**7111-AI-Driven Exploration and Prediction of Company Registration Trends with Registrar of Companies (RoC)**

**TEAM MEMBER**

**711121106048-V.MAHIBA**

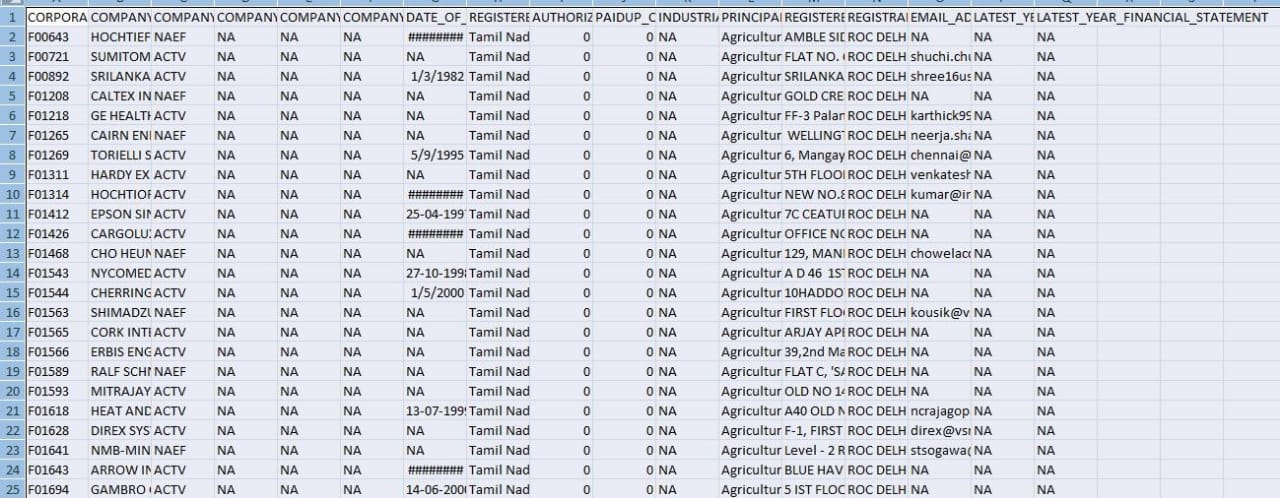
**Phase 4 :Development part 2**

**INTRODUCTION :**

**Company registration trends with RoC**

In the realm of commerce and industry, the Registrar of Companies (RoC) stands as a sentinel, diligently recording every new enterprise's inception. As businesses emerge and evolve, the data maintained by the RoC becomes a valuable testament to the ever-shifting economic landscape. In this era of unprecedented innovation and entrepreneurship, understanding and interpreting Company Registration Trends with RoC data has become indispensable. This exploration delves into the intricacies of registration patterns, regional dynamics, and industry preferences. With a focus on both historical and emerging trends, this discussion aims to provide a comprehensive view of the corporate world's heartbeat, offering insights that can steer strategic decisions, inform government policies, and fuel the engines of economic growth. Join us in this enlightening journey as we navigate the intricate tapestry of Company Registration Trends with the Registrar of Companies.In this phase we have to preprocessing and analyse the given dataset.

**GIVEN DATASET :**

****

150872 Rows × 12 Columns

NECESSARY STEPS TO FOLLOW :

1.import libraries

2.Load the Dataset

3.Exploratory Data Analysis(EDA)

DATA PREPROCESSING

Data preprocessing is a critical step in any AI-driven exploration and prediction task, including the analysis of company registration trends. Properly processed data helps ensure the accuracy and effectiveness of your predictive models. Here's a step-by-step guide on how to preprocess data for this specific task:

1. Data Collection:

- Gather relevant data sources, which might include historical company registration data, economic indicators, industry-specific information, and any other data that can potentially impact company registration trends.

2. Data Cleaning:

- Remove duplicates: Check for and eliminate duplicate records.

- Handle missing data: Identify and deal with missing values using techniques like imputation or removal.

- Correct inaccuracies: Address any inconsistencies or errors in the data.

3. Data Integration:

- If you have multiple data sources, integrate them into a single dataset. Ensure that common data fields are appropriately matched and merged.

4. Data Transformation:

- Convert data types: Ensure that data types are appropriate for analysis. Numeric data should be numerical, dates should be in date format, and categorical data should be properly labeled.

- Feature engineering: Create new features if they can provide valuable insights. For example, you might calculate monthly, quarterly, or yearly trends from daily data.

