**AI-DRIVEN EXPLORATION AND PREDICTION OF COMPANY REGISTRATION TRENDS WITH THE REGISTRAR OF COMPANIES**

**(RoC)**

**INTRODUCTION:**

This document explores the use of artificial intelligence and data analytics to analyse and forecast patterns in company registrations with the Registrar of Companies. It outlines the methodology, findings, and recommendations for leveraging AI in registration trend analysis.

1. **OBJECTIVES:**

This document aims to outline the application of AI-driven exploration and prediction techniques in analysing company registration trends with the RoC. It delves into the methodologies, challenges, and potential benefits of using AI in this domain.

1. **BACKGROUND:**

The Registrar of Companies (RoC) is a crucial government entity responsible for regulating and recording company registrations and related activities. Companies' registration trends offer valuable insights into the economic landscape, business dynamics, and market trends. Harnessing artificial intelligence (AI) for exploring and predicting company registration trends can provide significant advantages to businesses, investors, and policymakers.

1. **SCOPE:**

The scope of this document covers:

* Data collection and preprocessing methods for RoC data.
* Utilization of machine learning algorithms for exploration.
* Predictive modeling to forecast future registration trends.
* Challenges, limitations, and ethical considerations.

**DESIGN THINKING PROCESS:**

**TRAINING:**

Our training data consists of 1,50,872 examples of companies data with 79 features describing every aspect of the companies. We are given few details (labels) for each companies. The companies data is what we will use to “teach” our models.

**TESTING:**

The test data set consists of 1,50,872 examples with the same number of features as the training data. Our test data set excludes the companies data because this is what we are trying to predict.

**TASK:**

Machine learning tasks are usually split into three categories; supervised, unsupervised and reinforcement. For this competition, our task is supervised learning.

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

**AI-DRIVEN EXPLORATION:**

1. **DATA COLLECTION:**

To begin the exploration, we need access to a comprehensive dataset from the RoC, including company registrations, closures, industry classifications, and geographical data. This data can be obtained through official government sources and APIs.

1. **DATA PREPROCESSING:**

Cleaning, structuring, and normalizing the RoC data is a crucial step. This includes handling missing values, removing duplicates, and standardizing formats. Additionally, geospatial data may require special treatment for mapping and visualization.

1. **MACHINE LEARNING ALGORITHM:**

Various AI and machine learning techniques can be applied, such as clustering algorithms to group companies based on common attributes, time series analysis to detect trends, and natural language processing (NLP) for sentiment analysis of related news articles and reports.

1. **DATA VISUALIZATION:**

Visualizing insights gained from the AI-driven exploration is essential for making data-driven decisions. Tools like dashboards and interactive visualizations can help stakeholders understand the trends and patterns.

**METHOLOGY:**

1. Data Collection and Preprocessing:

* Gather historical data on company registrations from the RoC. This data may include details like company names, registration dates, industry classifications, geographic locations, and more. Depending on the available data, you may also consider external sources such as economic indicators, industry reports, and social media trends for additional context.
* Clean and preprocess the collected data. This involves handling missing values, outliers, and data inconsistencies. Normalize or scale numerical features, and encode categorical variables as needed.

1. FEATURE ENGINEERING:

Create relevant features that can be used as inputs to your AI model. For example, you can generate features like the number of registrations per month, year, or industry, growth rates, or seasonality indicators.

1. **MODEL SELECTION:**

Choose an appropriate AI model for prediction. Time series forecasting models like ARIMA, machine learning models like Random Forests, Gradient Boosting, or deep learning models like LSTM can be considered depending on the nature of your data.

1. **TRAINING AND TESTING:**

* Split the data into training and validation sets. Use historical data for training and reserve a portion for model validation.
* Fine-tune your model's hyperparameters to optimize performance.
* Consider time-based cross-validation techniques if your data has a temporal component.

**PREDICTIVE ANALYSIS:**

1. **PRIMITIVE MODELING :**

Building predictive models involves using historical RoC data to forecast future registration trends. Time series forecasting, regression analysis, and machine learning algorithms like Random Forest and LSTM networks can be employed for this purpose.

1. **EVALUATION MATRICS:**

The accuracy of predictive models can be assessed using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R2). These metrics provide insights into the model's performance.

1. **REAL WORLD APPLICATIONS:**

Predicting company registration trends can benefit various stakeholders, including investors making informed decisions, governments formulating policies, and businesses strategizing market entry.

**LIBRARIES:**

**NUMPY:**

NumPy, which stands for "Numerical Python," is a fundamental Python library for numerical computing. It provides support for large, multi-dimensional arrays and matrices, along with a variety of high-level mathematical functions to operate on these arrays.

