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

Phase 4

Devolpment Part 2

**Topic :**

EDA , Feature engineering , Model training

**922121106006**

**Bala Krishnan**

**au922121106006**

**SSMIET**

**Introduction**

AI-Driven Exploration and Prediction of Company Registration Trends with the Registrar of Companies (RoC) involves leveraging artificial intelligence (AI) methodologies to analyze data related to company registrations maintained by the Registrar of Companies. The Registrar of Companies is an authoritative entity responsible for overseeing and maintaining the registry of companies within a specific jurisdiction.

By employing AI algorithms, this approach aims to extract valuable insights and forecast patterns from the data compiled by the RoC. These insights can aid in understanding trends, emerging patterns, and other significant aspects of company registrations, empowering stakeholders to make informed decisions in the business landscape.

The utilization of AI in this domain encompasses data collection, processing, exploratory data analysis, machine learning modeling, and predictive analytics to anticipate future trends in company registrations. Ultimately, this AI-driven approach enables proactive decision-making and strategic planning based on comprehensive analyses of registration trends and associated data.

**Overview**

**For Phase 4**

**1.Data collecting**

**2.Exploratory Data Analysis (EDA)**

**Univariate Analysis**

**Bivariate Analysis**

**Multivariate Analysis**

**3.Feature Engineering**

**4.Model Training**

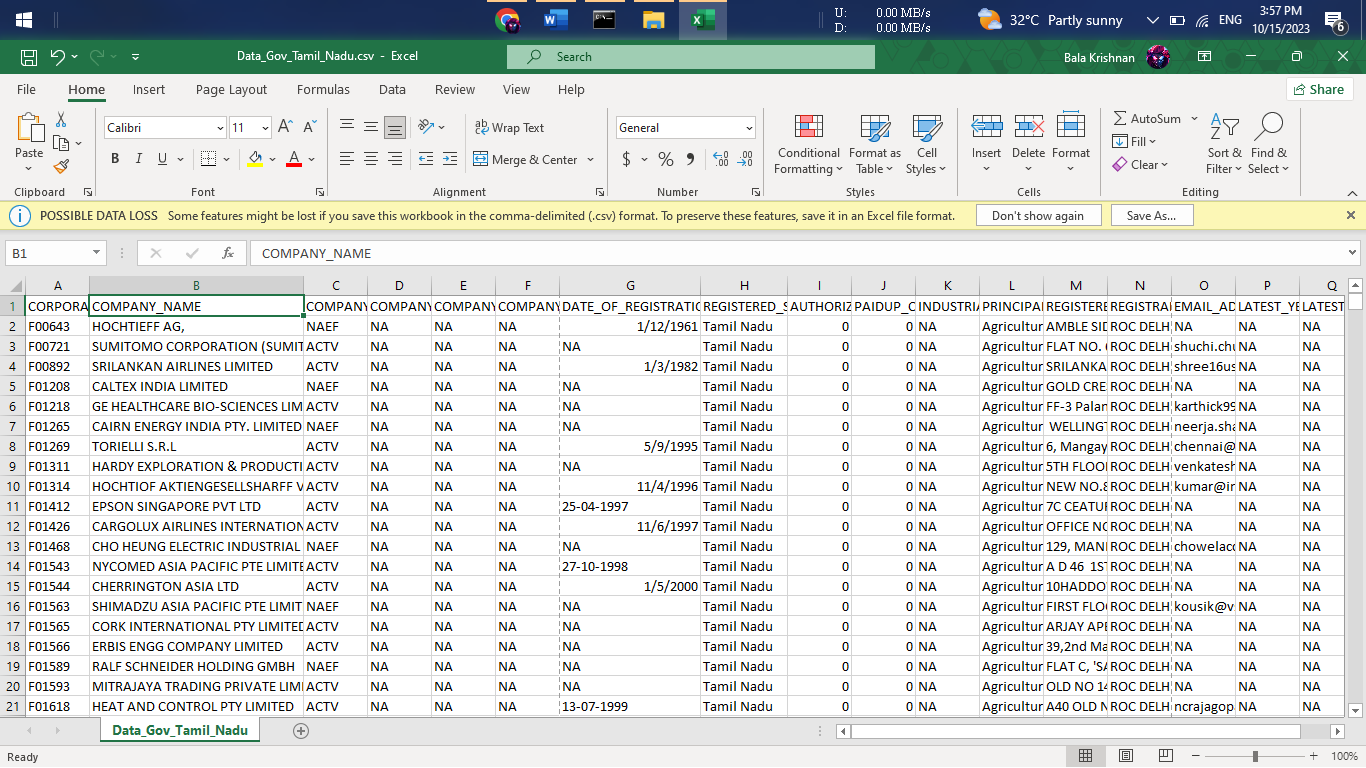
**Random Forest Algorithm**

**Xgboost Algorithm**

**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**

****

**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','REGISTERED\_STATE','AUTHORIZED\_CAP','PAIDUP\_CAPITAL','INDUSTRIAL\_CLASS','PRINCIPAL\_BUSINESS\_ACTIVITY\_AS\_PER\_CIN','REGISTERED\_OFFICE\_ADDRESS','REGISTRAR\_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))

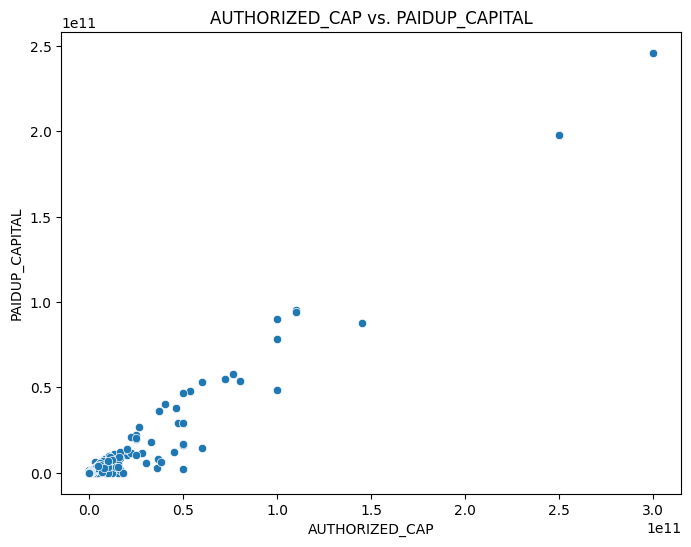
sns.scatterplot(x='AUTHORIZED\_CAP', y='PAIDUP\_CAPITAL', data=selected\_df)

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))

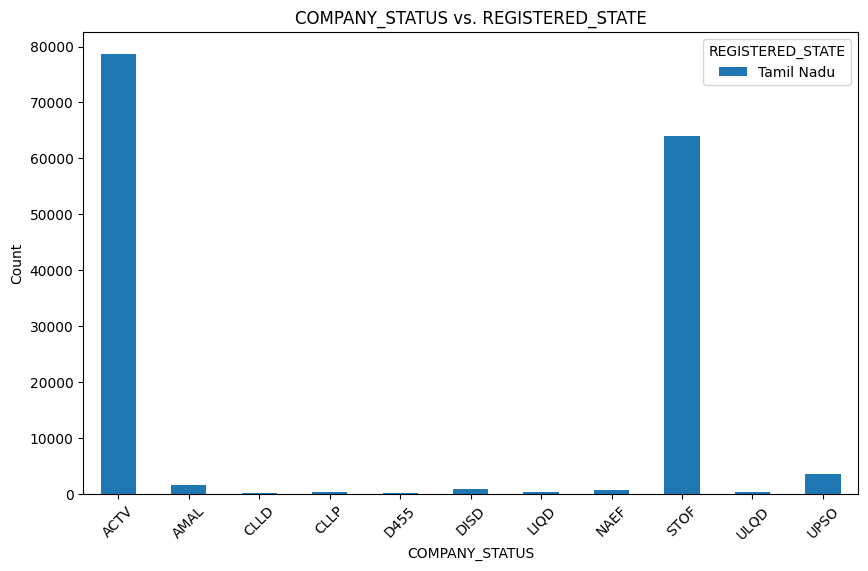
plt.title('COMPANY\_STATUS vs. REGISTERED\_STATE')

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)

