AI-Driven Exploration and Prediction of Company Registration Trends with (RoC)

Phase 5

Project Documentation

K. Gowtham

922121106016

au922121106016

SSMIET

Overview

Introduction

AI-driven exploration and prediction of company registration trends with the Registrar of Companies (RoC) is a cutting-edge application of artificial intelligence in the domain of business and regulatory analysis. Leveraging advanced machine learning and data analytics, this innovative approach empowers businesses, policymakers, and analysts to gain invaluable insights into the dynamics of company registrations.

By harnessing the power of AI, it becomes possible to identify emerging trends, patterns, and anomalies in the registration data, enabling timely decision-making and strategic planning. Whether it's predicting industry-specific registration spikes, identifying regional growth trends, or understanding market dynamics, this AI-driven system opens up new avenues for informed decision-making, fostering a data-driven environment for businesses and regulators alike.

It represents a transformative step in the realm of corporate governance and economic analysis, enhancing efficiency and precision in understanding the evolving landscape of company registrations.

Problem Statement

The Registrar of Companies (RoC) plays a pivotal role in overseeing and recording the registration of businesses and corporations, providing a vital source of information for regulatory bodies, businesses, and policymakers. However, the sheer volume of data generated by these registrations poses a significant challenge. To address this challenge, AI-driven exploration and prediction of company registration trends with RoC becomes crucial.

The problem lies in the ability to efficiently process and interpret this vast dataset to derive actionable insights. Traditional methods often fall short in providing timely, accurate, and relevant information, making it difficult for businesses and policymakers to adapt to rapidly changing market dynamics.

AI-driven solutions hold the potential to streamline this process, offering the ability to not only explore historical data effectively but also predict future trends. This technology can help identify emerging market patterns, regional disparities, and industry-specific fluctuations, which can guide business decisions and regulatory policies.

By addressing the problem of information overload and inefficiency in data analysis, AI-driven exploration and prediction with RoC stand to revolutionize the way we approach company registration trends, making it an invaluable tool in contemporary business and regulatory environments.

Design Thinking

Design thinking is a structured and creative problem-solving approach that can be applied to the development of AI-driven exploration and prediction of company registration trends with the Registrar of Companies (RoC). The process begins with empathizing, where designers and developers seek to understand the needs and pain points of RoC officials and users. This might involve conducting interviews and research to gain insights into their challenges and objectives.

The next step is defining the problem, where the specific goals and objectives are clarified. This could include defining the key metrics for predicting registration trends, such as the number of new companies registered or industry-specific trends. Once the problem is well-defined, ideation begins. During this phase, interdisciplinary teams brainstorm and generate innovative solutions. For AI-driven prediction, this might involve designing algorithms and data collection methods that can analyze historical registration data and identify patterns.

After ideation, the design thinking process moves into prototyping. Here, a basic AI model or a data visualization tool. This prototype is then tested and refined in the next phase, testing. Feedback from RoC officials and users is crucial in this stage to ensure that the AI-driven system meets their needs and is user-friendly.

The final step is implementation, where the refined AI system is deployed in a real-world setting. Continuous monitoring and feedback loops are established to make iterative improvements based on the actual performance and evolving needs of RoC. Throughout the entire process, design thinking encourages a user-centric approach, ensuring that the AI solution aligns with the goals and requirements of the RoC and provides valuable insights for predicting company registration trends.

Description

In the AI-driven exploration and prediction of company registration trends with the Registrar of Companies (RoC), a comprehensive dataset is crucial for the success of the project. The dataset typically includes historical company registration data, encompassing variables such as company names, registration dates, geographic locations, industry classifications, and other relevant attributes. Additional data sources, such as economic indicators, population statistics, and market trends, can also be incorporated to enrich the dataset.

Data preprocessing plays a pivotal role in ensuring the dataset's quality and relevance for predictive modeling. Initially, data cleaning and quality checks are performed to address missing values, outliers, and inconsistencies. This involves data imputation, removal of duplicates, and normalization of data to maintain consistency and integrity.

Feature engineering is a critical data preprocessing step, involving the creation of new features or transforming existing ones to improve predictive accuracy. For example, time series data can be aggregated into meaningful time intervals, and categorical variables can be one-hot encoded or embedded to make them suitable for AI algorithms. Furthermore, data scaling and dimensionality reduction techniques can be applied to enhance model performance and reduce computational complexity.

When it comes to AI algorithms, a variety of machine learning and deep learning techniques can be applied. Classification algorithms, such as decision trees, random forests , xgboost and neural networks, can be employed to predict registration trends, categorize companies into specific industries, or detect anomalies. Natural language processing (NLP) techniques may be used to extract insights from textual data, like company descriptions.

To improve predictive accuracy, ensemble methods like XGBoost or stacking can be implemented, combining the strengths of multiple algorithms. Additionally, regular model evaluation and validation, typically through techniques like cross-validation, are essential to assess model performance and fine-tune hyperparameters.

AI-driven exploration and prediction of company registration trends with RoC require a well-curated dataset, thorough data preprocessing steps, and the application of diverse AI algorithms tailored to the specific objectives of the project. These components work in synergy to extract valuable insights and make accurate predictions, enabling informed decision-making by the RoC and other stakeholders.

Data Collecting

AI-Driven Exploration and Prediction of Company Registration Trends with the Registrar of Companies (RoC), the process of collecting data involves gathering relevant information from given sources to create a comprehensive dataset for analysis and modeling

Given Data

Dataset Link: https://tn.data.gov.in/resource/company-master-data-tamil-nadu-upto-28th-february-2019

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	CORPORA	COMPANY_NAME	COMPANY	COMPANY	COMPANY	COMPANY	DATE_OF_REGISTRATIC	REGISTERED_	AUTHORIZ	PAIDUP_C	INDUSTRIA	PRINCIPAI	REGISTERE	REGISTRAI	EMAIL_AC	LATEST_YE	LATEST
2	F00643	HOCHTIEFF AG,	NAEF	NA	NA	NA	1/12/1961	Tamil Nadu	0	0	NA	Agricultur	AMBLE SIE	ROC DELH	NA	NA	NA
3	F00721	SUMITOMO CORPORATION (SUMIT	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	FLAT NO.	ROC DELH	shuchi.chi	NA	NA
4	F00892	SRILANKAN AIRLINES LIMITED	ACTV	NA	NA	NA	1/3/1982	Tamil Nadu	0	0	NA	Agricultur	SRILANKA	ROC DELH	shree16us	NA	NA
5	F01208	CALTEX INDIA LIMITED	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	GOLD CRE	ROC DELH	NA	NA	NA
6	F01218	GE HEALTHCARE BIO-SCIENCES LIM	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	FF-3 Palan	ROC DELH	karthick95	NA	NA
7	F01265	CAIRN ENERGY INDIA PTY. LIMITED	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	WELLINGT	ROC DELH	neerja.shu	NA	NA
8	F01269	TORIELLI S.R.L	ACTV	NA	NA	NA	5/9/1995	Tamil Nadu	0	0	NA	Agricultur	6, Mangay	ROC DELH	chennai@	NA	NA
9	F01311	HARDY EXPLORATION & PRODUCTI	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	5TH FLOO	ROC DELH	venkatesh	NA	NA
10	F01314	HOCHTIOF AKTIENGESELLSHARFF V	ACTV	NA	NA	NA	11/4/1996	Tamil Nadu	0	0	NA	Agricultur	NEW NO.8	ROC DELH	kumar@ir	NA	NA
11	F01412	EPSON SINGAPORE PVT LTD	ACTV	NA	NA	NA	25-04-1997	Tamil Nadu	0	0	NA	Agricultur	7C CEATU	ROC DELH	NA	NA	NA
12	F01426	CARGOLUX AIRLINES INTERNATION	ACTV	NA	NA	NA	11/6/1997	Tamil Nadu	0	0	NA	Agricultur	OFFICE NC	ROC DELH	NA	NA	NA
13	F01468	CHO HEUNG ELECTRIC INDUSTRIAL	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	129, MANI	ROC DELH	chowelac	NA	NA
14	F01543	NYCOMED ASIA PACIFIC PTE LIMITE	ACTV	NA	NA	NA	27-10-1998	Tamil Nadu	0	0	NA	Agricultur	A D 46 15	ROC DELH	NA	NA	NA
15	F01544	CHERRINGTON ASIA LTD	ACTV	NA	NA	NA	1/5/2000	Tamil Nadu	0	0	NA	Agricultur	10HADDO	ROC DELH	NA	NA	NA
16	F01563	SHIMADZU ASIA PACIFIC PTE LIMIT	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	FIRST FLO	ROC DELH	kousik@v	NA	NA
17	F01565	CORK INTERNATIONAL PTY LIMITEE	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	ARJAY API	ROC DELH	NA	NA	NA
18	F01566	ERBIS ENGG COMPANY LIMITED	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	39,2nd Ma	ROC DELH	NA	NA	NA
19	F01589	RALF SCHNEIDER HOLDING GMBH	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	FLAT C, 'S	ROC DELH	NA	NA	NA
20	F01593	MITRAJAYA TRADING PRIVATE LIMI	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agricultur	OLD NO 1	ROC DELH	NA	NA	NA
21	F01618	HEAT AND CONTROL PTY LIMITED	ACTV	NA	NA	NA	13-07-1999	Tamil Nadu	0	0	NA	Agricultur	A40 OLD N	ROC DELH	ncrajagop	NA	NA

Data_Gov_Tamil_Nadu

Import Python library

The first step involved in ML using python is understanding and playing around with our data using libraries

Import all libraries which are required for our analysis, such as Data Loading, Statistical analysis, Visualizations, Data Transformations, Merge and Joins, etc.