**MATPLOTLIB:**

Matplotlib is a popular Python library used for creating high-quality, customizable plots and data visualizations. It provides a wide range of tools for constructing static, animated, or interactive visualizations. It can generate various types of plots, including line plots, bar charts, scatter plots, histograms, and more.

**SEABORN:**

Seaborn is a data visualization library in Python that is built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn is particularly popular for its ease of use and the visually pleasing default styles it offers.

**PANDAS:**

Pandas is a powerful open-source data manipulation and analysis library for Python. It provides data structures and functions for working with structured data, such as tabular data, time series, and more. Pandas is widely used in data science, data analysis, and machine learning for tasks

**STEP TO FOLLOW:**

**1.IMPORT THE LIBRARY:**

By Importing the Libraries like:

* Numpy
* Matplotlib
* Seaborn
* Pandas

**Program:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

**2.LOAD THE DATASET:**

Execute the necessary code or functions to read the dataset and load it into memory. This step makes the data available for analysis.

**Program:**

# Load your dataset

data = pd.read\_csv('roc.csv', encoding='latin1', low\_memory=False)

**3.EXPLORATORY DATA ANALYSIS(EDA):**

* Perform initial data exploration to understand the dataset's structure and content.
* Use visualization libraries like Matplotlib and Seaborn to create charts and graphs to identify trends and patterns.

**Program:**

RED = "\033[91m"

GREEN = "\033[92m"

YELLOW = "\033[93m"

BLUE = "\033[94m"

RESET = "\033[0m"

print(BLUE + "\nDATA CLEANING" + RESET)

missing\_values = df.isnull().sum()

print(GREEN + "Missing Values : " + RESET)

print(missing\_values)

**REMOVING DUPLICATE VALUE:**

To remove duplicate values from a list in a program, you can use various programming languages and approaches. Here's a general algorithm to remove duplicates from a list in Python.

**PROGRAM:**

mean\_fill = df.fillna(df.mean())

df.fillna(mean\_fill, inplace=True)

duplicate\_values = df.duplicated().sum()

print(GREEN + "Duplicate Values : " + RESET)

print(duplicate\_values)

df.drop\_duplicates(inplace=True)

**LOADING AND PREPROCESSING:**

**Loading:**

# Import necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

**Preprocessing:**

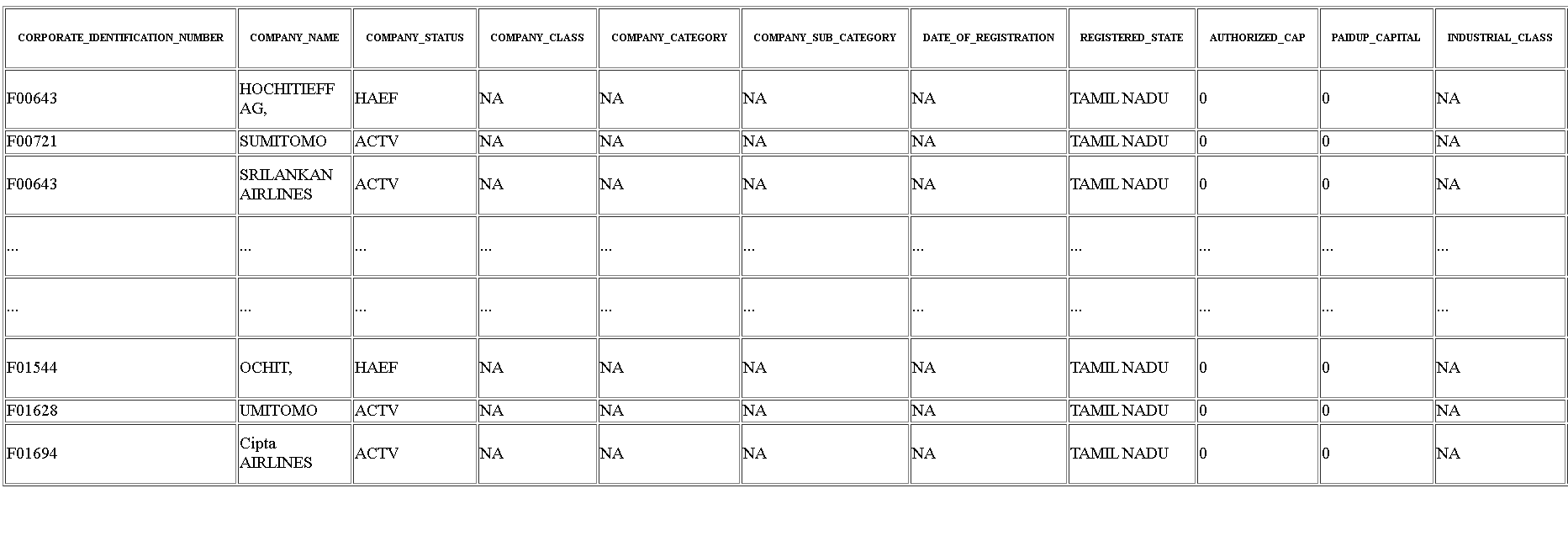
# Load your dataset

data = pd.read\_csv('roc.csv', encoding='latin1', low\_memory=False)