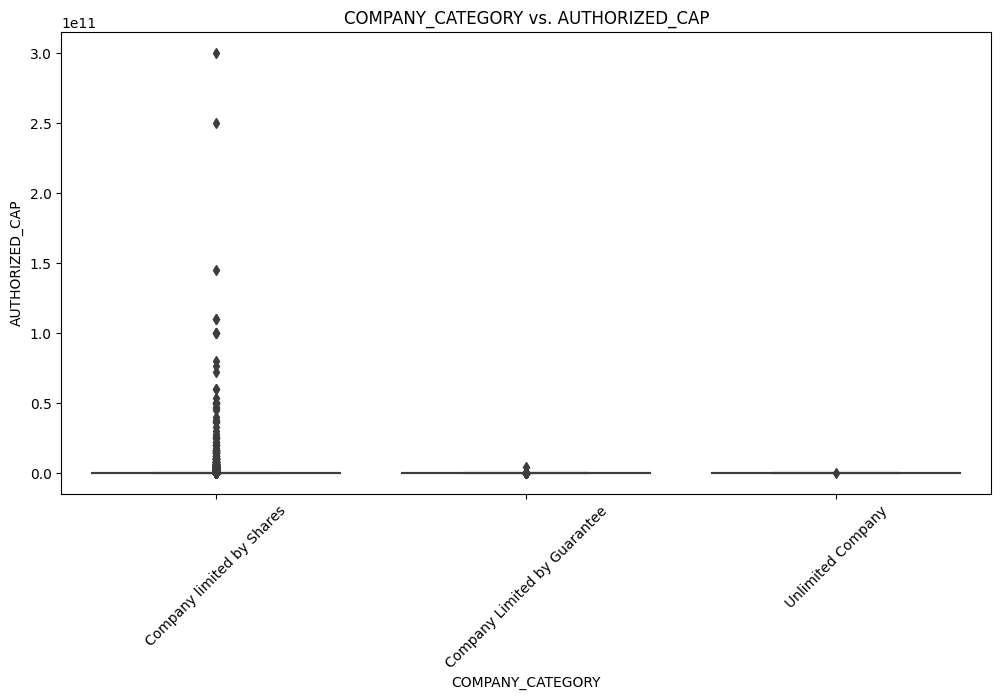
plt.title('COMPANY\_CATEGORY vs. AUTHORIZED\_CAP')

plt.xlabel('COMPANY\_CATEGORY')

plt.ylabel('AUTHORIZED\_CAP')

plt.xticks(rotation=45)

plt.show()

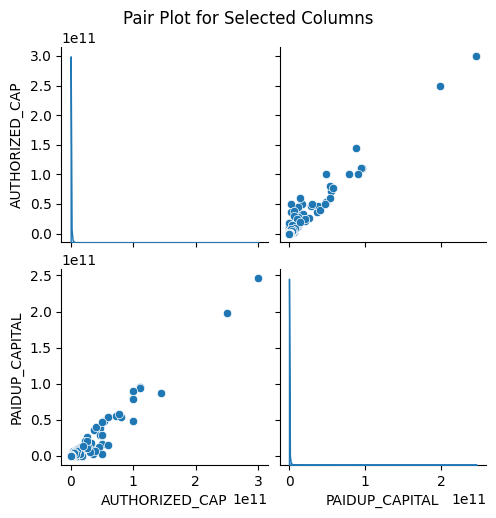


# Plot the pair plot

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']

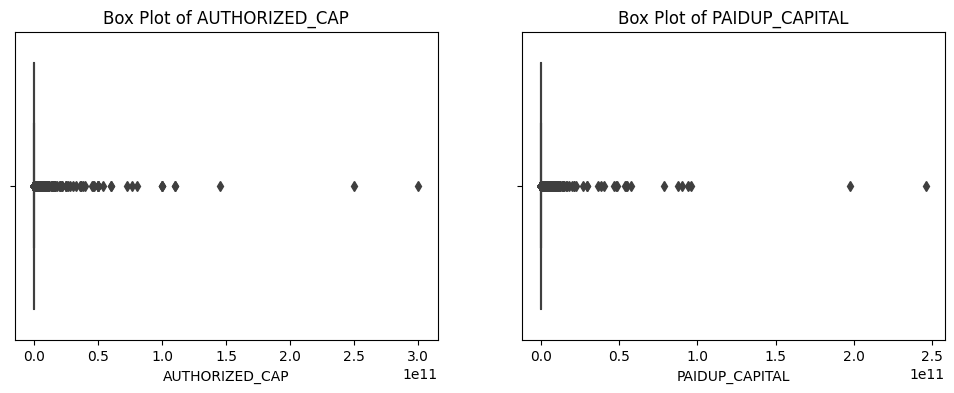
plt.figure(figsize=(12, 4))

