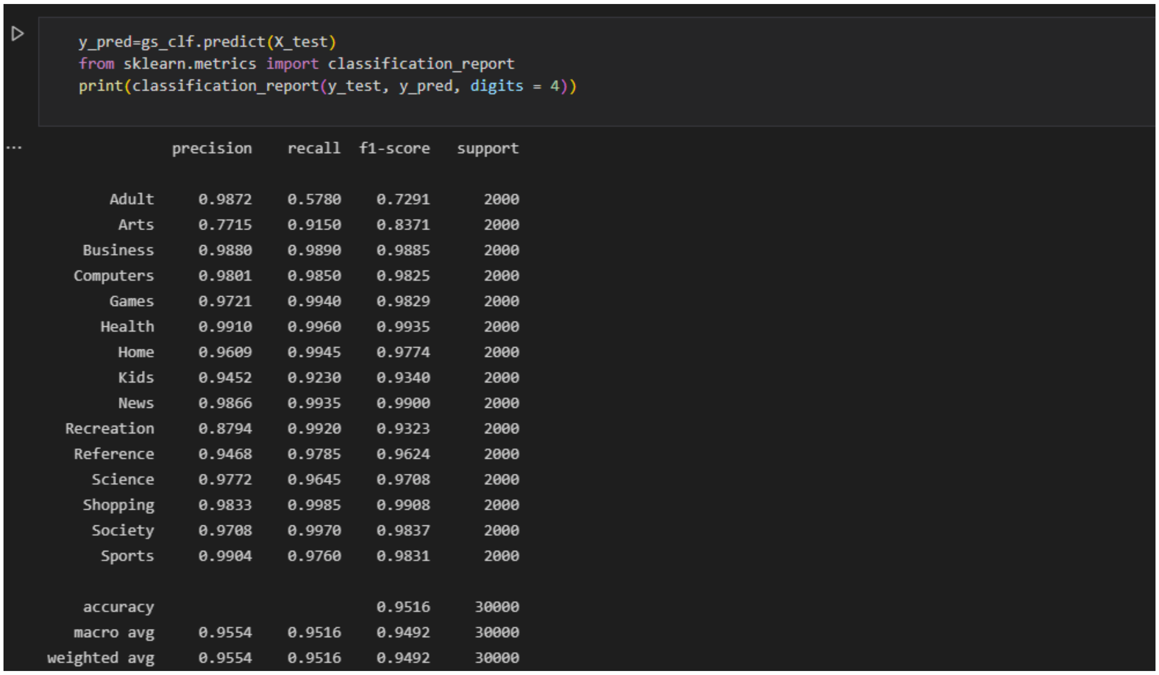
```
Text(0.5, 0, 'Number of Categories of URLs')
```



```
X_test=test_data['URL']
y_test=test_data['Category']
#print(X_test)
X_test.shape
```

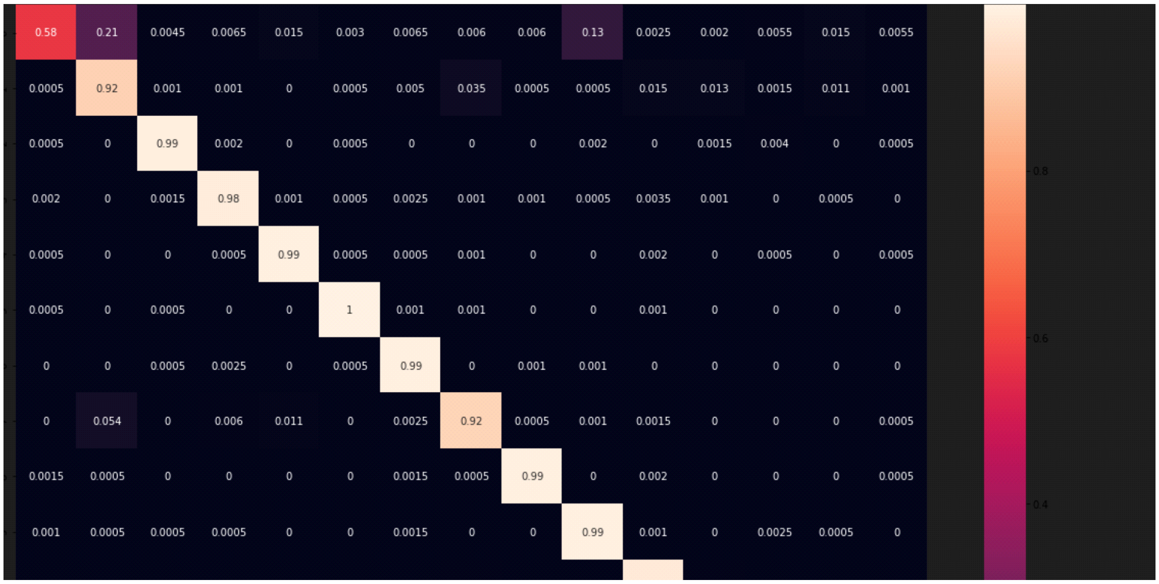
```
(30000,)
```

```
from sklearn.pipeline import Pipeline
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.snowball import SnowballStemmer

stemmed_count_vect = CountVectorizer(stop_words='english', ngram_range=(3,3))
gs_clf = Pipeline([('vect', stemmed_count_vect),
                        ('tfidf', TfidfTransformer()),
                        ('clf', MultinomialNB(fit_prior=False, alpha = 0.0001)),
  ])
gs_clf = gs_clf.fit(X_train, y_train)
```