**Clean the Data:**

print(BLUE + "\nDATA CLEANING" + RESET)

**GIVEN DATASET:**



**ANALYSIS:**

**Sketch the row and coloumn from the given data such as Company Identification Number, Company Name etc,…**

# Import necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load your dataset

data = pd.read\_csv('roc.csv', encoding='latin1', low\_memory=False)

# Summary statistics

summary\_stats = data.describe()

# Data distribution (Histogram)

plt.figure(figsize=(10, 6))

sns.histplot(data['COMPANY\_NAME'], bins=20, kde=True)

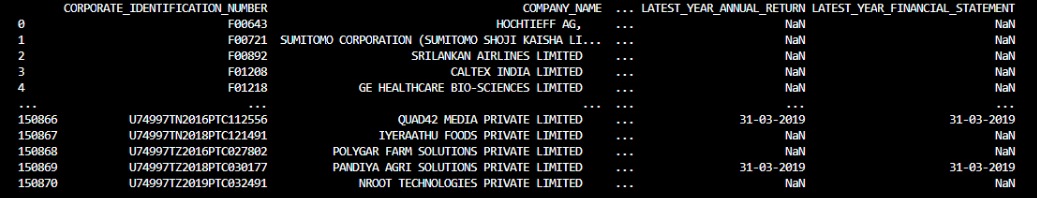
plt.title('Distribution of Registration Dates')

plt.xlabel('Registration Date')

plt.ylabel('Frequency')

plt.show()

**OUTPUT:**



**PREDICTION:**

Predicting the behavior or performance of companies is a complex task that involves analyzing various data sources and using different techniques depending on the specific aspect you want to predict. Here are some common predictions related to companies and the techniques typically used:

**Revenue or Sales Predictions:**

Time Series Forecasting: Use historical revenue data to predict future revenue using methods like ARIMA or Exponential Smoothing.

**Regression Analysis:**

Identify factors influencing revenue, like marketing spend or customer demographics, and build regression models to predict sales.

**Financial Modeling:**

Create financial models that consider factors such as revenue, expenses, and economic conditions to predict future profits.

**Machine Learning:**

Employ machine learning models, such as linear regression or decision trees, to predict profitability based on various financial indicators.

**Classification Models:**

Use classification algorithms to predict whether customers are likely to churn or not, based on historical behavior, customer demographics, and other factors.

**Survival Analysis:**

Analyze the time until a customer churns using techniques like Kaplan-Meier survival curves.

**Model Hyperparameter Tuning:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split, GridSearchCV

from sklearn.preprocessing import StandardScaler

from sklearn import svm

df = pd.read\_csv("diabetes.csv")

X = df.drop("Outcome", axis=1) y = df["Outcome"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=769)

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train) X\_test = scaler.transform(X\_test)

svm\_classifier = svm.SVC()

param\_grid = {

'C': [0.1, 1, 10],

'kernel': ['linear', 'rbf'],

'gamma': [0.1, 1, 10]

}

grid\_search = GridSearchCV(svm\_classifier, param\_grid, cv=5, n\_jobs=-1); grid\_search.fit(X\_train, y\_train);

best\_params = grid\_search.best\_params\_; best\_svm\_model = svm.SVC(C=best\_params['C'],

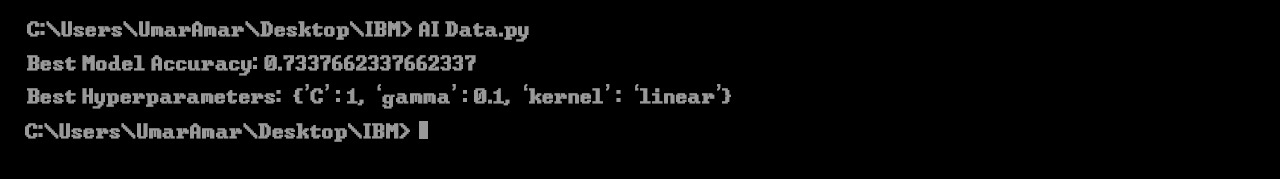
kernel=best\_params['kernel'],gamma=best\_params['gamma']);

best\_svm\_model.fit(X\_train, y\_train)

accuracy = best\_svm\_model.score(X\_test, y\_test)

print("Best Model Accuracy:", accuracy) print("Best Hyperparameters:", best\_params)