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'),

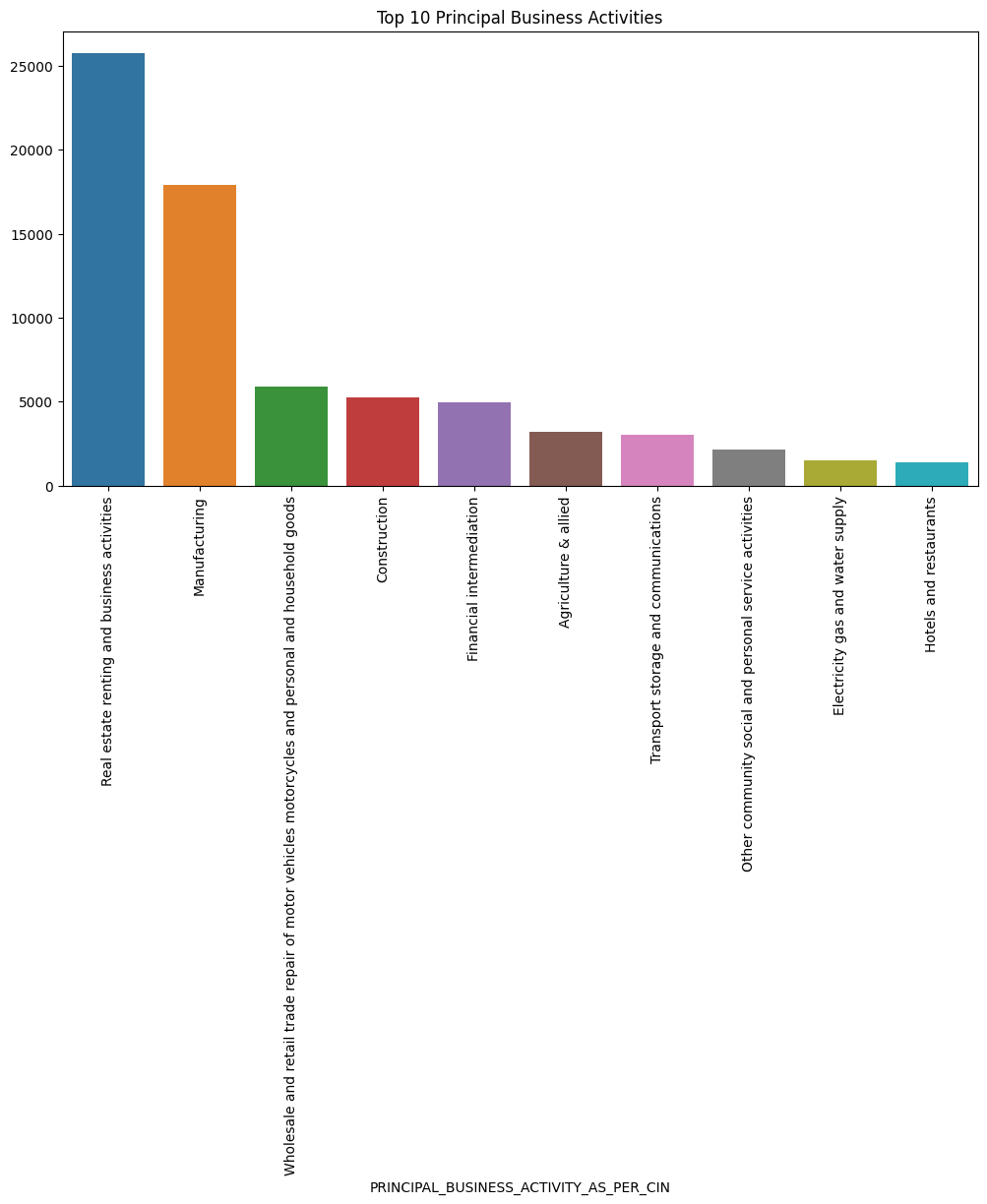
Text(5, 0, 'Agriculture & allied'),

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()

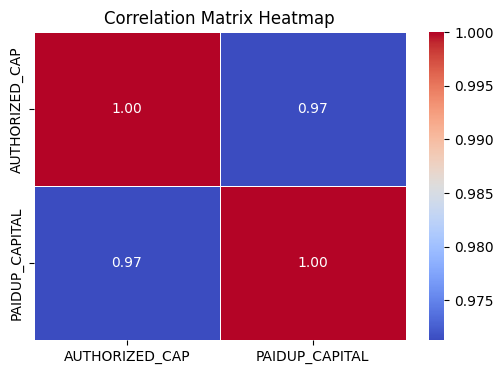
# Plot the heatmap

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()