- Normalization or scaling: Scale numerical features if needed to bring them to the same range.

5. Data Reduction:

- Reduce dimensionality: If you have a high number of features, consider using techniques like Principal Component Analysis (PCA) or feature selection to reduce the dimensionality.

- Outlier detection: Identify and handle outliers using techniques like Z-score or IQR.

6. Data Encoding:

- Encode categorical variables: Convert categorical variables into numerical format using techniques like one-hot encoding or label encoding.

7. Data Splitting:

- Split your data into training, validation, and test sets to evaluate model performance. Typically, this is done in an 80-20 or 70-30 ratio.

8. Time Series Considerations:

- If you are dealing with time series data, pay special attention to time-based features, lag variables, and seasonality adjustments.

9. Data Exploration:

- Visualize the data to gain insights into trends, patterns, and correlations. Tools like histograms, scatter plots, and correlation matrices can be helpful.

10. Feature Scaling:

- Normalize or standardize your features, especially if you are using models like SVM or k-means clustering.

11. Data Imbalance:

- If there is an imbalance in your target variable (e.g., more non-registrations than registrations), consider techniques like oversampling, undersampling, or using different evaluation metrics.

12. Validation:

- Validate that your preprocessed data is suitable for your machine learning algorithms. This may involve testing your model on a small subset of data before full-scale training.

13. Documentation:

- Maintain thorough documentation of all your preprocessing steps. This documentation can be helpful in reproducing results and debugging issues.

14. Automate the Pipeline:

- Consider creating an automated data preprocessing pipeline, which can save time and ensure consistency in data preparation for ongoing analysis.

15. Model Building:

- Finally, use the preprocessed data to build and train your AI models for predicting company registration trends.

Continuous monitoring and adjustment of your data preprocessing pipeline are crucial as new data becomes available and as you update your models. This iterative approach ensures that your predictions remain accurate and relevant.

**IMPORTING LIBRARIES:**

Importing necessary libraries to do basic things on the Dataset

**Program:**

import pandas as pd

import numpy as np

import seaborne as sns

import matplotlib.pyplot as plt

print(‘Successfully imported’)

**Output :**

Successfully imported

**LOAD THE DATASET:**

Load your Dataset into a pandas dataframe .You can typically find company registration dataset in CSV format , but you can adapt this code to other formats as needed

**Program:**

data=pd.read\_csv(‘datagovtamilnadu.csv’,encoding=unicode\_escape’)

data.head()

**EXPLORATORY DATA ANALYSIS (EDA):**

Perforn EDA to understand your data better . This includes checking for missing values , exploring the data’s statistics and visualization it to identify patterns.

**Program:**

**#**check the dataset for missing data

if data.isnull().sum().sum()==0:

print(‘There is no missing data in our dataset’)

else:

print(‘There is {} missing data in ourdataset’.format(data.isnull().sum().sum()))

**Output:**

There is no missing data in our dataset

**Program :**

**In[1]**

frame=pd.concat([data.isnull().sum() , data.nunique() , data.dtypes] , axis=1, sort=False)

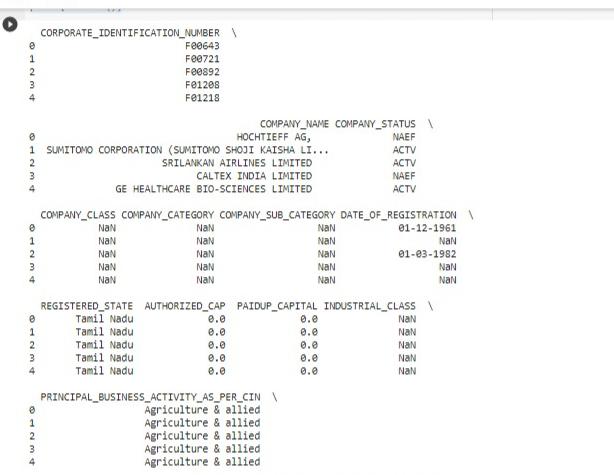
frame

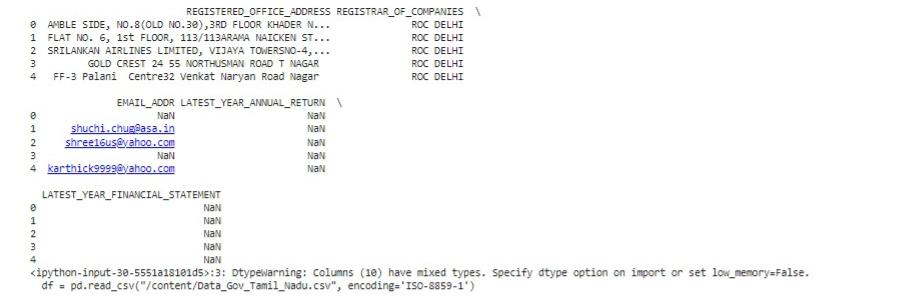
file\_path = 'Data\_Gov\_Tamil\_Nadu.csv'

df = pd.read\_csv("/content/Data\_Gov\_Tamil\_Nadu.csv", encoding='ISO-8859-1')

print(df.head())

