UNIVERSITY COLLEGE OF ENGINEERING VILLUPURAM

NAAN MUDHALVAN

IBM – ARTIFICIAL INTELLIGENCE

HariPriya R

422521104012

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

PROBLEM DEFINITION:

The problem is to perform an AI-driven exploration and predictive analysis on the master details of companies registered with the Registrar of Companies (RoC). The objective is to uncover hidden patterns, gain insights into the company landscape, and forecast future registration trends. This project aims to develop predictive models using advanced Artificial Intelligence techniques to anticipate future company registrations and support informed decision-making for businesses, investors, and policymakers.

DESIGN THINKING:

1. Data Source:

- Dataset: You will start with a dataset containing information about registered companies. This dataset should include various columns, such as company name, status (active, dissolved, etc.), class (private, public, etc.), category (industry type), registration date, authorized capital, paid-up capital, and potentially more relevant attributes.

2. Data Preprocessing:

- Cleaning Data: This involves removing or correcting any data errors or inconsistencies, such as typos or outliers.

- Handling Missing Values: Address missing data points by either imputing values or deciding how to handle them appropriately.

- Categorical to Numerical Conversion: Convert categorical features like company status, class, and category into numerical representations using techniques like one-hot encoding or label encoding.

3. Exploratory Data Analysis (EDA):

- Distribution Analysis: Visualize and analyze the distribution of various attributes, overall characteristics.

- Relationship Exploration: Identify correlations and relationships between different features. For instance, you might explore how registration date relates to company status.

- Unique Characteristics: Discover unique patterns or characteristics in the data that could be valuable for predictive modeling.

4. Feature Engineering:

- Create Relevant Features: Develop new features or transformations of existing ones that may be informative for predicting future company registrations. For example, you could create a feature that calculates the age of each company since its registration date.

5. Predictive Modeling:

- Algorithm Selection: Choose appropriate machine learning or AI algorithms for your predictive modeling task. Common choices include regression, classification, or time series forecasting algorithms, depending on your specific goals.

- Training and Testing: Split your data into training and testing sets.

Train your model on the training data and evaluate its performance on the testing data.

- Hyperparameter Tuning: Optimize the model's hyperparameters to improve its predictive accuracy.

- Cross-Validation: Implement cross-validation techniques to ensure your model generalizes well to unseen data.

6. Model Evaluation:

- Metrics: Evaluate the predictive models using relevant metrics such as accuracy, precision, recall, F1-score, or ROC AUC, depending on the nature of your predictive task (e.g., classification or regression).

- Visualization: Visualize model performance using plots like ROC curves or confusion matrices.

- Model Comparison: Compare different models to select the one that best meets your project's objectives.

SAMPLE PROGRAM:

```
import pandas as pd

# Load the dataset

df = pd.read_csv('company_data.csv')

import matplotlib.pyplot as plt

import seaborn as sns

# Example EDA - histogram of authorized capital

plt.hist(df['authorized_capital'], bins=20)

plt.xlabel('Authorized Capital')

plt.ylabel('Frequency')

plt.title('Distribution of Authorized Capital')

plt.show()

from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split

# Split data 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)

# Create and train the model

model = RandomForestClassifier()

model.fit(X_train, y_train)
```

```
from sklearn.metrics import accuracy_score,
precision_score, recall_score, f1_score

# Make predictions

y_pred = model.predict(X_test)

# Evaluate the model

accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred)

recall = recall_score(y_test, y_pred)

f1 = f1_score(y_test, y_pred)
```