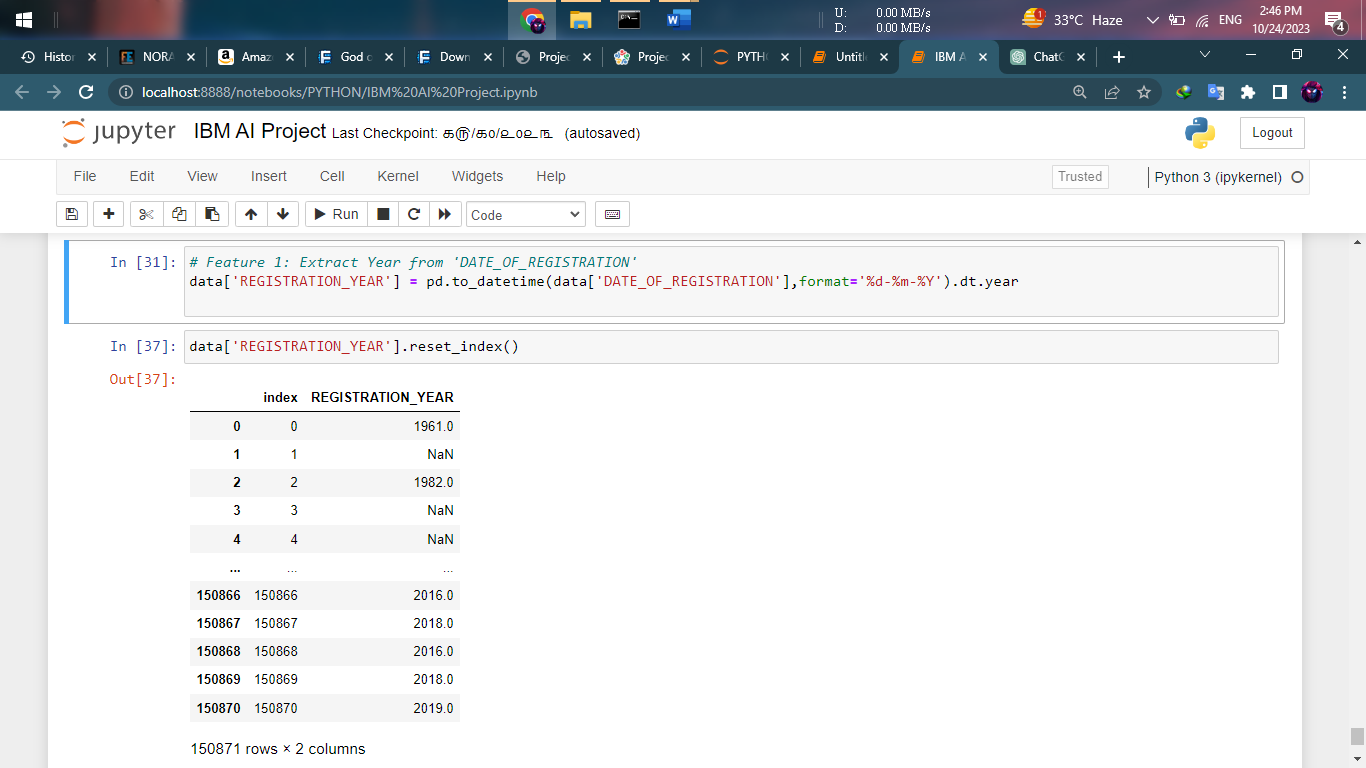
**Feature Engineering**

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'