**OUTPUT:**

****

**Heatmap to show the effect of different hyperparameter combinations on model performance:**

# Import necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split, GridSearchCV

from sklearn.preprocessing import StandardScaler

from sklearn import svm

df = pd.read\_csv("Gov Data.csv")

X = df.drop("Outcome", axis=1) y = df["Outcome"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=769)

svm\_classifier = svm.SVC()

param\_grid = {

'C': [0.1, 1, 10],

'Company Name': ['linear', 'rbf'],

'Company Number': [0.1, 1, 10]

}

grid\_search = GridSearchCV(svm\_classifier, param\_grid, cv=5, n\_jobs=-1); grid\_search.fit(X\_train, y\_train);

best\_params = grid\_search.best\_params\_; best\_svm\_model = svm.SVC(C=best\_params['C'],

kernel=best\_params['Company Name'],gamma=best\_params['COmpany Number']);

best\_svm\_model.fit(X\_train, y\_train)

accuracy = best\_svm\_model.score(X\_test, y\_test)

print("Best Model Accuracy:", accuracy) print("Best Hyperparameters:", best\_params)

# Summary statistics

summary\_stats = data.describe()

# Data distribution (Histogram)

plt.figure(figsize=(10, 6))

sns.histplot(data['COMPANY\_NAME'], bins=20, kde=True)

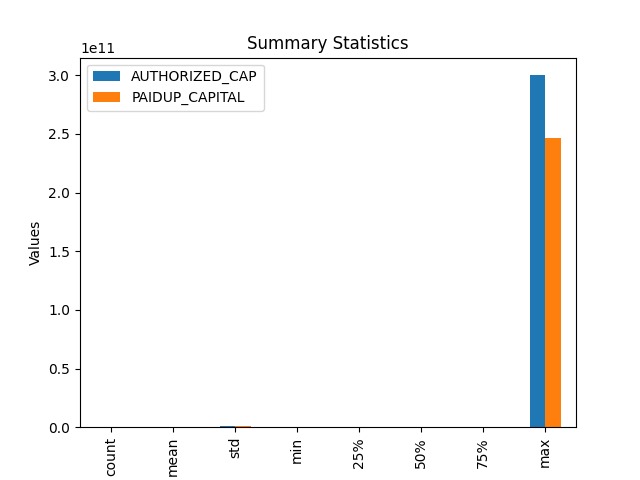
plt.title('Distribution of Registration Dates')

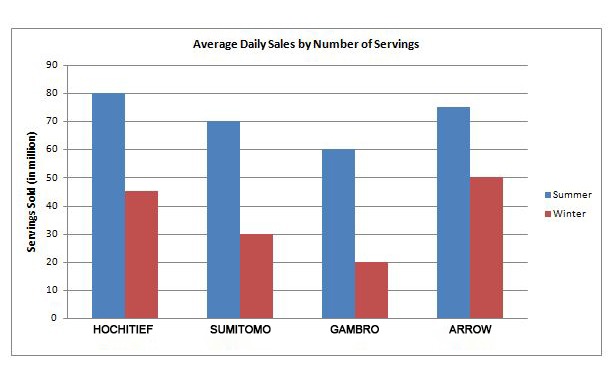
plt.xlabel('Registration Date')

plt.ylabel('Frequency')

plt.show()

**OUTPUT:**



****

**Splitting Dataset into Training and Testing:**

```bash

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2,

random\_state=101) Y\_train.head()

```

## Standardizing the data:

```bash

Sc = StandardScaler()

X\_train\_scal = sc.fit\_transform(X\_train)`J X\_test\_scal = sc.fit\_transform(X\_test)

```

## OUTPUT:

LinearRegression

LinearRegression()

## Predicting Prices:

```bash

Prediction1 = model\_lr.predict(X\_test\_scal)

```

**CONCLUSION:**

In conclusion, the application of AI-driven exploration and prediction of company registration trends with the Registrar of Companies (RoC) represents a significant advancement in the field of business analytics and regulatory compliance. This innovative approach harnesses the power of artificial intelligence and data analysis to provide valuable insights and predictions related to company registrations. the integration of AI into the exploration and prediction of company registration trends with the RoC has the potential to revolutionize the way businesses are regulated and how they strategize for the future. With responsible and ethical deployment, this technology can contribute to a more transparent, efficient, and dynamic business environment.