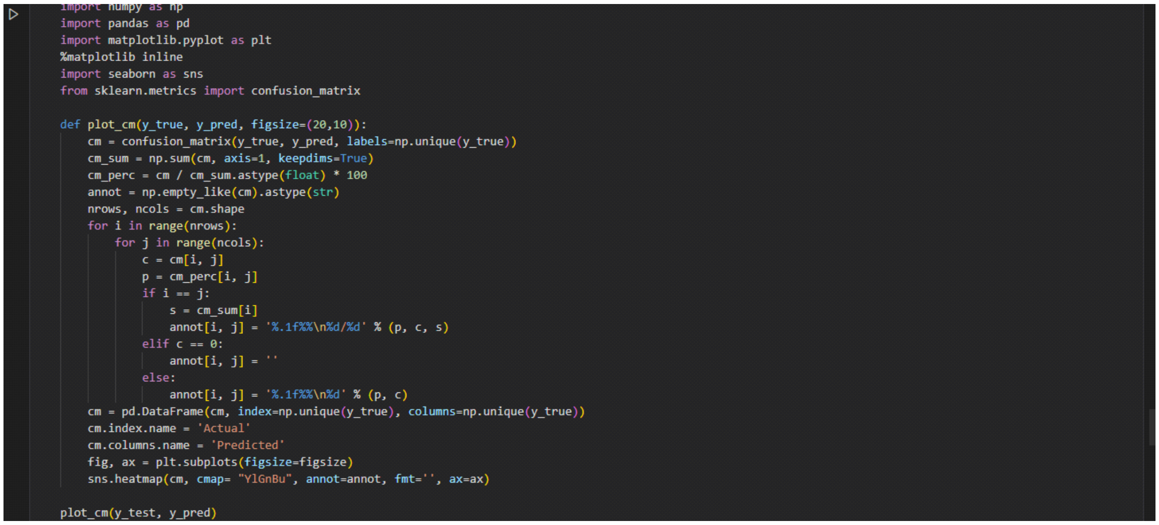


```
y_pred=gs_clf.predict(X_test)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred, digits = 4))
```