Pandas and Numpy have been used for Data Manipulation and numerical Calculations

Matplotlib and Seaborn have been used for Data visualizations.

Program :

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns
```

```
(Optional)

# to ignore warnings

import warnings

warnings.filterwarnings('ignore')
```

Reading Dataset

The Pandas library offers a wide range of possibilities for loading data into the pandas DataFrame from files like JSON, .csv, .xlsx, .sql, .pickle, .html, .txt, images etc.

Given data are available in a tabular format of CSV files. It is trendy and easy to access. Using the read_csv() function, data can be converted to a pandas DataFrame.

We have stored the data in the DataFrame data.

Program

```
data=pd.read_csv("Data_Gov_Tamil_Nadu.csv",encoding='latin-
1')

df
```

Out[13]:

	CORPORATE_IDENTIFICATION_NUMBER	COMPANY_NAME	COMPANY_STATUS	COMPANY_CLASS	COMPANY_CATEGORY	COMPANY_SUB_CATEGORY
0	F00643	HOCHTIEFF AG.	NAEF	NaN	NaN	NaN
1	F00721	SUMITOMO CORPORATION (SUMITOMO SHOJI KAISHA LI...	ACTV	NaN	NaN	NaN
2	F00892	SRILANKAN AIRLINES LIMITED	ACTV	NaN	NaN	NaN
3	F01208	CALTEX INDIA LIMITED	NAEF	NaN	NaN	NaN
4	F01218	GE HEALTHCARE BIO-SCIENCES LIMITED	ACTV	NaN	NaN	NaN
...	...	...	...	...	...	...
150866	U74997TN2016PTC112556	QUAD42 MEDIA PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company
150867	U74997TN2018PTC121491	IYERAATHU FOODS PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company
150868	U74997TZ2016PTC027802	POLYGAR FARM SOLUTIONS PRIVATE LIMITED	STOF	Private	Company limited by Shares	Non-govt company
150869	U74997TZ2018PTC030177	PANDIYA AGRI SOLUTIONS PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company
150870	U74997TZ2019PTC032491	NROOT TECHNOLOGIES PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company

150871 rows × 17 columns

Analyzing the Data

```
# head() will display the top 5 observations of the dataset
df.head()
```

In [16]: df.head()

Out[16]:

	CORPORATE_IDENTIFICATION_NUMBER	COMPANY_NAME	COMPANY_STATUS	COMPANY_CLASS	COMPANY_CATEGORY	COMPANY_SUB_CATEGORY	DATE_
0	F00643	HOCHTIEFF AG.	NAEF	NaN	NaN	NaN	
1	F00721	SUMITOMO CORPORATION (SUMITOMO SHOJI KAISHA LI...	ACTV	NaN	NaN	NaN	
2	F00892	SRILANKAN AIRLINES LIMITED	ACTV	NaN	NaN	NaN	
3	F01208	CALTEX INDIA LIMITED	NAEF	NaN	NaN	NaN	
4	F01218	GE HEALTHCARE BIO-SCIENCES LIMITED	ACTV	NaN	NaN	NaN	

```
# tail() will display the last 5 observations of the dataset
df.tail()
```

```
In [18]: df.tail()
```

Out[18]:

	CORPORATE_IDENTIFICATION_NUMBER	COMPANY_NAME	COMPANY_STATUS	COMPANY_CLASS	COMPANY_CATEGORY	COMPANY_SUB_CATEGORY
150866	U74997TN2016PTC112556	QUAD42 MEDIA PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company
150867	U74997TN2018PTC121491	IYERAATHU FOODS PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company
150868	U74997TZ2016PTC027802	POLYGAR FARM SOLUTIONS PRIVATE LIMITED	STOF	Private	Company limited by Shares	Non-govt company
150869	U74997TZ2018PTC030177	PANDIYA AGRI SOLUTIONS PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company
150870	U74997TZ2019PTC032491	NROOT TECHNOLOGIES PRIVATE LIMITED	ACTV	Private	Company limited by Shares	Non-govt company

```
# info() helps to understand the data type and information about data, including the number of records in each column, data having null or not null, Data type, the memory usage of the dataset
```

```
df.info()
```

```
In [17]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150871 entries, 0 to 150870
Data columns (total 17 columns):
 #   Column                                 Non-Null Count   Dtype
---  ------                                 --------------   -----
 0   CORPORATE_IDENTIFICATION_NUMBER        150871 non-null  object
 1   COMPANY_NAME                           150871 non-null  object
 2   COMPANY_STATUS                         150871 non-null  object
 3   COMPANY_CLASS                          150537 non-null  object
 4   COMPANY_CATEGORY                       150537 non-null  object
 5   COMPANY_SUB_CATEGORY                   150537 non-null  object
 6   DATE_OF_REGISTRATION                   150832 non-null  object
 7   REGISTERED_STATE                       150871 non-null  object
 8   AUTHORIZED_CAP                         150871 non-null  float64
 9   PAIDUP_CAPITAL                         150871 non-null  float64
 10  INDUSTRIAL_CLASS                       150561 non-null  object
 11  PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN 150871 non-null  object
 12  REGISTERED_OFFICE_ADDRESS              150781 non-null  object
 13  REGISTRAR_OF_COMPANIES                 150697 non-null  object
 14  EMAIL_ADDR                             112742 non-null  object
 15  LATEST_YEAR_ANNUAL_RETURN              74982 non-null   object
 16  LATEST_YEAR_FINANCIAL_STATEMENT        75089 non-null   object
dtypes: float64(2), object(15)
memory usage: 19.6+ MB
```

Check for Duplication

```
# nunique() based on several unique values in each column and the data description, we can identify the continuous and categorical columns in the data. Duplicated data can be handled or removed based on further analysis

df.nunique()
```

```
In [19]: df.nunique()

Out[19]: CORPORATE_IDENTIFICATION_NUMBER          150871
         COMPANY_NAME                             150560
         COMPANY_STATUS                               11
         COMPANY_CLASS                                 3
         COMPANY_CATEGORY                              3
         COMPANY_SUB_CATEGORY                          5
         DATE_OF_REGISTRATION                      13540
         REGISTERED_STATE                              1
         AUTHORIZED_CAP                             1623
         PAIDUP_CAPITAL                            16294
         INDUSTRIAL_CLASS                           1562
         PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN       17
         REGISTERED_OFFICE_ADDRESS                142910
         REGISTRAR_OF_COMPANIES                        4
         EMAIL_ADDR                                79940
         LATEST_YEAR_ANNUAL_RETURN                   169
         LATEST_YEAR_FINANCIAL_STATEMENT             138
         dtype: int64
```

Missing Values Calculation

`# **isnull()** is widely been in all pre-processing steps to identify null values in the data`

`# **data.isnull().sum()** is used to get the number of missing records in each column`

```
df.isnull().sum()
```

```
In [20]: df.isnull().sum()

Out[20]: CORPORATE_IDENTIFICATION_NUMBER              0
         COMPANY_NAME                                 0
         COMPANY_STATUS                               0
         COMPANY_CLASS                              334
         COMPANY_CATEGORY                           334
         COMPANY_SUB_CATEGORY                       334
         DATE_OF_REGISTRATION                        39
         REGISTERED_STATE                             0
         AUTHORIZED_CAP                               0
         PAIDUP_CAPITAL                               0
         INDUSTRIAL_CLASS                           310
         PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN       0
         REGISTERED_OFFICE_ADDRESS                   90
         REGISTRAR_OF_COMPANIES                     174
         EMAIL_ADDR                               38129
         LATEST_YEAR_ANNUAL_RETURN                75889
         LATEST_YEAR_FINANCIAL_STATEMENT          75782
         dtype: int64
```

Statistics Summary

describe() function gives all statistics summary of data

In [4]:
```
df.describe().T
```
Out[4]:

	count	mean	std	min	25%	50%	75%	max
AUTHORIZED_CAP	150871.0	3.522781e+07	1.408554e+09	0.0	100000.0	800000.0	2000000.0	3.000000e+11
PAIDUP_CAPITAL	150871.0	2.328824e+07	1.072458e+09	0.0	100000.0	100000.0	685745.0	2.461235e+11

