Phase 4 Submission Document

AI-Driven Exploration and Prediction of Company Registration Trends With Registrar of Company(ROC)

510521205032:Rasmi S

Phase 4:Development Part 2

Project Title:ROC company Analysis

Development part 2:Continue building the AI-driven exploration and prediction project by performing exploratory data analysis,feature engineering ,and predictive modeling.

INTRODUCTION:

In today's data-driven world, harnessing the power of artificial intelligence (AI) has become essential for extracting valuable insights and making informed decisions. This project aims to leverage AI techniques to perform data exploration, feature engineering, and predictive modeling. By doing so, we can uncover hidden patterns in data, create meaningful features, and make accurate predictions, ultimately leading to informed decision-making in various domains.

1. Exploratory Data Analysis (EDA):

Exploratory Data Analysis is the first crucial step in our journey. EDA involves understanding the dataset's characteristics, uncovering patterns, and identifying relationships within the data. Key components of EDA include data cleaning, summary statistics, and data visualization. By examining the data in depth, we gain insights that pave the way for effective predictive modeling. EDA allows us to:

- Identify missing values and outliers, ensuring data quality.
- Summarize data through statistical measures.
- Visualize data using various charts and plots to understand its distribution and relationships.
- Recognize patterns and trends that can guide feature engineering and modeling.

2. Feature Engineering:

Feature engineering is the process of creating or transforming features (variables) in the dataset to enhance model performance. Effective feature engineering can significantly impact the predictive power of our models. In this phase, we will:

- Select relevant features and discard irrelevant ones.
- Create new features that capture domain-specific information.
- Handle categorical variables through one-hot encoding or other suitable techniques.

- Normalize or scale numerical features for improved model convergence.
- Prepare the data for modeling by addressing any data-specific challenges.

3. Predictive Modeling:

With the data well-prepared, we will venture into predictive modeling. Predictive modeling is the core of our project, where we utilize various machine learning and AI algorithms to make informed predictions. This phase involves:

- Data splitting: Separating the dataset into training, validation, and test sets.
- Model selection: Choosing the most suitable algorithm for the specific prediction task.
- Model training: Training the selected model using the training data.
- Model evaluation: Assessing the model's performance using various metrics.
- Hyperparameter tuning: Optimizing the model's parameters to enhance performance.
- Model validation: Testing the model on unseen data to ensure generalizability.
- Interpretability: Employing techniques to understand how the model makes predictions, ensuring transparency.

Ultimately, the AI-driven exploration and prediction project combines data analysis, feature engineering, and predictive modeling to empower organizations and individuals to make data-informed decisions. Whether it's predicting sales, identifying anomalies, or forecasting trends, the project's outcomes will be instrumental in a wide range of applications, fostering better decision-making and a deeper understanding of complex data.

GIVEN DATASET:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	CORPORATE_	COMPANY_NA	COMPANY_ST	COMPANY_CL	COMPANY_CA	COMPANY_SU	DATE_OF_RE	REGISTERED_	AUTHORIZED_	PAIDUP_CAPI	INDUSTRIAL_(	PRINCIPAL_B	REGISTERED_	REGISTRAR_(	EMAIL_ADDR	LATEST_YEAR	LATEST_YEAR_FINANCIAL_STATEMENT		
2	F00643	HOCHTIEFF AC	NAEF	NA	NA	NA	1/12/1961	Tamil Nadu	0	0	NA	Agriculture & a	AMBLE SIDE, N	ROC[illegible]ELHI	NA	NA	NA		
3	F00721	SUMITOMO CO	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	FLAT NO. 6, Ist	ROC[illegible]ELHI	shuchi.chug@a	NA	NA		
4	F00892	SRILANKAN AI	ACTV	NA	NA	NA	1/3/1982	Tamil Nadu	0	0	NA	Agriculture & a	SRILANKAN AIR	ROC[illegible]ELHI	shree16us@ya	NA	NA		
5	F01208	CALTEX INDIA	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	GOLD CREST 2	ROC[illegible]ELHI	NA	NA	NA		
6	F01218	GE HEALTHCA	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	FF-3 Palani C	ROC[illegible]ELHI	karthick9999@	NA	NA		
7	F01265	CAIRN ENERG	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	WELLINGTON	ROC[illegible]ELHI	neerja.sharma	NA	NA		
8	F01269	TORIELLI S.R.L	ACTV	NA	NA	NA	5/9/1995	Tamil Nadu	0	0	NA	Agriculture & a	6, Mangayark	ROC[illegible]ELHI	chennai@torie	NA	NA		
9	F01311	HARDY EXPLC	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	5TH FLOOR, WI	ROC[illegible]ELHI	venkatesh.v@	NA	NA		
10	F01314	HOCHTIOF AK	ACTV	NA	NA	NA	11/4/1996	Tamil Nadu	0	0	NA	Agriculture & a	NEW NO.86, O	ROC[illegible]ELHI	kumar@intern	NA	NA		
11	F01412	EPSON SINGAF	ACTV	NA	NA	NA	25-04-1997	Tamil Nadu	0	0	NA	Agriculture & a	7C CEATURY I	ROC[illegible]ELHI	NA	NA	NA		
12	F01426	CARGOLUX AI	ACTV	NA	NA	NA	11/6/1997	Tamil Nadu	0	0	NA	Agriculture & a	OFFICE NO 9IN	ROC[illegible]ELHI	NA	NA	NA		
13	F01468	CHO HEUNG EI	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	129, MANPUR '	ROC[illegible]ELHI	chowelaccount	NA	NA		
14	F01543	NYCOMED ASI	ACTV	NA	NA	NA	27-10-1998	Tamil Nadu	0	0	NA	Agriculture & a	A D 46 IST ST	ROC[illegible]ELHI	NA	NA	NA		
15	F01544	CHERRINGTON	ACTV	NA	NA	NA	1/5/2000	Tamil Nadu	0	0	NA	Agriculture & a	10HADDOWS R	ROC[illegible]ELHI	NA	NA	NA		
16	F01563	SHIMADZU ASI	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	FIRST FLOOR,	ROC[illegible]ELHI	kousik@vsnl.c	NA	NA		
17	F01565	CORK INTERN	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	ARJAY APEX C	ROC[illegible]ELHI	NA	NA	NA		
18	F01566	ERBIS ENGG C	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	39,2nd Main R	ROC[illegible]ELHI	NA	NA	NA		
19	F01589	RALF SCHNEID	NAEF	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	FLAT C, 'SAI V	ROC[illegible]ELHI	NA	NA	NA		
20	F01593	MITRAJAYA TR	ACTV	NA	NA	NA	NA	Tamil Nadu	0	0	NA	Agriculture & a	OLD NO 148 N	ROC[illegible]ELHI	NA	NA	NA		