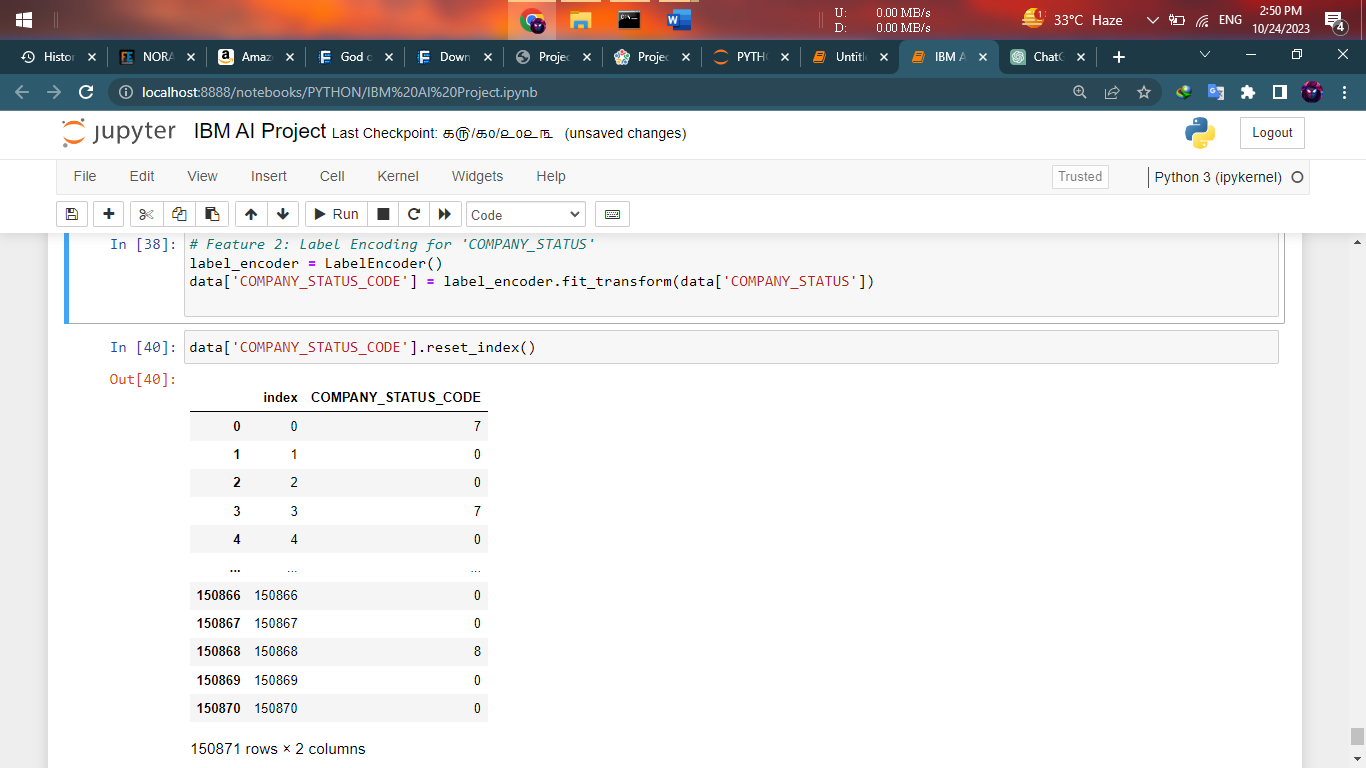
data['REGISTRATION\_YEAR'] = pd.to\_datetime(data['DATE\_OF\_REGISTRATION'],format='%d-%m-%Y').dt.year