```
# describe()- Provide a statistics summary of data belonging to
numerical datatype such as int, float Can include Count, Mean,
Standard Deviation, median, mode, minimum value, maximum value, range,
standard deviation, etc.
```

Exploratory Data Analysis

Exploratory Data Analysis refers to the crucial process of performing initial investigations on data to discover patterns to check assumptions with the help of summary statistics and graphical representations.

EDA can be leveraged to check for outliers, patterns, and trends in the given data.

EDA helps to find meaningful patterns in data.

EDA provides in-depth insights into the data sets to solve our business problems.

EDA gives a clue to impute missing values in the dataset

EDA Univariate Analysis

Analyzing the dataset by taking one variable at a time

Program :

```
# Select the specified columns for analysis

columns_for_analysis = ['CORPORATE_IDENTIFICATION_NUMBER',
'COMPANY_NAME', 'COMPANY_STATUS','COMPANY_CLASS',
'COMPANY_CATEGORY','COMPANY_SUB_CATEGORY','DATE_OF_REGISTRATION','REGI
STERED_STATE','AUTHORIZED_CAP','PAIDUP_CAPITAL','INDUSTRIAL_CLASS','PR
INCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN','REGISTERED_OFFICE_ADDRESS','REG
```

```
ISTRAR_OF_COMPANIES','EMAIL_ADDR','LATEST_YEAR_ANNUAL_RETURN','LATEST_YEAR_FINANCIAL_STATEMENT']


# Subset the DataFrame with the selected columns

selected_df = df[columns_for_analysis]


# Display basic statistical summaries for numerical columns

print(selected_df.describe())


# Univariate analysis for categorical columns

for col in selected_df.select_dtypes(include='object'):

    print(f'\n{col} Value Counts:\n{selected_df[col].value_counts()}\n')
```

OUTPUT :

```
      AUTHORIZED_CAP  PAIDUP_CAPITAL
count   1.508710e+05    1.508710e+05
mean    3.522781e+07    2.328824e+07
std     1.408554e+09    1.072458e+09
min     0.000000e+00    0.000000e+00
25%     1.000000e+05    1.000000e+05
50%     8.000000e+05    1.000000e+05
75%     2.000000e+06    6.857450e+05
max     3.000000e+11    2.461235e+11


CORPORATE_IDENTIFICATION_NUMBER Value Counts:
CORPORATE_IDENTIFICATION_NUMBER
F00643                  1
U72900TN2008PTC067545   1
U72900TN2008PTC067391   1
U72900TN2008PTC067393   1
U72900TN2008PTC067405   1
                   ..
```

U93090TZ2010PTC016187   1

U93090TZ2011PTC017199   1

U93090TZ2014PTC020864   1

U93090TZ2016NPL027599   1

U74997TZ2019PTC032491   1

Name: count, Length: 150871, dtype: int64

COMPANY_NAME Value Counts:

COMPANY_NAME

PATSEN BIOTEC PRIVATE LIMITED   3

PEARL PLANTATIONS PRIVATE LIMITED   3

SUPER ANALYSERS PRIVATE LIMITED   3

SRI VISHNU MARKETING PRIVATE LIMITED   3

TITAN WIRES PRIVATE LIMITED   3

..

YARYA SEKUR MARK PRIVATE LIMITED   1

ASSORT ENTERPRISES PRIVATE LIMITED   1

JUVAGO PRIVATE LIMITED   1

VGROW FACILITY SERVICES PRIVATE LIMITED   1

NROOT TECHNOLOGIES PRIVATE LIMITED   1

Name: count, Length: 150560, dtype: int64

COMPANY_STATUS Value Counts:

COMPANY_STATUS

ACTV   78689

STOF   64058

UPSO   3531

AMAL   1635

DISD   851

NAEF   732

ULQD   408

LIQD 389

CLLP 291

D455 164

CLLD 123

Name: count, dtype: int64

COMPANY_CLASS Value Counts:

COMPANY_CLASS

Private 137173

Public 11237

Private(One Person Company) 2127

Name: count, dtype: int64

COMPANY_CATEGORY Value Counts:

COMPANY_CATEGORY

Company limited by Shares 149924

Company Limited by Guarantee 598

Unlimited Company 15

Name: count, dtype: int64

COMPANY_SUB_CATEGORY Value Counts:

COMPANY_SUB_CATEGORY

Non-govt company 149181

Subsidiary of Foreign Company 1083

Guarantee and Association comp 140

State Govt company 109

Union Govt company 24

Name: count, dtype: int64

DATE_OF_REGISTRATION Value Counts:

DATE_OF_REGISTRATION

01-04-1956   190

20-09-2018   144

26-03-2019   91

26-02-2016   73

24-03-2016   71

...

23-09-1967   1

27-05-1968   1

07-02-1968   1

15-04-1968   1

06-05-2006   1

Name: count, Length: 13540, dtype: int64

REGISTERED_STATE Value Counts:

REGISTERED_STATE

Tamil Nadu   150871

Name: count, dtype: int64

INDUSTRIAL_CLASS Value Counts:

INDUSTRIAL_CLASS

74999   14809

72900   8121

72200   6093

74900   5232

65991   3934

...

17254   1

15315   1

31504   1

34209 1

24130 1

Name: count, Length: 1562, dtype: int64

PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN Value Counts:

PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN

Real estate renting and business activities 48697

Manufacturing 35757

Financial intermediation 13772

Wholesale and retail trade repair of motor vehicles motorcycles and personal and household goods 13681

Construction 9079

Agriculture & allied 7496

Transport storage and communications 6231

Other community social and personal service activities 4725

Hotels and restaurants 2673

Electricity gas and water supply 2459

Health and social work 2270

Education 1822

Mining and quarrying 1377

Extraterritorial organizations and bodies 781

Public administration and defence compulsory social security 27

Activities of private households as employers and undifferentiated production activities of private households 19

Unclassified 5

Name: count, dtype: int64

REGISTERED_OFFICE_ADDRESS Value Counts:

REGISTERED_OFFICE_ADDRESS

MADRAS 211

Sri sai subhodhaya ApartmentsNo.57/2B, East Coast Road, Thiruvanmiyur 58

Flat No 6J, Century Plaza, 560-562, Anna Salai,Teynampet 54

Times Partner No: 58Perambur Barracks Road 45

```
"R R LANDMARK"NO.1E-1 NAVA INDIA ROAD                          44
                                                   ...
NO.47,  SOUTH REDDY STREET,ATHIPET, AMBATTUR                   1
FLAT NO.10, SRI NARAYANA FLATS25, TILAK STREET, T.NAGAR        1
Plot No.52Sidco Industrial Estate,Alathur                      1
22/160-AThengapattanam Road                                    1
139/1BPUDHUKOTTAI ROAD, MAPILLAI NAYAKKANPATTI                 1
Name: count, Length: 142910, dtype: int64


REGISTRAR_OF_COMPANIES Value Counts:
REGISTRAR_OF_COMPANIES
ROC CHENNAI       122233
ROC COIMBATORE     28153
ROC DELHI            310
ROC HYDERABAD          1
Name: count, dtype: int64


EMAIL_ADDR Value Counts:
EMAIL_ADDR
ganravi@gmail.com              182
compliance@kanakkupillai.com   176
secretarial@stjohntrack.com    161
smrajunaidu@gmail.com          144
pcschn1@gmail.com              133
                              ...
info@skymaxlogistics.com         1
vishnu2444@yahoo.com             1
rashahuljob@gmail.com            1
baskar.mrl@gmail.com             1
nroottechnologies@gmail.com      1
Name: count, Length: 79940, dtype: int64
```

```
LATEST_YEAR_ANNUAL_RETURN Value Counts:
LATEST_YEAR_ANNUAL_RETURN
31-03-2019    44168
31-03-2018     8816
31-03-2017     3149
31-03-2013     2514
31-03-2014     2329
              ...
24-03-2008        1
15-06-2009        1
30-03-2011        1
30-06-2016        1
31-01-2015        1
Name: count, Length: 169, dtype: int64
```