1992170 Rows&17 Columns

OVERVIEW OF THE PROCESS:

Building an AI-driven exploration and prediction project involves a systematic process that encompasses exploratory data analysis (EDA), feature engineering, and predictive modeling. This project aims to extract valuable insights from data, create meaningful features, and make accurate predictions. Below is an overview of the key steps involved in this process:

1. Data Collection and Understanding:

The project begins with the collection of relevant data. This data could come from various sources, such as databases, APIs, or external datasets. Understanding the data is essential, including its format, structure, and context. This stage may also involve data preprocessing to handle missing values, outliers, and data quality issues.

```
import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import missingno as msno

import plotly.express as px

import plotly.graph_objs as go

import os

for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
data=pd.read_csv("/kaggle/input/all-indian-companies-registration-data-1900-2019/registered_companies.csv")
```

```
In[1]: print(f"There are {data.shape[0]} rows and {data.shape[1]} columns in the data.")
Out[1]: There are 1992170 rows and 17 columns in the data.
In[2]: print(f"Total Values : {len(df)}\n")
       for x in df.columns:
       print(f'{len(df)-df[x].count()} values missing in {x}')
```

```
Out[2]:Total Values : 1992170


0 values missing in CORPORATE_IDENTIFICATION_NUMBER
0 values missing in COMPANY_NAME
0 values missing in COMPANY_STATUS
5078 values missing in COMPANY_CLASS
5085 values missing in COMPANY_CATEGORY
5090 values missing in COMPANY_SUB_CATEGORY
2525 values missing in DATE_OF_REGISTRATION
0 values missing in REGISTERED_STATE
0 values missing in AUTHORIZED_CAP
0 values missing in PAIDUP_CAPITAL
4811 values missing in INDUSTRIAL_CLASS
12 values missing in PRINCIPAL_BUSINESS_ACTIVITY_AS_PER_CIN
15259 values missing in REGISTERED_OFFICE_ADDRESS
42198 values missing in REGISTRAR_OF_COMPANIES
370208 values missing in EMAIL_ADDR
831317 values missing in LATEST_YEAR_ANNUAL_RETURN
828829 values missing in LATEST_YEAR_FINANCIAL_STATEMENT
```

2. Exploratory Data Analysis (EDA):

EDA is a critical phase that involves gaining a deep understanding of the dataset. Key steps in EDA include:

- Data Cleaning: Identifying and handling missing values, duplicates, and inconsistencies.
- Summary Statistics: Calculating and visualizing statistical measures like mean, median, and standard deviation.
- Data Visualization: Creating visualizations (e.g., histograms, scatter plots) to reveal data patterns and relationships.
- Feature Correlation Analysis: Assessing the relationships between features and identifying correlations.
- Domain-Specific Insights: Leveraging domain knowledge to uncover patterns and trends within the data.

