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

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

**Phase-5: Project Documentation and Submission.**

**Documentation**

**Problem Statement**



The problem addressed in this project is the exploration and prediction of Company Registration trends with the Registrar of Companies (RoC). The objective is to leverage AI and data analysis techniques to understand historical company registration trends, identify potential influencing factors, and develop predictive models to forecast future registration patterns. The project aims to assist government bodies, businesses, and policymakers in making data-driven decisions related to company registrations.

**Design Thinking Process**

The project followed a design thinking process with the following key stages:

**Empathize**: Understanding the stakeholders' needs and concerns, such as government regulators, business owners, and investors, and defining the problem.

**Define**: Clearly articulating the problem statement and project goals, which includes exploring trends in company registration with RoC and building predictive models.

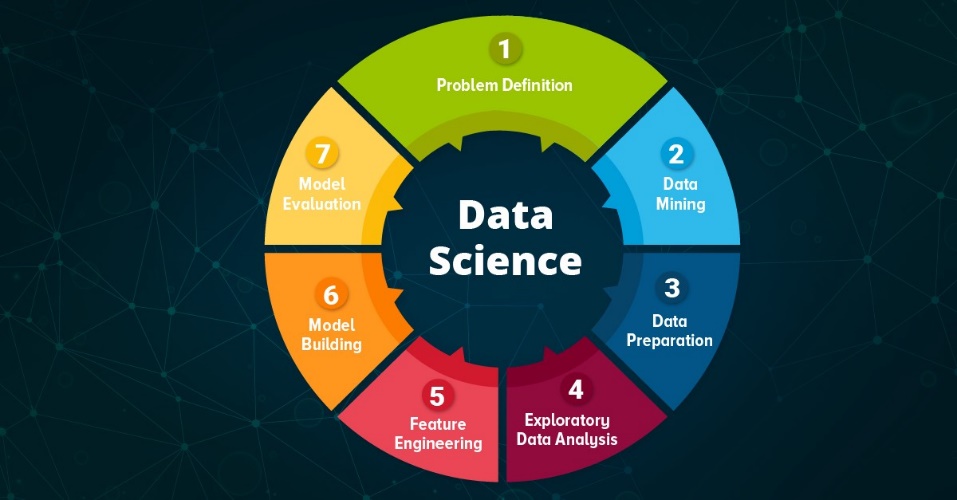
**Ideate**: Brainstorming various data sources, AI techniques, and methodologies to address the problem effectively.

**Prototype**: Developing a preliminary plan for data collection, preprocessing, model development, and evaluation.

**Test & Refine**: Iteratively refining the approach, based on data exploration and model performance.

**Implement & Scale**: Building the final solution and ensuring it's scalable for real-world applications.

**Phases of Development**



The development of this project involved several key phases:

**Data Collection**: Gathering historical company registration data from the RoC and other relevant datasets.

**Data Preprocessing**: Cleaning and transforming the data, handling missing values, and preparing it for analysis and modeling.

**Exploratory Data Analysis (EDA**): Analyzing the data to uncover insights, patterns, and relationships within the data, including company registration trends, regional variations, and potential influencing factors.

**Feature Engineering:** Creating relevant features from the data that could improve the performance of predictive models.