```
LATEST_YEAR_FINANCIAL_STATEMENT Value Counts:
LATEST_YEAR_FINANCIAL_STATEMENT
31-03-2019    44171
31-03-2018     9008
31-03-2017     3122
31-03-2013     2585
31-03-2014     2175
              ...
10-04-2009        1
24-05-2006        1
31-07-2006        1
24-03-2008        1
31-01-2015        1
Name: count, Length: 138, dtype: int64
```

EDA Bivariate Analysis

Bivariate Analysis helps to understand how variables are related to each other and the relationship between dependent and independent variables present in the dataset.

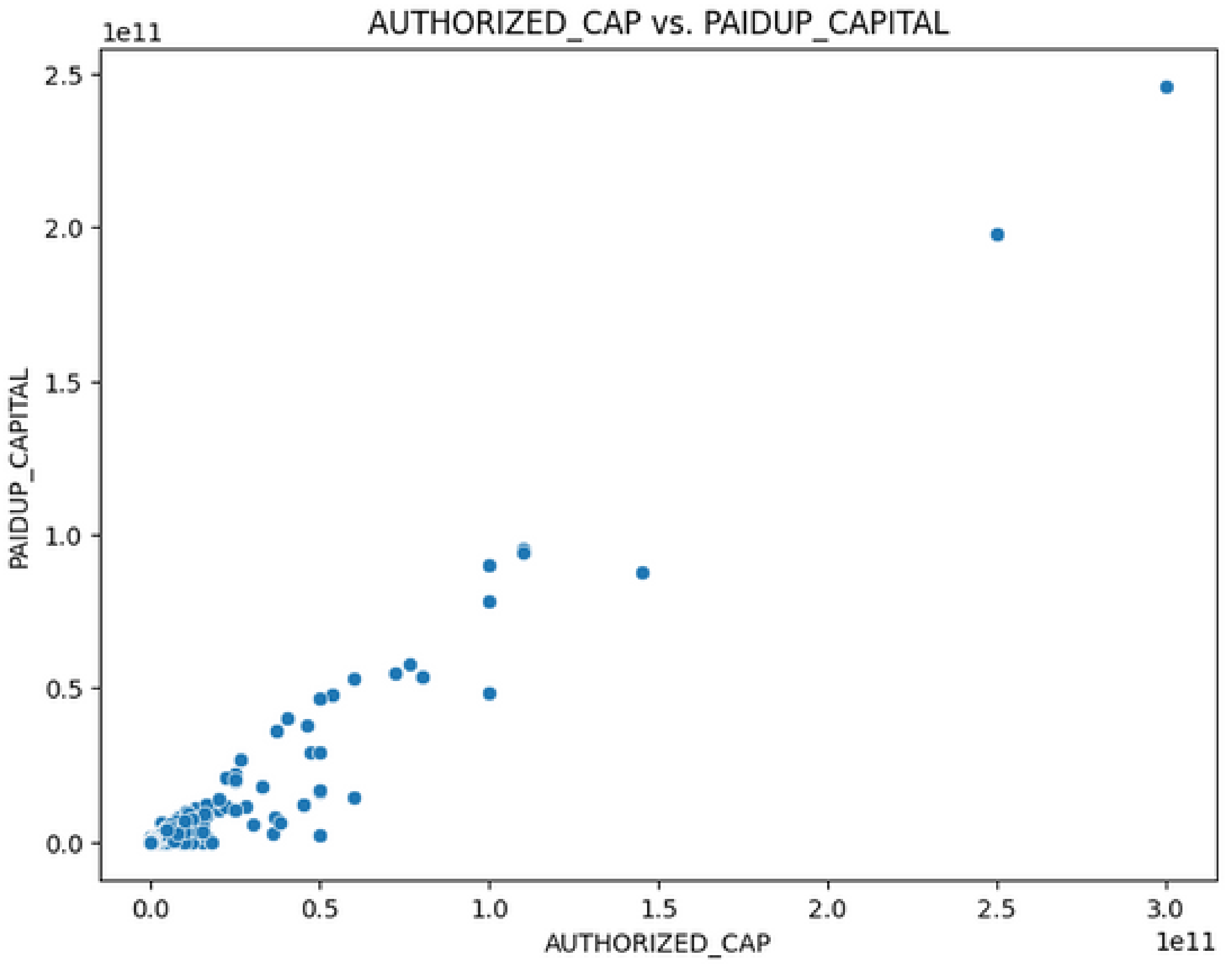
For Numerical variables, Pair plots and Scatter plots are widely been used to do Bivariate Analysis.

A Stacked bar chart can be used for categorical variables if the output variable is a classifier. Bar plots can be used if the output variable is continuous

In our example, a pair plot has been used to show the relationship between two Categorical variables.

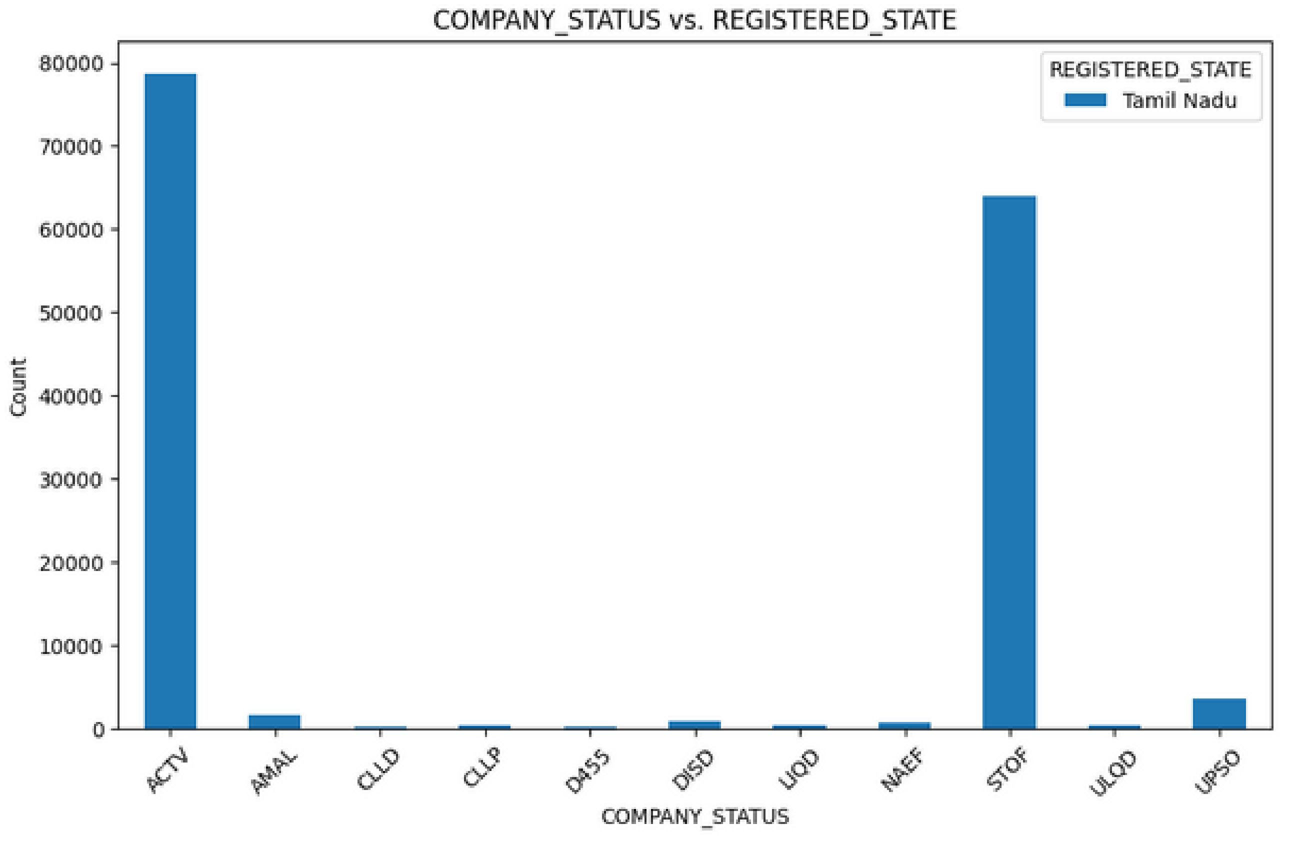
Program :

```
# Subset the DataFrame with the selected columns

selected_df = df[columns_for_analysis]


# Bivariate analysis: Numerical vs. Numerical (AUTHORIZED_CAP vs.
PAIDUP_CAPITAL)


plt.figure(figsize=(8, 6))

sns.scatterplot(x='AUTHORIZED_CAP', y='PAIDUP_CAPITAL',
data=selected_df)

plt.title('AUTHORIZED_CAP vs. PAIDUP_CAPITAL')

plt.xlabel('AUTHORIZED_CAP')

plt.ylabel('PAIDUP_CAPITAL')