# Feature 2: Label Encoding for 'COMPANY\_STATUS'

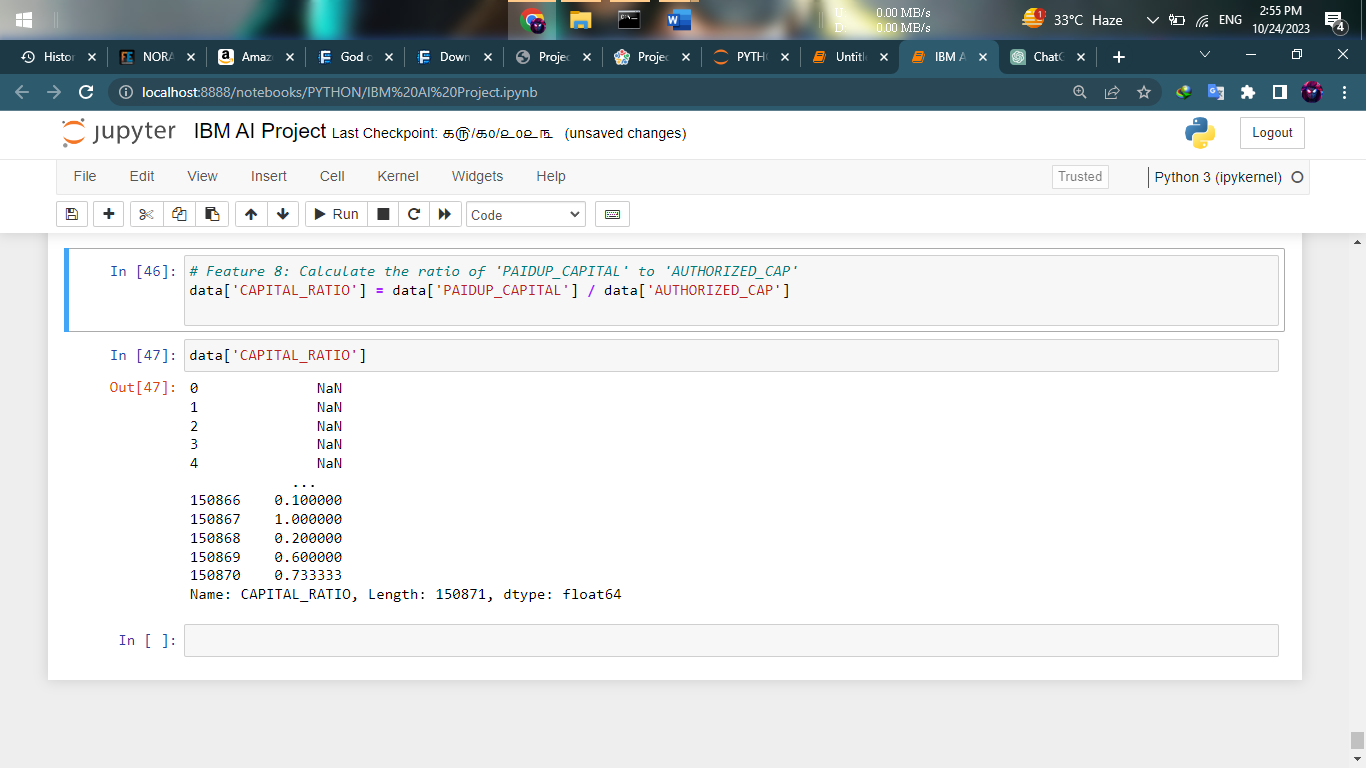
label\_encoder = LabelEncoder()

data['COMPANY\_STATUS\_CODE'] = label\_encoder.fit\_transform(data['COMPANY\_STATUS'])



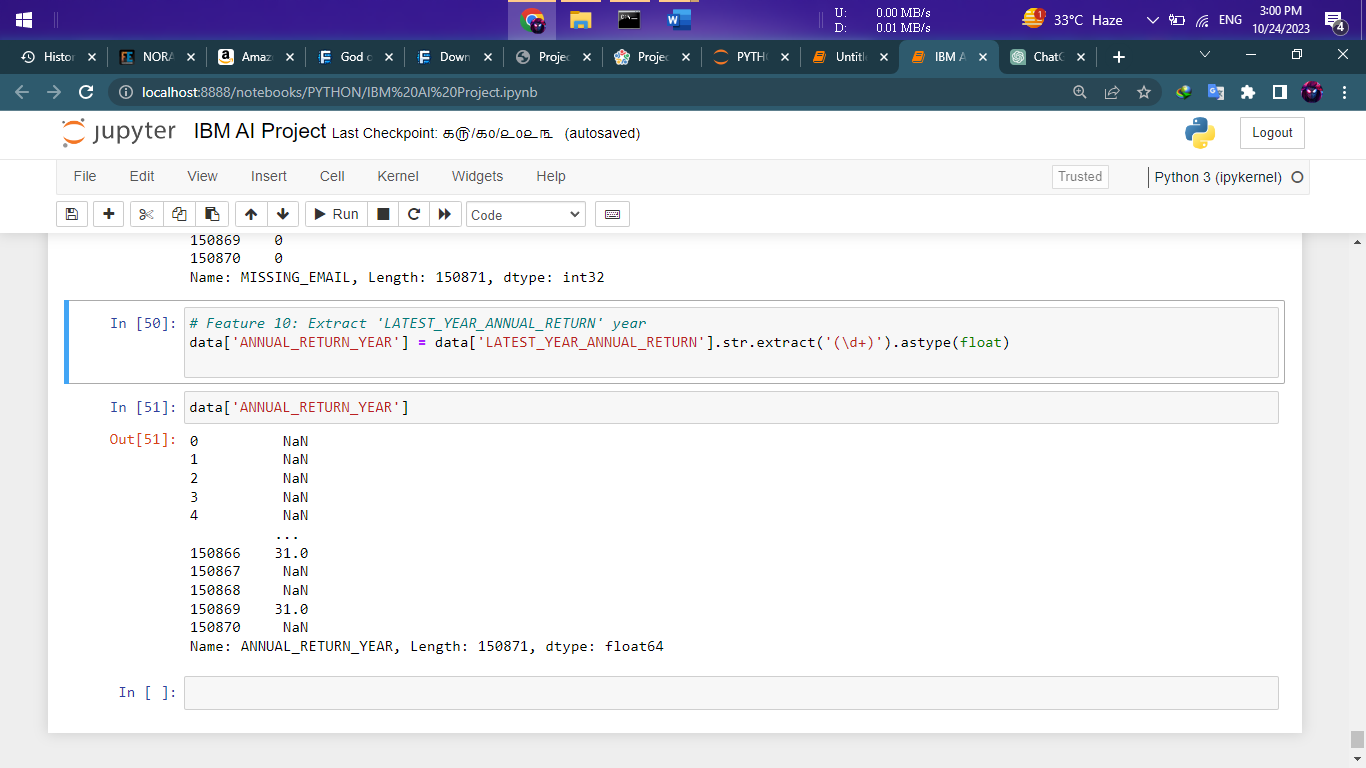
# Feature 3: Calculate the ratio of 'PAIDUP\_CAPITAL' to 'AUTHORIZED\_CAP'