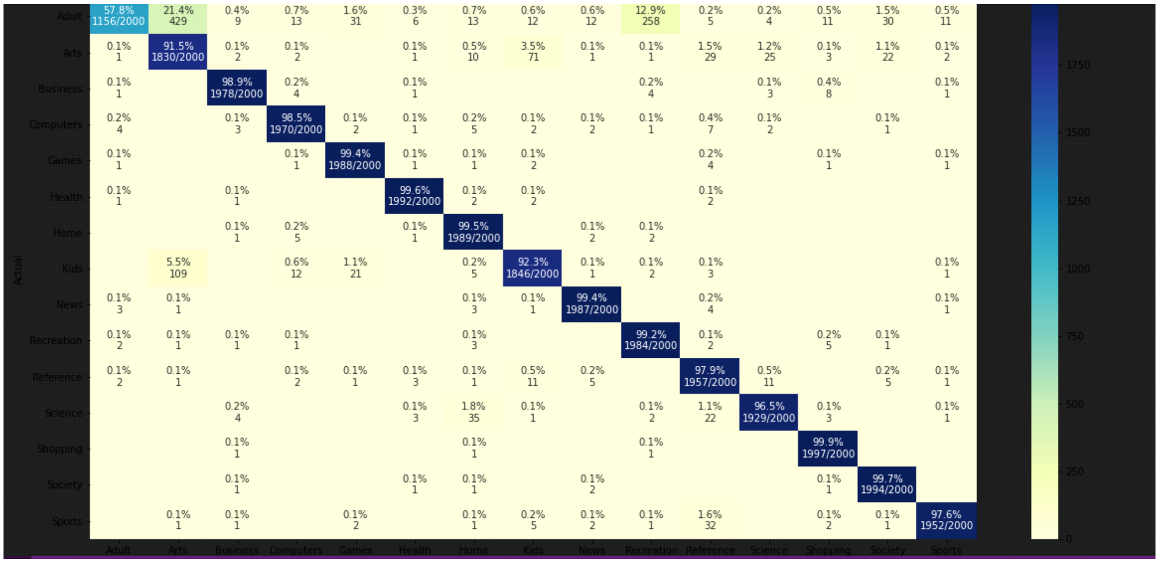
```
              precision    recall  f1-score   support

       Adult     0.9872    0.5780    0.7291      2000
        Arts     0.7715    0.9150    0.8371      2000
    Business     0.9880    0.9890    0.9885      2000
   Computers     0.9801    0.9850    0.9825      2000
       Games     0.9721    0.9940    0.9829      2000
      Health     0.9910    0.9960    0.9935      2000
        Home     0.9609    0.9945    0.9774      2000
        Kids     0.9452    0.9230    0.9340      2000
        News     0.9866    0.9935    0.9900      2000
  Recreation     0.8794    0.9920    0.9323      2000
   Reference     0.9468    0.9785    0.9624      2000
     Science     0.9772    0.9645    0.9708      2000
    Shopping     0.9833    0.9985    0.9908      2000
     Society     0.9708    0.9970    0.9837      2000
      Sports     0.9904    0.9760    0.9831      2000

    accuracy                         0.9516     30000
   macro avg     0.9554    0.9516    0.9492     30000
weighted avg     0.9554    0.9516    0.9492     30000
```



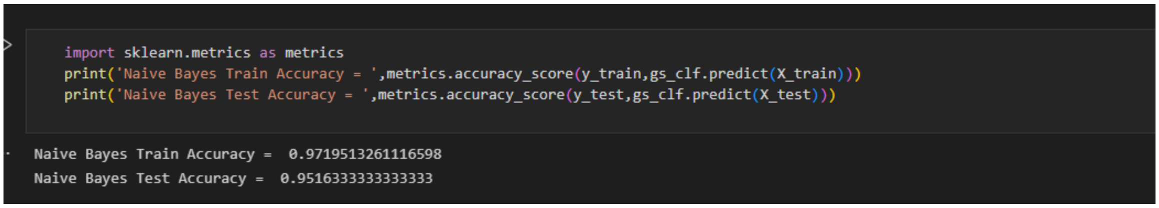
0.58	0.21	0.0045	0.0065	0.015	0.003	0.0065	0.006	0.006	0.13	0.0025	0.002	0.0055	0.015	0.0055
0.0005	0.92	0.001	0.001	0	0.0005	0.005	0.035	0.0005	0.0005	0.015	0.013	0.0015	0.011	0.001
0.0005	0	0.99	0.002	0	0.0005	0	0	0	0.002	0	0.0015	0.004	0	0.0005
0.002	0	0.0015	0.98	0.001	0.0005	0.0025	0.001	0.001	0.0005	0.0035	0.001	0	0.0005	0
0.0005	0	0	0.0005	0.99	0.0005	0.0005	0.001	0	0	0.002	0	0.0005	0	0.0005
0.0005	0	0.0005	0	0	1	0.001	0.001	0	0	0.001	0	0	0	0
0	0	0.0005	0.0025	0	0.0005	0.99	0	0.001	0.001	0	0	0	0	0
0	0.054	0	0.006	0.011	0	0.0025	0.92	0.0005	0.001	0.0015	0	0	0	0.0005
0.0015	0.0005	0	0	0	0	0.0015	0.0005	0.99	0	0.002	0	0	0	0.0005
0.001	0.0005	0.0005	0.0005	0	0	0.0015	0	0	0.99	0.001	0	0.0025	0.0005	0