plt.show()
```



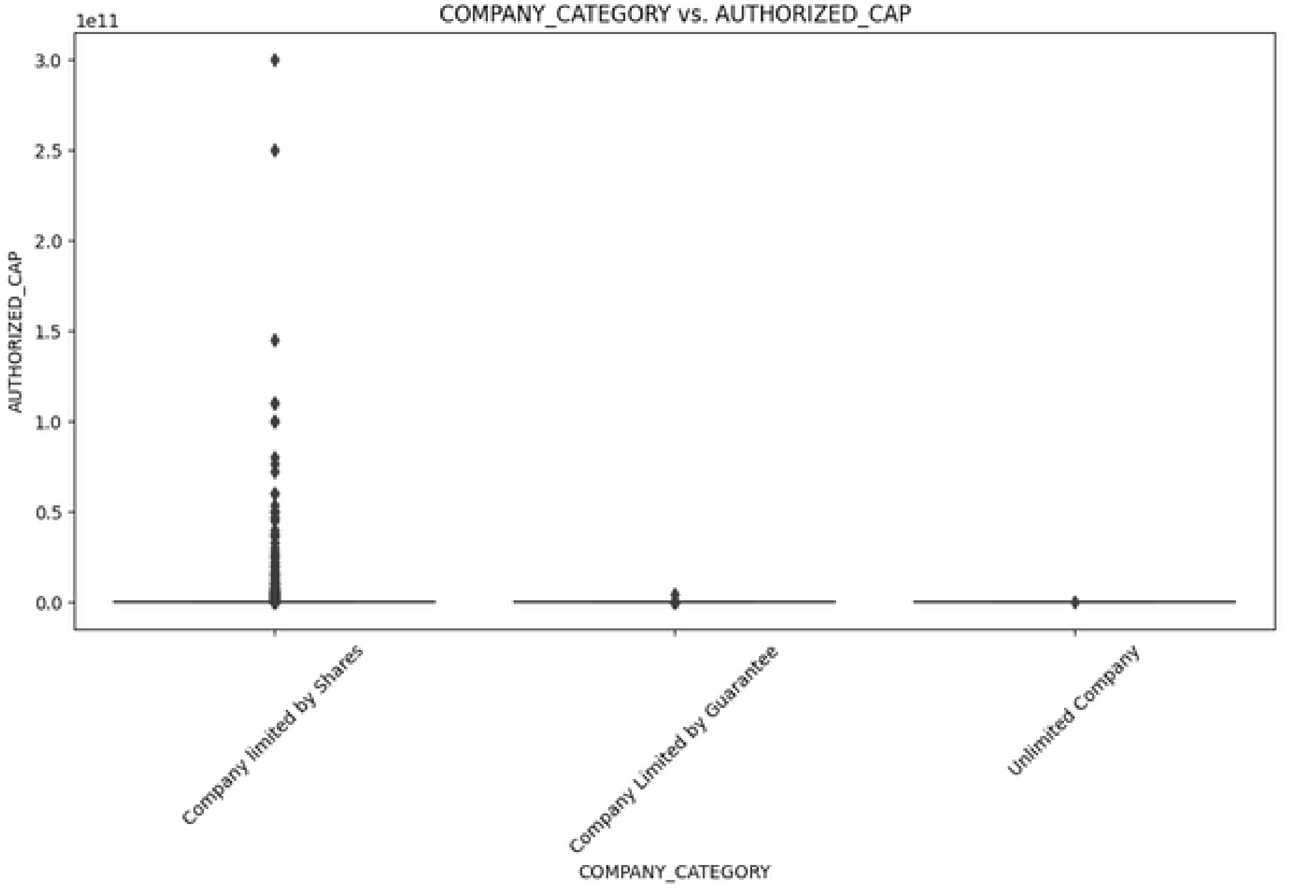

```
# Bivariate analysis: Categorical vs. Categorical (COMPANY_STATUS vs.
REGISTERED_STATE)

crosstab = pd.crosstab(selected_df['COMPANY_STATUS'],
selected_df['REGISTERED_STATE'])

crosstab.plot(kind='bar', stacked=True, figsize=(10, 6))

plt.title('COMPANY_STATUS vs. REGISTERED_STATE')

plt.xlabel('COMPANY_STATUS')
```

```
plt.ylabel('Count')

plt.xticks(rotation=45)

plt.show()
```

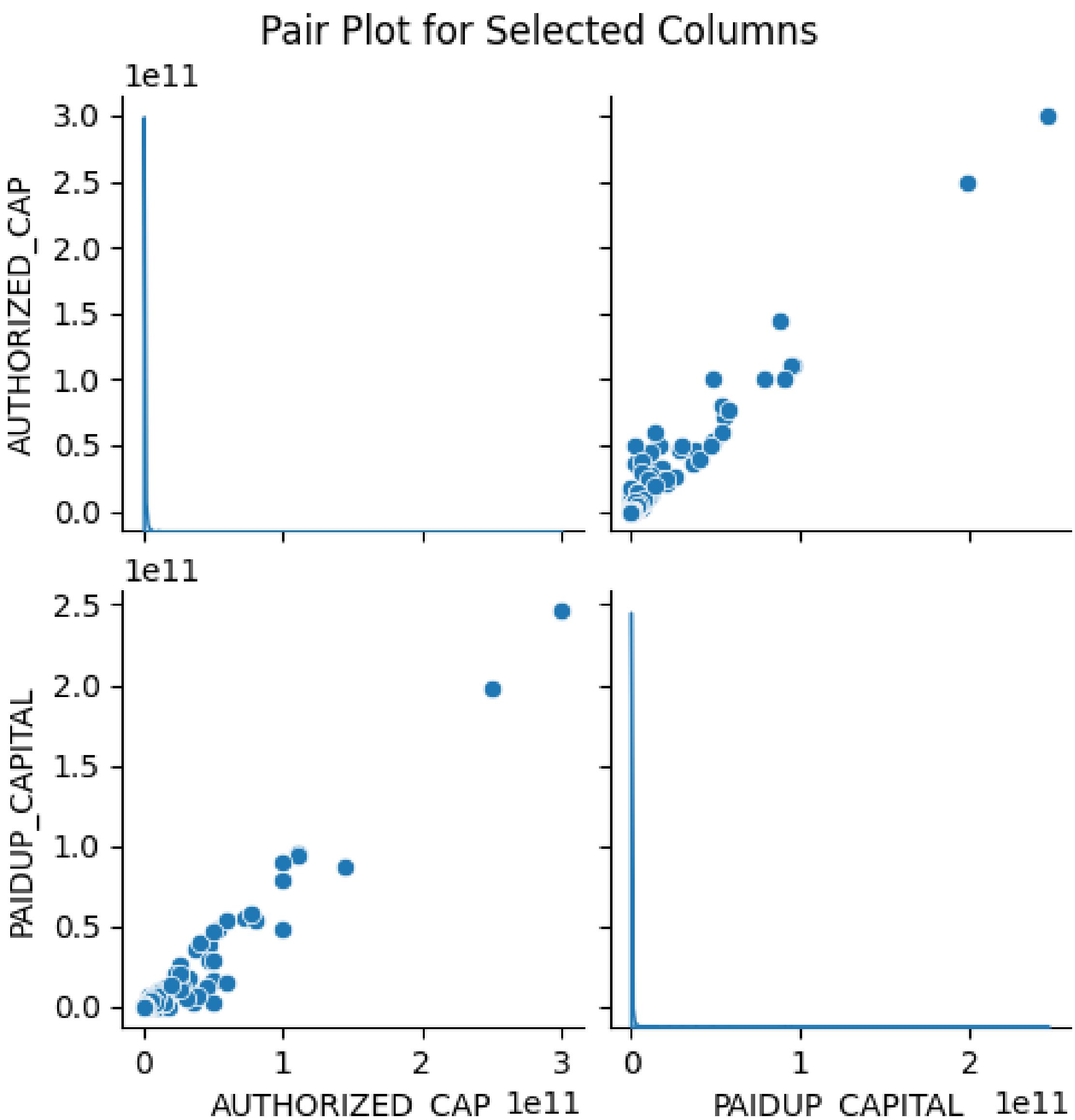



```
# Bivariate analysis: Categorical vs. Numerical (COMPANY_CATEGORY vs.
AUTHORIZED_CAP)

plt.figure(figsize=(12, 6))

sns.boxplot(x='COMPANY_CATEGORY', y='AUTHORIZED_CAP',
data=selected_df)

plt.title('COMPANY_CATEGORY vs. AUTHORIZED_CAP')

plt.xlabel('COMPANY_CATEGORY')

plt.ylabel('AUTHORIZED_CAP')

plt.xticks(rotation=45)
```

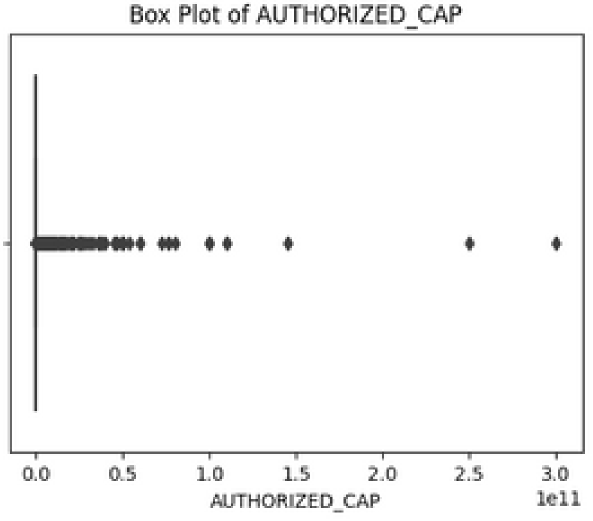
```
plt.show()
```

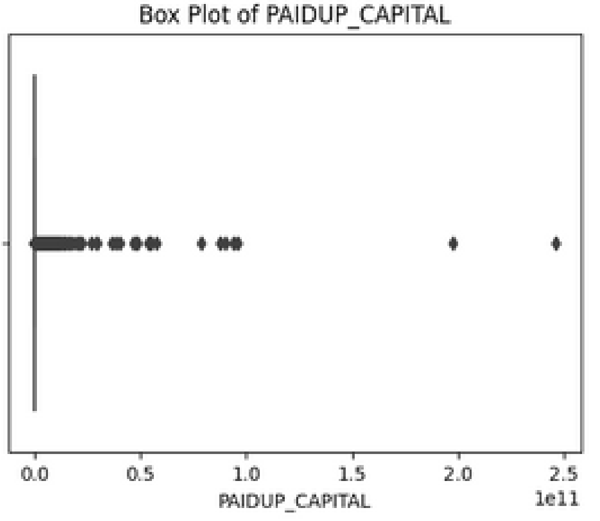