data['CAPITAL\_RATIO'] = data['PAIDUP\_CAPITAL'] / data['AUTHORIZED\_CAP']



# Feature 4: Extract 'LATEST\_YEAR\_ANNUAL\_RETURN' year

data['ANNUAL\_RETURN\_YEAR'] = data['LATEST\_YEAR\_ANNUAL\_RETURN'].str.extract('(\d+)').astype(float)



**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

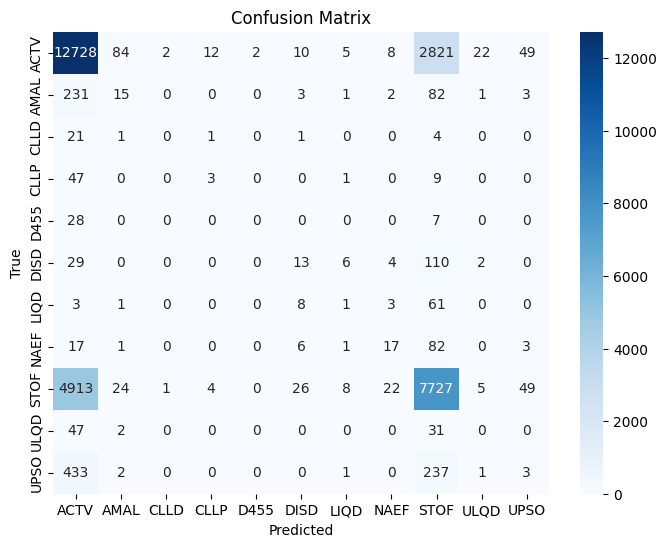
ULQD 0.00 0.00 0.00 80

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

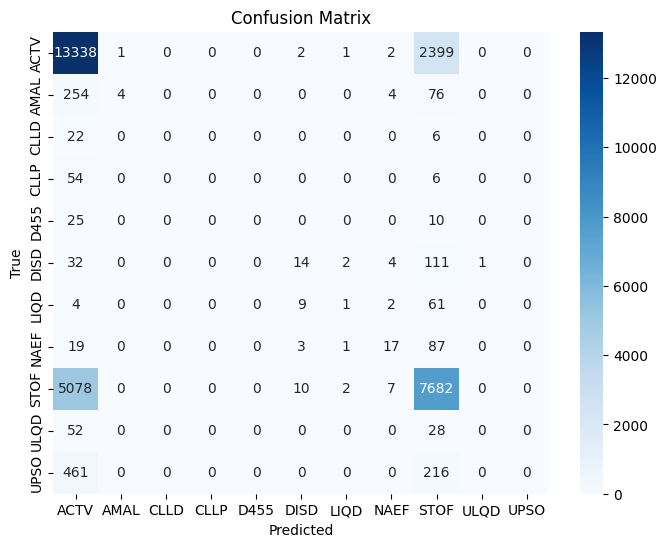
ULQD 0.00 0.00 0.00 80

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.