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.metrics import confusion_matrix

def plot_cm(y_true, y_pred, figsize=(20,10)):
    cm = confusion_matrix(y_true, y_pred, labels=np.unique(y_true))
    cm_sum = np.sum(cm, axis=1, keepdims=True)
    cm_perc = cm / cm_sum.astype(float) * 100
    annot = np.empty_like(cm).astype(str)
    nrows, ncols = cm.shape
    for i in range(nrows):
        for j in range(ncols):
            c = cm[i, j]
            p = cm_perc[i, j]
            if i == j:
                s = cm_sum[i]
                annot[i, j] = '%.1f%%\n%d/%d' % (p, c, s)
            elif c == 0:
                annot[i, j] = ''
            else:
                annot[i, j] = '%.1f%%\n%d' % (p, c)
    cm = pd.DataFrame(cm, index=np.unique(y_true), columns=np.unique(y_true))
    cm.index.name = 'Actual'
    cm.columns.name = 'Predicted'
    fig, ax = plt.subplots(figsize=figsize)
    sns.heatmap(cm, cmap= "YlGnBu", annot=annot, fmt='', ax=ax)

plot_cm(y_test, y_pred)
```



Actual	Adult	Arts	Business	Computers	Games	Health	Home	Kids	News	Recreation	Reference	Science	Shopping	Society	Sports
Adult	57.8% 1156/2000	21.4% 429	0.4% 9	0.7% 13	1.6% 31	0.3% 6	0.7% 13	0.6% 12	0.6% 12	12.9% 258	0.2% 5	0.2% 4	0.5% 11	1.5% 30	0.5% 11
Arts	0.1% 1	91.5% 1830/2000	0.1% 2	0.1% 2		0.1% 1	0.5% 10	3.5% 71	0.1% 1	0.1% 1	1.5% 29	1.2% 25	0.1% 3	1.1% 22	0.1% 2
Business	0.1% 1		98.9% 1978/2000	0.2% 4		0.1% 1				0.2% 4		0.1% 3	0.4% 8		0.1% 1
Computers	0.2% 4		0.1% 3	98.5% 1970/2000	0.1% 2	0.1% 1	0.2% 5	0.1% 2	0.1% 2	0.1% 1	0.4% 7	0.1% 2		0.1% 1	
Games	0.1% 1			0.1% 1	99.4% 1988/2000	0.1% 1	0.1% 1	0.1% 2			0.2% 4		0.1% 1		0.1% 1
Health	0.1% 1		0.1% 1			99.6% 1992/2000	0.1% 2	0.1% 2			0.1% 2				
Home			0.1% 1	0.2% 5		0.1% 1	99.5% 1989/2000		0.1% 2	0.1% 2					
Kids		5.5% 109		0.6% 12	1.1% 21		0.2% 5	92.3% 1846/2000	0.1% 1	0.1% 2	0.1% 3				0.1% 1
News	0.1% 3	0.1% 1					0.1% 3	0.1% 1	99.4% 1987/2000		0.2% 4				0.1% 1
Recreation	0.1% 2	0.1% 1	0.1% 1	0.1% 1			0.1% 3			99.2% 1984/2000	0.1% 2		0.2% 5	0.1% 1	
Reference	0.1% 2	0.1% 1		0.1% 2	0.1% 1	0.1% 3	0.1% 1	0.5% 11	0.2% 5		97.9% 1957/2000	0.5% 11		0.2% 5	0.1% 1
Science			0.2% 4			0.1% 3	1.8% 35	0.1% 1		0.1% 2	1.1% 22	96.5% 1929/2000	0.1% 3		0.1% 1
Shopping			0.1% 1				0.1% 1			0.1% 1			99.9% 1997/2000		
Society			0.1% 1			0.1% 1	0.1% 1		0.1% 2				0.1% 1	99.7% 1994/2000	
Sports		0.1% 1	0.1% 1		0.1% 2		0.1% 1	0.2% 5	0.1% 2	0.1% 1	1.6% 32		0.1% 2	0.1% 1	97.6% 1952/2000



```
import sklearn.metrics as metrics
print('Naive Bayes Train Accuracy = ',metrics.accuracy_score(y_train,gs_clf.predict(X_train)))
print('Naive Bayes Test Accuracy = ',metrics.accuracy_score(y_test,gs_clf.predict(X_test)))
```

```
Naive Bayes Train Accuracy =  0.9719513261116598
Naive Bayes Test Accuracy =  0.9516333333333333
```

CONCLUSION AND FUTURE WORK(PHASE2):

PROJECT conclusion:

The project has successfully developed an AI-driven system for exploring and predicting company registration trends with the Registrar of Companies. The system has been trained on a large dataset of historical company registration data, and it is able to identify patterns and trends in the data that would be difficult or impossible for humans to identify on their own.

The system has been evaluated on a held-out test set, and it has been shown to be able to predict future company registration trends with high accuracy. The system is also able to generate visualizations and reports that communicate the findings of the AI analysis to users in a way that is easy to understand and use