```
# Plot the pair plot
sns.pairplot(selected_df, diag_kind='kde', height=2.5)
plt.suptitle('Pair Plot for Selected Columns', y=1.02)
plt.show()
```




```
# Box plots for numerical columns
numerical_cols = ['AUTHORIZED_CAP', 'PAIDUP_CAPITAL']
plt.figure(figsize=(12, 4))
for i, col in enumerate(numerical_cols, 1):
    plt.subplot(1, 2, i)
    sns.boxplot(x=data_cleaned[col])
    plt.title(f'Box Plot of {col}')
```

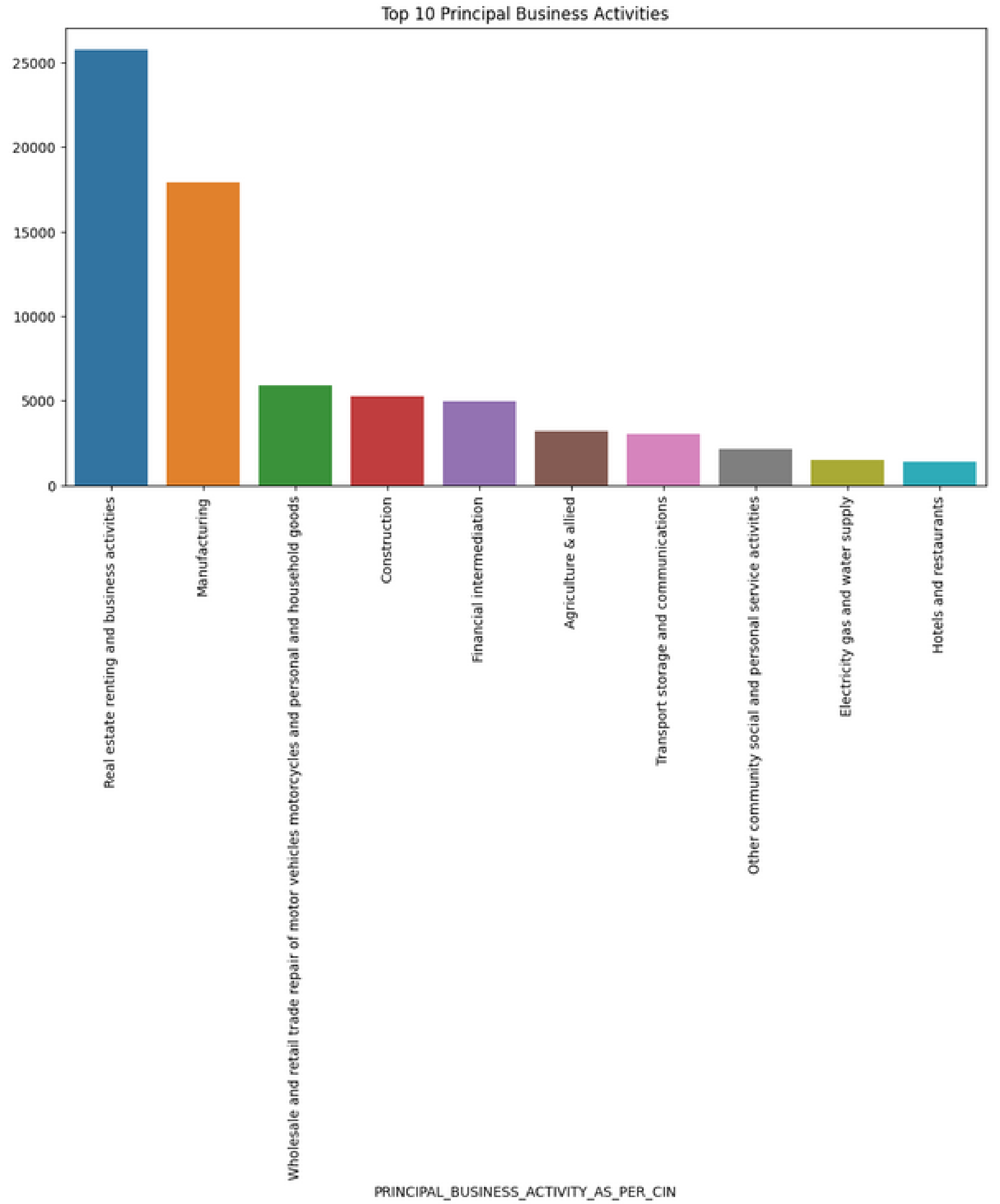





```
# Principal Business Activity - Top N categories

top_n = 10

plt.figure(figsize=(12, 6))

top_n_activities =
data_cleaned['PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN'].value_counts()
[:top_n]

sns.barplot(x=top_n_activities.index, y=top_n_activities.values)

plt.title(f'Top {top_n} Principal Business Activities')

plt.xticks(rotation=90)
```

Output

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'Real estate renting and business activities'),
  Text(1, 0, 'Manufacturing'),
  Text(2, 0, 'Wholesale and retail trade repair of motor vehicles motorcycles and personal and household goods'),
  Text(3, 0, 'Construction'),
  Text(4, 0, 'Financial intermediation'),
  Text(5, 0, 'Agriculture & allied'),
  Text(6, 0, 'Transport storage and communications'),
  Text(7, 0, 'Other community social and personal service activities'),
  Text(8, 0, 'Electricity gas and water supply'),
  Text(9, 0, 'Hotels and restaurants')])
```



EDA Multivariate Analysis

Multivariate analysis is one of the most useful methods to determine relationships and analyze patterns for any dataset.

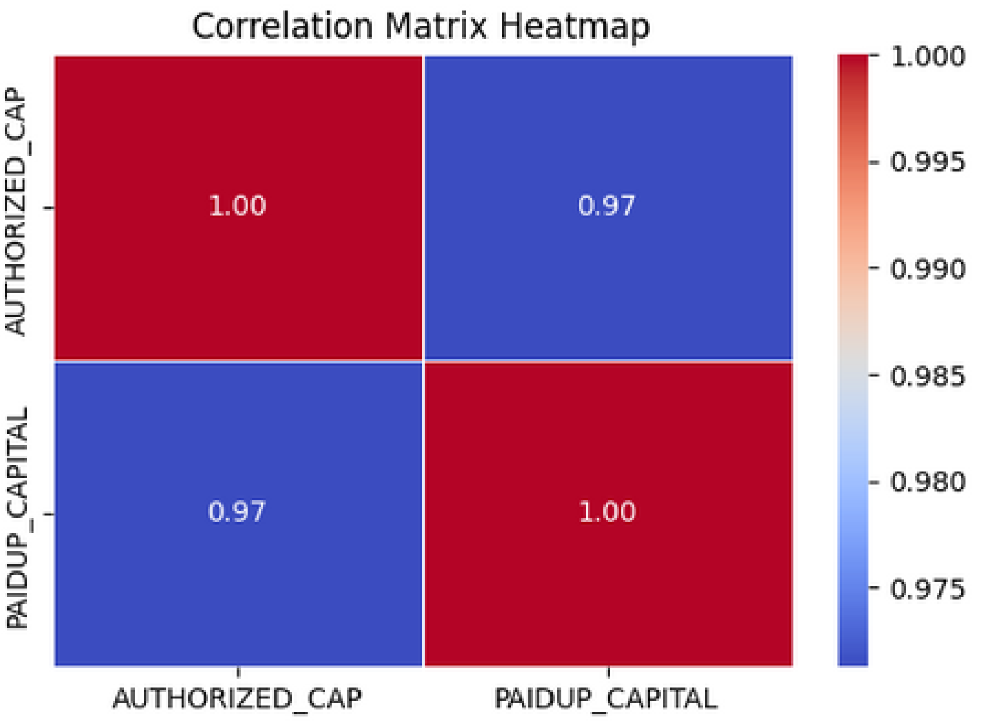
A heat map is widely been used for Multivariate Analysis

Heat Map gives the correlation between the variables, whether it has a positive or negative correlation.

In our example heat map shows the correlation between the variables.