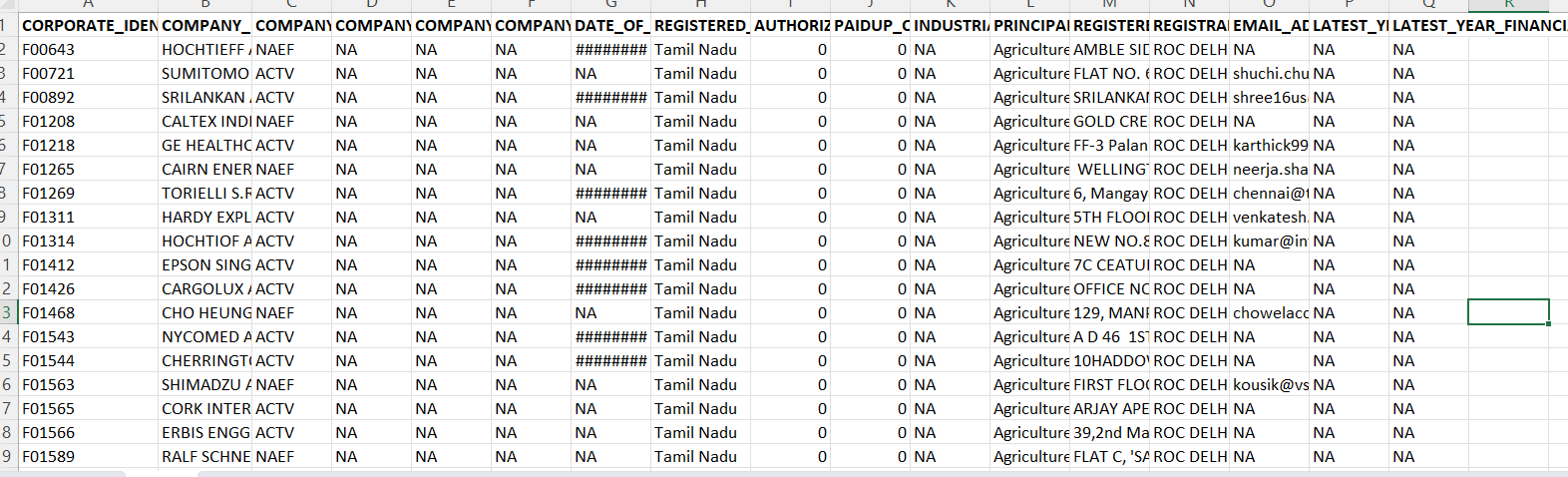
**Model Development:** Implementing AI algorithms, such as regression, time series analysis, or machine learning techniques, to build predictive models.

**Model Evaluation**: Assessing the performance of predictive models using appropriate evaluation metrics.

**Model Interpretation**: Understanding the factors that drive company registration trends and interpreting model predictions.

**Documentation and Reporting**: Preparing project documentation, reports, and recommendations for stakeholders.

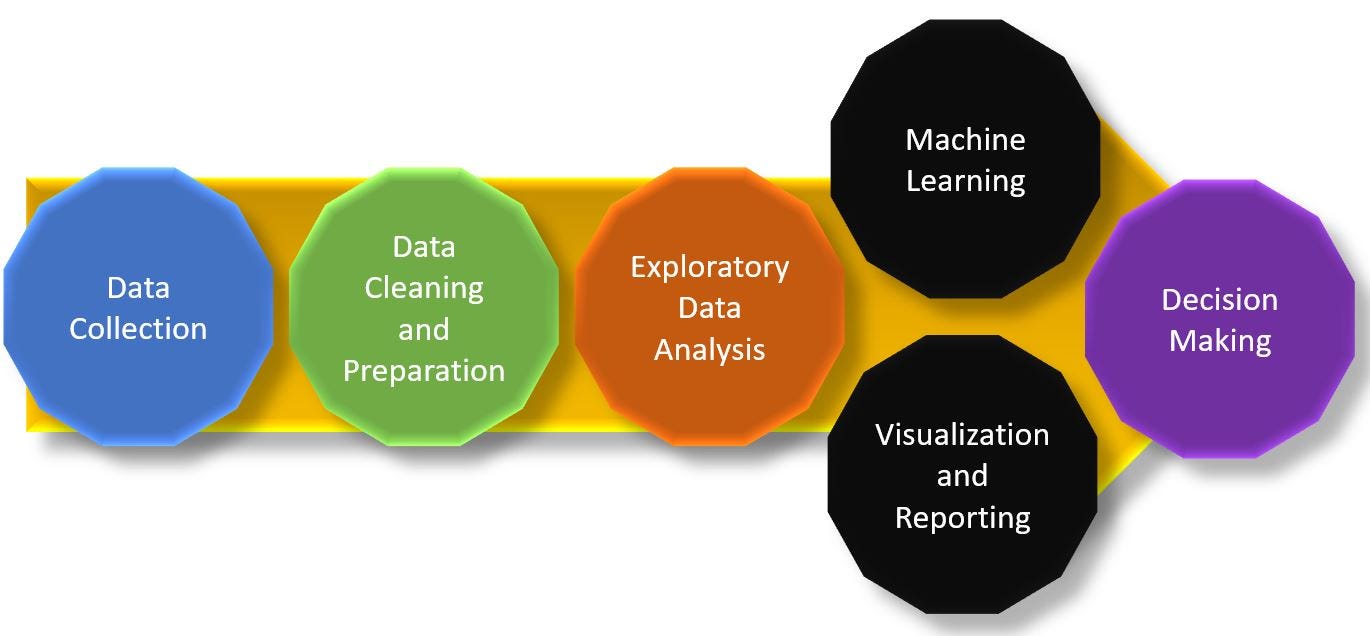
**Dataset Description**



The dataset used in this project is obtained from the Tamil Nadu government and includes master details of companies registered with the Registrar of Companies (RoC) in Tamil Nadu. The dataset comprises the following fields:

Corporate Identification Number (CIN), Company Name, Company Status, Company Class, Company Category, Authorized Capital (in INR), Paid-up Capital (in INR), Date of Registration, Registered State and Registrar of Companies.