**Out[1]:**

****

****

**In[2]**

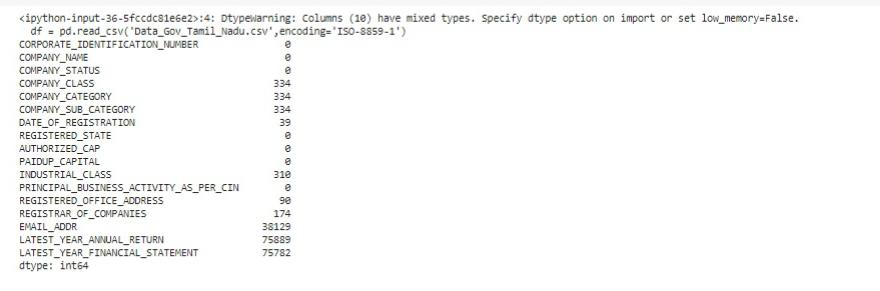
# Load the CSV file into a DataFrame

df = pd.read\_csv('Data\_Gov\_Tamil\_Nadu.csv',encoding='ISO-8859-1')

# Print the number of null values in first 10000 column

print(df.isnull().sum())

**Out[2]**

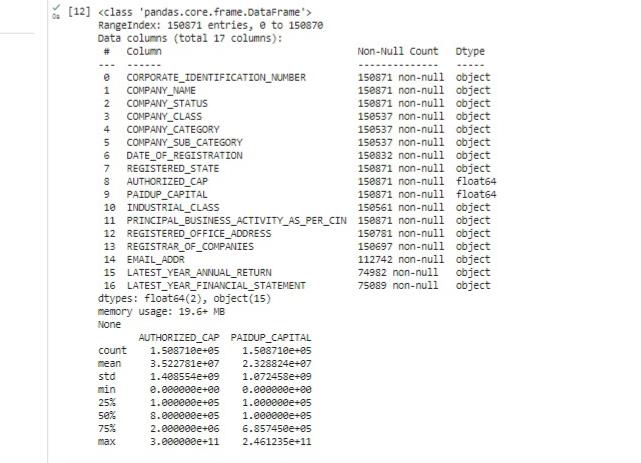
****

**In[3]**

print(df.info())

print(df.describe())

**Out[3]**

****

**VISUALIZATION AND PRE-PROCESSING OF DATA:**

In[1]

# Print the columns of your dataframe to verify the presence of the 'INDEX ' column

print(df.columns)

# If the 'COMPANY\_STATUS ' column is present in your dataframe

if 'COMPANY\_STATUS ' in df.columns:

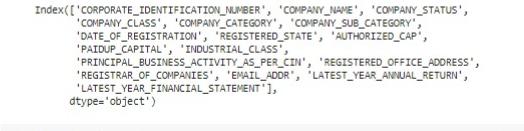
X = df.drop('COMPANY\_STATUS ', axis=1)

y = df['COUNTING']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

print(df.columns)

Out[1]



In[2]

Total\_AUTHORIZED\_CAP = df\_grp\_d['AUTHORIZED\_CAP'].sum()

Total\_PAIDUP\_CAPITAL = df\_grp\_d['PAIDUP\_CAPITAL'].sum()