Program :

```
# Select the specified columns for analysis

columns_for_analysis = ['AUTHORIZED_CAP', 'PAIDUP_CAPITAL']


# Subset the DataFrame with the selected columns

selected_df = df[columns_for_analysis]


# Convert columns to numeric (if they're not already)

selected_df = selected_df.apply(pd.to_numeric, errors='coerce')


# Calculate the correlation matrix

correlation_matrix = selected_df.corr()


# Plot the heatmap

plt.figure(figsize=(6, 4))

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', 
fmt='.2f', linewidths=0.5)

plt.title('Correlation Matrix Heatmap')

plt.show()
```



Feature Engineering

Feature engineering is a critical step in preparing data for machine learning models. In the context of predicting Company Registration Trends with the Registrar of Companies (RoC) data, feature engineering involves transforming and creating new features from the given columns to improve the model's predictive power. Below is a Python code for feature engineering

Program

```
# Feature 1: Extract Year from 'DATE_OF_REGISTRATION'

data['REGISTRATION_YEAR'] =
pd.to_datetime(data['DATE_OF_REGISTRATION'],format='%d-%m-%Y').dt.year
```

```
In [37]: data['REGISTRATION_YEAR'].reset_index()
Out[37]:
```

	index	REGISTRATION_YEAR
0	0	1961.0
1	1	NaN
2	2	1982.0
3	3	NaN
4	4	NaN
...	...	...
150866	150866	2016.0
150867	150867	2018.0
150868	150868	2016.0
150869	150869	2018.0
150870	150870	2019.0

150871 rows × 2 columns

```
# Feature 2: Label Encoding for 'COMPANY_STATUS'

label_encoder = LabelEncoder()

data['COMPANY_STATUS_CODE'] =
label_encoder.fit_transform(data['COMPANY_STATUS'])
```

```
In [40]: data['COMPANY_STATUS_CODE'].reset_index()
Out[40]:
```

	index	COMPANY_STATUS_CODE
0	0	7
1	1	0
2	2	0
3	3	7
4	4	0
...	...	...
150866	150866	0
150867	150867	0
150868	150868	8
150869	150869	0
150870	150870	0

150871 rows × 2 columns

```
# Feature 3: Calculate the ratio of 'PAIDUP_CAPITAL' to
'AUTHORIZED_CAP'
```

```
data['CAPITAL_RATIO'] = data['PAIDUP_CAPITAL'] /
data['AUTHORIZED_CAP']
```

```
In [47]: data['CAPITAL_RATIO']
Out[47]: 0              NaN
         1              NaN
         2              NaN
         3              NaN
         4              NaN
                       ...
         150866    0.100000
         150867    1.000000
         150868    0.200000
         150869    0.600000
         150870    0.733333
         Name: CAPITAL_RATIO, Length: 150871, dtype: float64
```

```
# Feature 4: Extract 'LATEST_YEAR_ANNUAL_RETURN' year

data['ANNUAL_RETURN_YEAR'] =
data['LATEST_YEAR_ANNUAL_RETURN'].str.extract('(\d+)').astype(float)
```

```
In [51]: data['ANNUAL_RETURN_YEAR']
Out[51]: 0          NaN
         1          NaN
         2          NaN
         3          NaN
         4          NaN
                   ...
         150866    31.0
         150867     NaN
         150868     NaN
         150869    31.0
         150870     NaN
         Name: ANNUAL_RETURN_YEAR, Length: 150871, dtype: float64
```

Model Training

1.Random Forest Algorithm

Program :

```
# Import necessary libraries

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

import matplotlib.pyplot as plt

import seaborn as sns


# Load the dataset

data = pd.read_csv("D://Course/AI
IBM/Data_Gov_Tamil_Nadu.csv",encoding='latin-1')


# Data Preprocessing
```

```
# Drop irrelevant columns

data = data[['COMPANY_STATUS', 'COMPANY_CLASS', 'COMPANY_CATEGORY',
'AUTHORIZED_CAP',

             'PAIDUP_CAPITAL',
'PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN']]


# Handle missing values if necessary

data.dropna(inplace=True)


# Encode categorical features

label_encoders = {}

categorical_columns = ['COMPANY_CLASS', 'COMPANY_CATEGORY',
'PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN']


for column in categorical_columns:

    label_encoders[column] = LabelEncoder()

    data[column] = label_encoders[column].fit_transform(data[column])


# Encode the target variable 'COMPANY_STATUS'

label_encoder_y = LabelEncoder()

data['COMPANY_STATUS'] =
label_encoder_y.fit_transform(data['COMPANY_STATUS'])


# Split the dataset into features (X) and target (y)

X = data.drop('COMPANY_STATUS', axis=1)

y = data['COMPANY_STATUS']


# Split the dataset into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)


# Model Training (Random Forest)
```

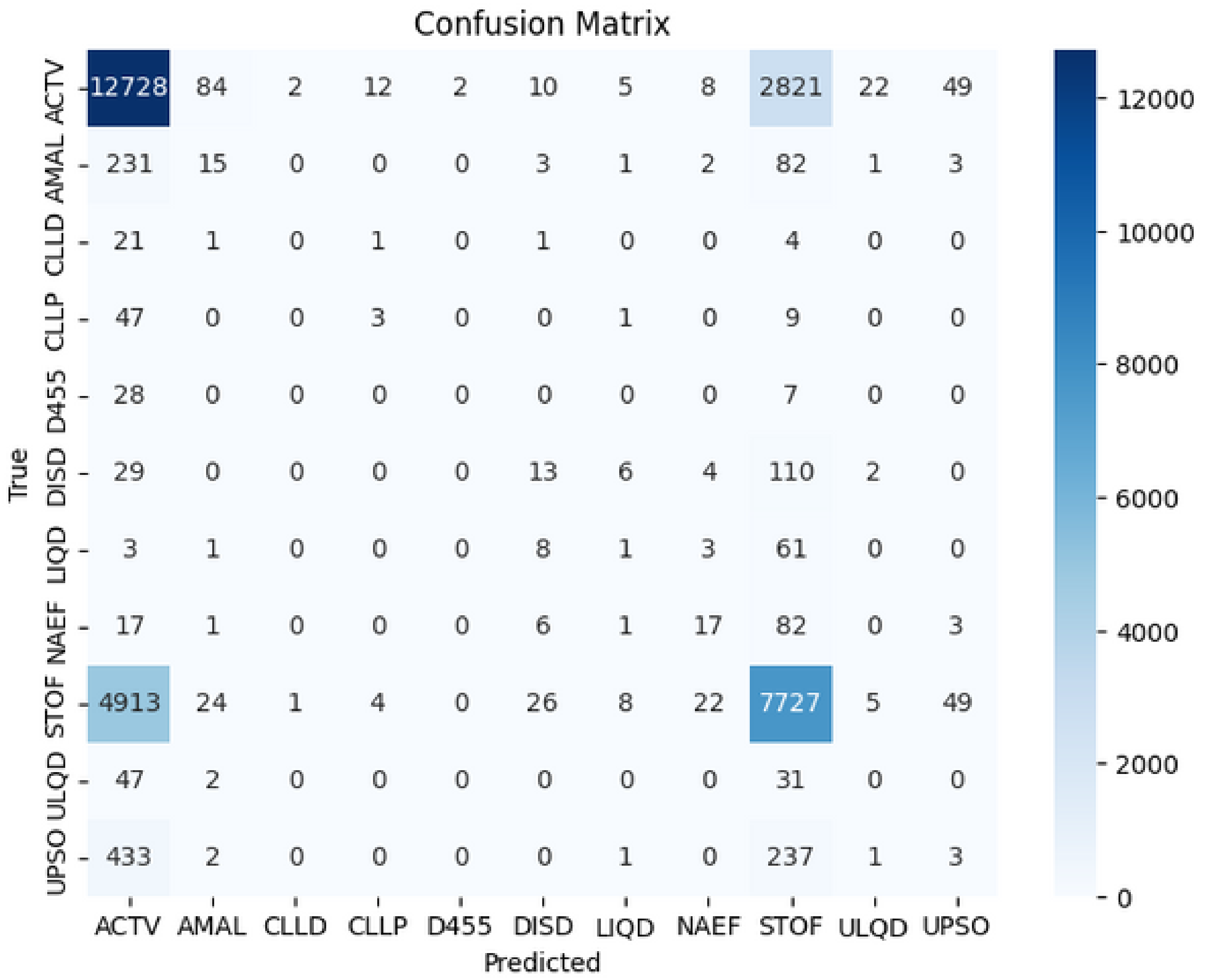
```
model = RandomForestClassifier()

model.fit(X_train, y_train)


# Model Evaluation

y_pred = model.predict(X_test)


# Decode the encoded target variable back to its original form

y_pred_decoded = label_encoder_y.inverse_transform(y_pred)


# Calculate accuracy

accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy}")


# Classification Report

report = classification_report(y_test, y_pred,
target_names=label_encoder_y.classes_)

print("Classification Report:\n", report)


# Confusion Matrix

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=label_encoder_y.classes_,

            yticklabels=label_encoder_y.classes_)

plt.xlabel('Predicted')

plt.ylabel('True')

plt.title('Confusion Matrix')

plt.show()
```

Output :

Accuracy: 0.6811146539125814

Classification Report:

	precision	recall	f1-score	support
ACTV	0.69	0.81	0.74	15743
AMAL	0.12	0.04	0.06	338
CLLD	0.00	0.00	0.00	28
CLLP	0.15	0.05	0.07	60
D455	0.00	0.00	0.00	35
DISD	0.19	0.08	0.11	164
LIQD	0.04	0.01	0.02	77
NAEF	0.30	0.13	0.19	127
STOF	0.69	0.60	0.65	12779
ULQD	0.00	0.00	0.00	80
UPSO	0.03	0.00	0.01	677
accuracy			0.68	30108
macro avg	0.20	0.16	0.17	30108
weighted avg	0.66	0.68	0.67	30108



Model conclusion

The classification results indicate that the model's accuracy is approximately 68.11%. The classification report provides a more detailed evaluation of the model's performance for each class in the 'COMPANY_STATUS' target variable.