**Data Preprocessing Steps**



Data preprocessing is a crucial step to ensure that the dataset is clean, consistent, and ready for analysis and modeling. The following data preprocessing steps were applied to the dataset:

**Data Cleaning**: Identifying and handling missing or erroneous values in the dataset, which may include missing CINs, incorrect capital values, or inconsistent date formats.

**Data Encoding**: Converting categorical variables (e.g., Company Status, Company Class) into numerical representations using techniques like one-hot encoding or label encoding.

**Feature Engineering:** Creating new features that could be useful for analysis or modeling, such as deriving the age of the company based on the registration date.

**Scaling**: Scaling numerical features like Authorized Capital and Paid-up Capital to ensure that they are on a consistent scale for modeling.

**Outlier Detection:** Identifying and addressing outliers in the data that could affect the accuracy of predictive models.

**Data Validation:** Ensuring that the data conforms to the expected format and business rules, and handling any anomalies or inconsistencies.

**Data Integration**: Combining the Tamil Nadu government dataset with additional datasets, such as economic indicators or regional data, if necessary for a more comprehensive analysis.

**Data Splitting**: Splitting the dataset into training and testing subsets for model development and evaluation.

**AI Algorithms Applied**

Various AI algorithms were applied in this project, depending on the specific task, including but not limited to:

Linear Regression

Time Series Analysis (e.g., ARIMA or Prophet)

Machine Learning Algorithms (e.g., Random Forest, XGBoost, or LSTM for time series forecasting)

**Insights Gained from Exploratory Data Analysis**

**During EDA, several valuable insights were gained, including**:

Seasonal trends in company registrations.

Regional variations in registration patterns.

Potential correlations between economic indicators and registration trends.

The impact of external events (e.g., economic crises) on registration.

**Performance of Predictive Models:**

The performance of predictive models was assessed using appropriate metrics, such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), or R-squared (R²), depending on the type of model. Model performance was evaluated against historical data to validate predictive accuracy.

**Compiled Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

#to ignore warnings

import warnings

warnings.filterwarnings('ignore')

data = pd.read\_csv("Data\_Gov\_Tamil\_Nadu.csv")

data.head()

data.tail()

data.info()

data.nunique()

data.isnull().sum()

(data.isnull().sum()/(len(data)))\*100

data = data.drop(['S.No.'], axis = 1)

data.info()

from datetime import date

date.today().year

data['COMPANY\_STATUS']=date.today().year-data['PAIDUP\_CAPITAL']

data.describe().T

cat\_cols=data.select\_dtypes(include=['object']).columns

num\_cols = data.select\_dtypes(include=np.number).columns.tolist()

print("Categorical Variables:")

print(cat\_cols)

print("Numerical Variables:")

print(num\_cols)

for col in num\_cols:

print(col)

print('Skew :', round(data[col].skew(), 2))

plt.figure(figsize = (15, 4))

plt.subplot(1, 2, 1)

data[col].hist(grid=False)

plt.ylabel('count')

plt.subplot(1, 2, 2)

sns.boxplot(x=data[col])

plt.show()

fig, axes = plt.subplots(3, 2, figsize = (18, 18))

fig.suptitle('Bar plot for all categorical variables in the dataset')

sns.countplot(ax = axes[0, 0], x = 'COMPANY\_NAME', data = data, color = 'blue',order = data['COMPANY\_NAME'].value\_counts().index);

sns.countplot(ax = axes[0, 1], x = 'COMPANY\_STATUS', data = data, color = 'blue',order = data['COMPANY\_STATUS'].value\_counts().index);

sns.countplot(ax = axes[1, 0], x = 'PAIDUP\_CAPITAL', data = data, color = 'blue',order = data['PAIDUP\_CAPITAL'].value\_counts().index);

sns.countplot(ax = axes[1, 1], x = 'AUTHORIZED\_CAP', data = data, color = 'blue',order = data['AUTHORIZED\_CAP'].value\_counts().index);