Total\_INDUSTRIAL\_CLASS = df\_grp\_d['INDUSTRIAL\_CLASS'].sum()

data = [['AUTHORIZED\_CAP', Total\_AUTHORIZED\_CAP], ['PAIDUP\_CAPITAL', Total\_PAIDUP\_CAPITAL], ['INDUSTRIAL\_CLASS', Total\_INDUSTRIAL\_CLASS]]

df = pd.DataFrame(data, columns = ['COMPANY\_NAME', 'count'])

fig = px.pie(df,

values="count",

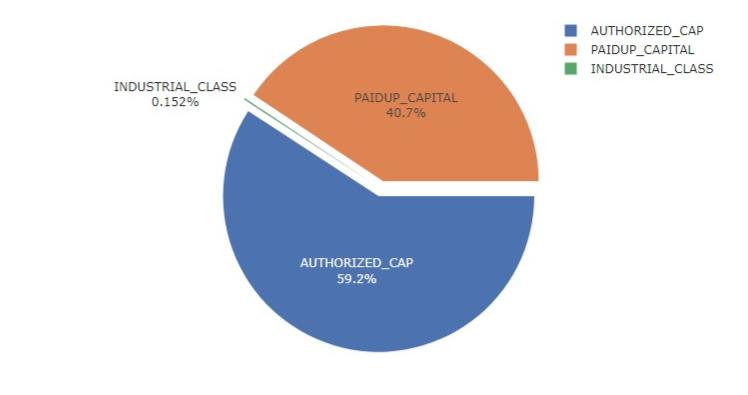
names="COMPANY\_NAME",

title="'Data\_Gov\_Tamil\_Nadu.csv ",

template="seaborn")

fig.update\_traces(rotation=90, pull=0.05, textinfo="percent+label")

fig.show()

Out[2]

**In[3]**

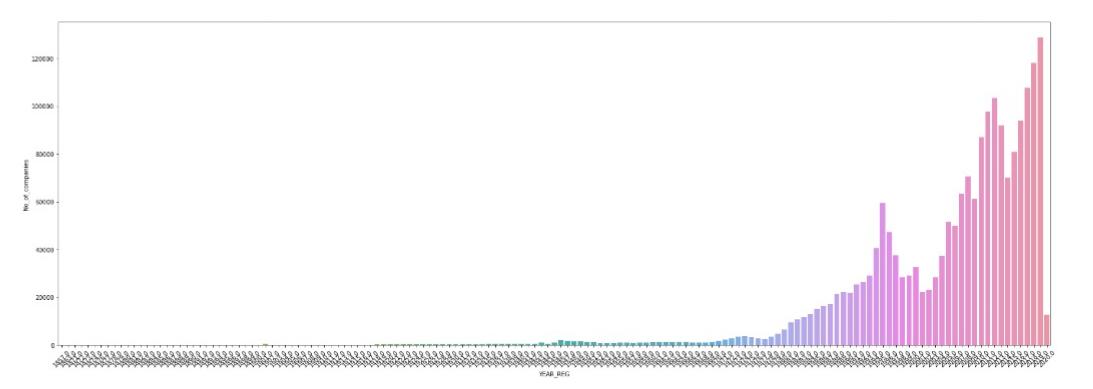
plt.figure(figsize=(30,10))

plt.xticks(rotation=45)

sns.barplot(data=number\_of\_companies,x='YEAR\_REG',y='No\_of\_companies')

**Out[3]**

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f2c8c5c6190>



**In[4]**

company\_type = data['company\_type'].value\_counts().keys().tolist()

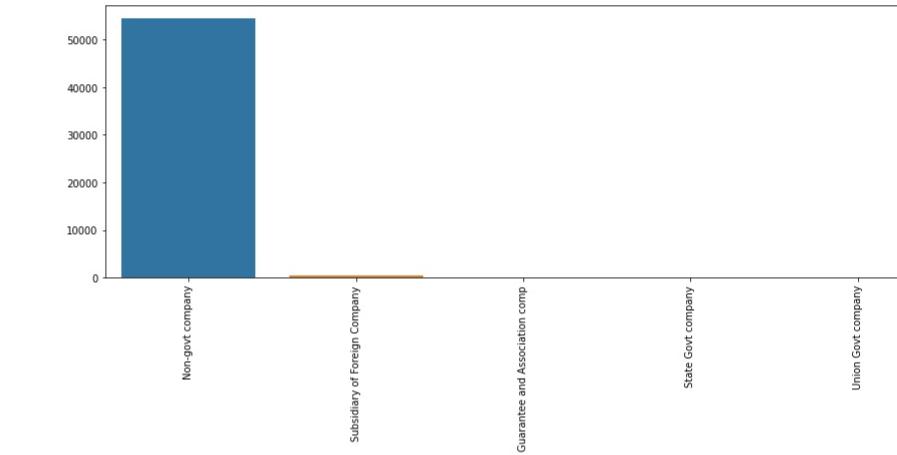
counts = data['company\_type'].value\_counts().tolist()

plt.figure(figsize = (15,5))

sns.barplot(company\_type,counts)

plt.xticks(rotation = 90)

plt.show()

**Out[4]**