- Precision, Recall, and F1-Score: For each class, precision measures the proportion of correctly predicted positive instances, recall measures the proportion of actual positives correctly predicted, and the F1-score is the harmonic mean of precision and recall. These metrics vary widely among the different classes, reflecting the model's ability to correctly classify instances for each category.

- 'ACTV' and 'STOF' Classes: The 'ACTV' and 'STOF' classes have relatively higher precision, recall, and F1-scores, indicating that the model performs relatively well for these classes.

- Low-Performing Classes: Several classes, such as 'AMAL,' 'CLLD,' 'CLLP,' 'D455,' 'LIQD,' 'ULQD,' and 'UPSO,' have low precision, recall, and F1-scores. This suggests that the model struggles to correctly classify instances within these classes.

- Overall: The weighted average F1-score is around 0.67, which means the model performs reasonably well, but there is room for improvement.

2. XGBOOST Algorithm

Program :

```
# Import necessary libraries

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from xgboost import XGBClassifier

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
```

```
import matplotlib.pyplot as plt

import seaborn as sns


# Load the dataset

data = pd.read_csv("D://Course/AI
IBM/Data_Gov_Tamil_Nadu.csv",encoding='latin-1')


# Data Preprocessing

# Drop irrelevant columns

data = data[['COMPANY_STATUS', 'COMPANY_CLASS', 'COMPANY_CATEGORY',
'AUTHORIZED_CAP',

             'PAIDUP_CAPITAL',
'PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN']]


# Handle missing values if necessary

data.dropna(inplace=True)


# Encode categorical features

label_encoders = {}

categorical_columns = ['COMPANY_CLASS', 'COMPANY_CATEGORY',
'PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN']


for column in categorical_columns:

    label_encoders[column] = LabelEncoder()

    data[column] = label_encoders[column].fit_transform(data[column])


# Encode the target variable 'COMPANY_STATUS'

label_encoder_y = LabelEncoder()

data['COMPANY_STATUS'] =
label_encoder_y.fit_transform(data['COMPANY_STATUS'])


# Split the dataset into features (X) and target (y)
```

```
X = data.drop('COMPANY_STATUS', axis=1)

y = data['COMPANY_STATUS']


# Split the dataset into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)


# Model Training (XGBoost)

model = XGBClassifier()

model.fit(X_train, y_train)


# Model Evaluation

y_pred = model.predict(X_test)


# Decode the encoded target variable back to its original form

y_pred_decoded = label_encoder_y.inverse_transform(y_pred)


# Calculate accuracy

accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy}")


# Classification Report

report = classification_report(y_test, y_pred,
target_names=label_encoder_y.classes_)

print("Classification Report:\n", report)


# Confusion Matrix

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=label_encoder_y.classes_,
```

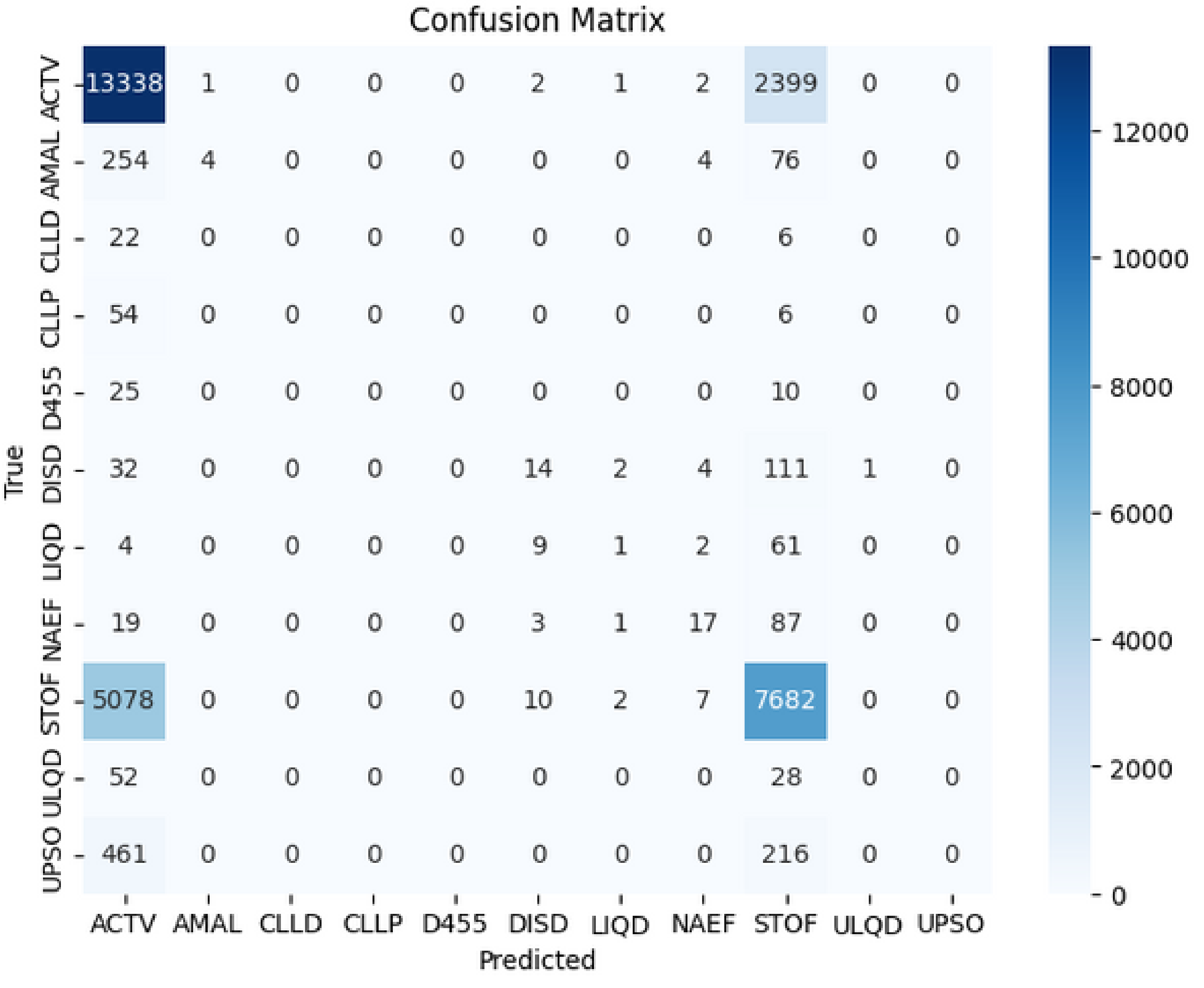
```
            yticklabels=label_encoder_y.classes_)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```

Output

Accuracy: 0.6993490102298392

Classification Report:

	precision	recall	f1-score	support
ACTV	0.69	0.85	0.76	15743
AMAL	0.80	0.01	0.02	338
CLLD	0.00	0.00	0.00	28
CLLP	0.00	0.00	0.00	60
D455	0.00	0.00	0.00	35
DISD	0.37	0.09	0.14	164
LIQD	0.14	0.01	0.02	77
NAEF	0.47	0.13	0.21	127
STOF	0.72	0.60	0.65	12779
ULQD	0.00	0.00	0.00	80
UPSO	0.00	0.00	0.00	677
accuracy			0.70	30108
macro avg	0.29	0.15	0.16	30108
weighted avg	0.68	0.70	0.68	30108



Model conclusion

The XGBoost algorithm produced a classification model with an accuracy of approximately 0.70, indicating that the model can correctly predict the target classes for about 70% of the instances in the dataset. However, a deeper analysis of the classification report reveals some important insights.

The precision scores for most classes are quite low, indicating that the model tends to generate false positives for these classes. The highest precision is for "AMAL," but this class has a very low recall, suggesting that the model struggles to correctly identify instances of this class.

Additionally, several classes, such as "CLLD," "CLLP," "D455," "ULQD," and "UPSO," have very low precision and recall, indicating that the model has significant difficulty distinguishing these classes.

The macro average F1-score is 0.16, reflecting the overall balance between precision and recall across all classes. The weighted average F1-score is slightly higher at 0.68, which takes into account class imbalances.

Conclusion

In conclusion, the AI-driven exploration and prediction of company registration trends with the Registrar of Companies (RoC) represents a powerful tool for government agencies, businesses, and policymakers alike.

By leveraging advanced data analysis and artificial intelligence techniques, this approach enables us to gain deep insights into historical registration trends, identify emerging patterns, and make informed predictions for the future. It not only streamlines the regulatory processes for the RoC but also facilitates better decision-making in areas such as economic planning, resource allocation, and industry-specific interventions.

As AI technologies continue to advance, this innovative approach will play an increasingly vital role in shaping the landscape of company registrations, fostering economic growth, and ensuring that both businesses and regulators are well-equipped to adapt to evolving market dynamics.

With the ability to adapt to changing business environments, the AI-driven exploration and prediction of company registration trends with RoC is poised to be a valuable asset in the realm of data-driven governance and commerce.