# sns.countplot(ax = axes[2, 0], x = 'PAIDUP\_CAPITAL', data = data, color = 'blue',order = data['Brand'].head(20).value\_counts().index);

# sns.countplot(ax = axes[2, 1], x = 'COMPANY\_STATUS', data = data, color = 'blue', order = data['Model'].head(20).value\_counts().index);

axes[1][1].tick\_params(labelrotation=45);

axes[2][0].tick\_params(labelrotation=90);

axes[2][1].tick\_params(labelrotation=90);

def log\_transform(data,col):

for colname in col:

if (data[colname] == 1.0).all():

data[colname + '\_log'] = np.log(data[colname]+1)

else:

data[colname + '\_log'] = np.log(data[colname])

data.info()

log\_transform(data,['PAIDUP\_CAPITAL','Price'])

plt.figure(figsize=(13,17))

sns.pairplot(data=data.drop(['COMPANY\_STATUS','Price'],axis=1))

plt.show()

fig, axarr = plt.subplots(4, 2, figsize=(12, 18))

data.groupby('Location')['AUTHORIZED\_CAPg'].mean().sort\_values(ascending=False).plot.bar(ax=axarr[0][0], fontsize=12)

axarr[0][0].set\_title("COMPANY\_STATUS", fontsize=18)

data.groupby('Transmission')['Price\_log'].mean().sort\_values(ascending=False).plot.bar(ax=axarr[0][1], fontsize=12)

axarr[0][1].set\_title("COMPANY\_STATUS", fontsize=18)

data.groupby('Fuel\_Type')['Price\_log'].mean().sort\_values(ascending=False).plot.bar(ax=axarr[1][0], fontsize=12)

axarr[1][0].set\_title("Fuel\_Type Vs Price", fontsize=18)

data.groupby('Owner\_Type')['Price\_log'].mean().sort\_values(ascending=False).plot.bar(ax=axarr[1][1], fontsize=12)

axarr[1][1].set\_title("AUTHORIZED\_CAP", fontsize=18)

data.groupby('Brand')['Price\_log'].mean().sort\_values(ascending=False).head(10).plot.bar(ax=axarr[2][0], fontsize=12)

axarr[2][0].set\_title("Brand Vs Price", fontsize=18)

data.groupby('Model')['Price\_log'].mean().sort\_values(ascending=False).head(10).plot.bar(ax=axarr[2][1], fontsize=12)

axarr[2][1].set\_title("AUTHORIZED\_CAP", fontsize=18)

data.groupby('Seats')['Price\_log'].mean().sort\_values(ascending=False).plot.bar(ax=axarr[3][0], fontsize=12)

axarr[3][0].set\_title("Seats Vs Price", fontsize=18)

data.groupby('Car\_Age')['AUTHORIZED\_CAP'].mean().sort\_values(ascending=False).plot.bar(ax=axarr[3][1], fontsize=12)

axarr[3][1].set\_title("AUTHORIZED\_CAP", fontsize=18)

plt.subplots\_adjust(hspace=1.0)

plt.subplots\_adjust(wspace=.5)

sns.despine()

plt.figure(figsize=(12, 7))

sns.heatmap(data.drop(['COMPANY\_STATUS','Price'],axis=1).corr(), annot = True, vmin = -1, vmax = 1)

plt.show()

data.loc[data["PAIDUP\_CAPITAL"]==0.0,'Mileage']=np.nan

data.PAIDUP\_CAPITAL.isnull().sum()

data.COMPANY\_STATUS.isnull().sum()

data['COMPANY\_STATUS'].fillna(value=np.nan,inplace=True)

data['COMPANY\_STATUS']=data.groupby(['AUTHORIZED\_CAP','Brand'])['COMPANY\_STATUS'].apply(lambda x:x.fillna(x.median()))

data['AUTHORIZED\_CAP']=data.groupby(['COMPANY\_STATUS','COMPANY\_STATUS'])['AUTHORIZED\_CAP'].apply(lambda x:x.fillna(x.median()))

data['AUTHORIZED\_CAP']=data.groupby(['COMPANY\_STATUS','AUTHORIZED\_CAP'])['Power'].apply(lambda x:x.fillna(x.median()))