Missing Values Pattern:

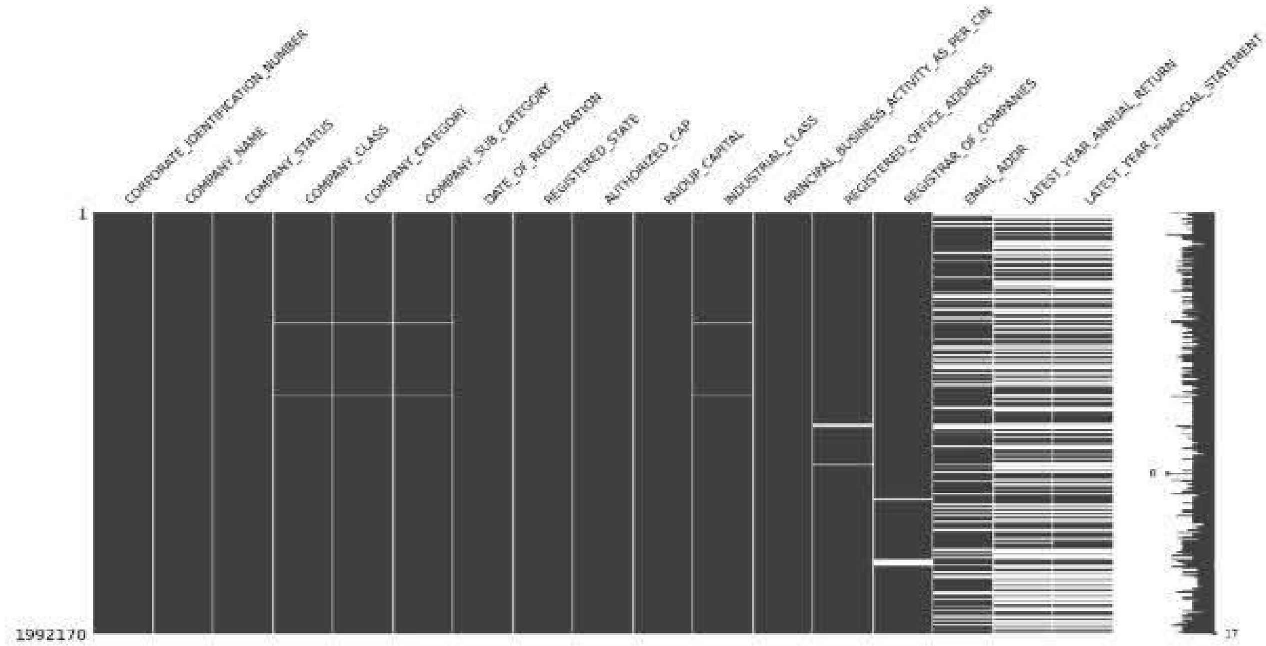
We got an elegant library missingno to vizualize the missing data.The white line in the below matrix shows a missing/NA/None value in a column.

In [3]:

```
msno.matrix(data)
```

Out[3]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f0b974f0e10>DD



Data Visualization:

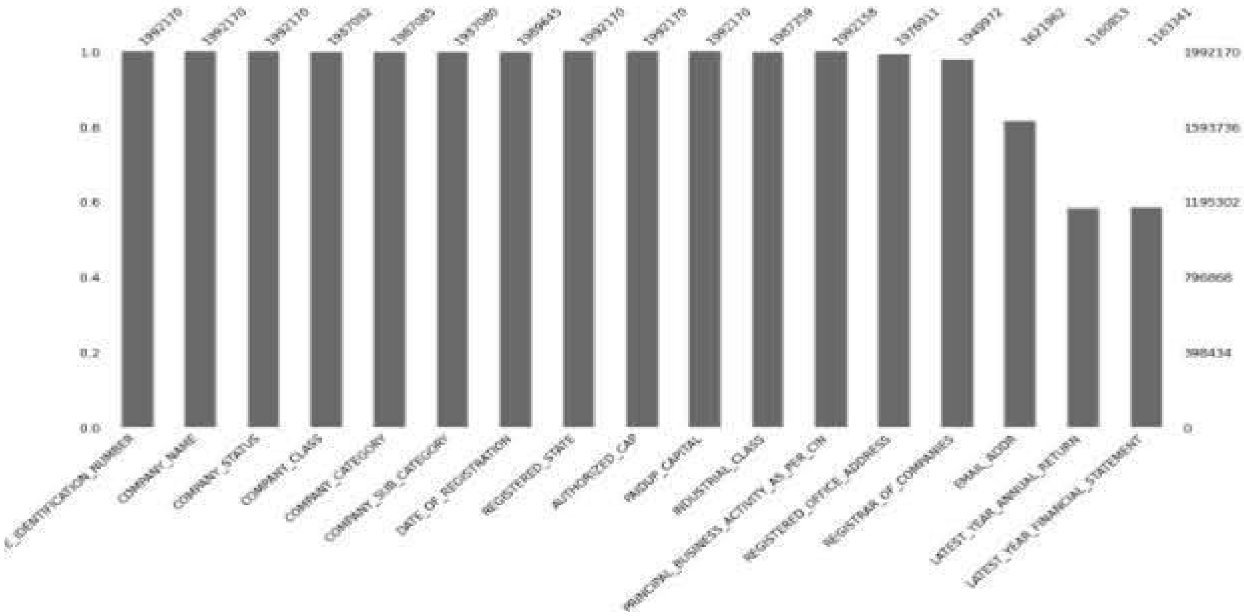
The bar graph makes more sense in understanding the exact numbers of missing value in our dataset.

In[4]:

```
msno.bar(data)
```

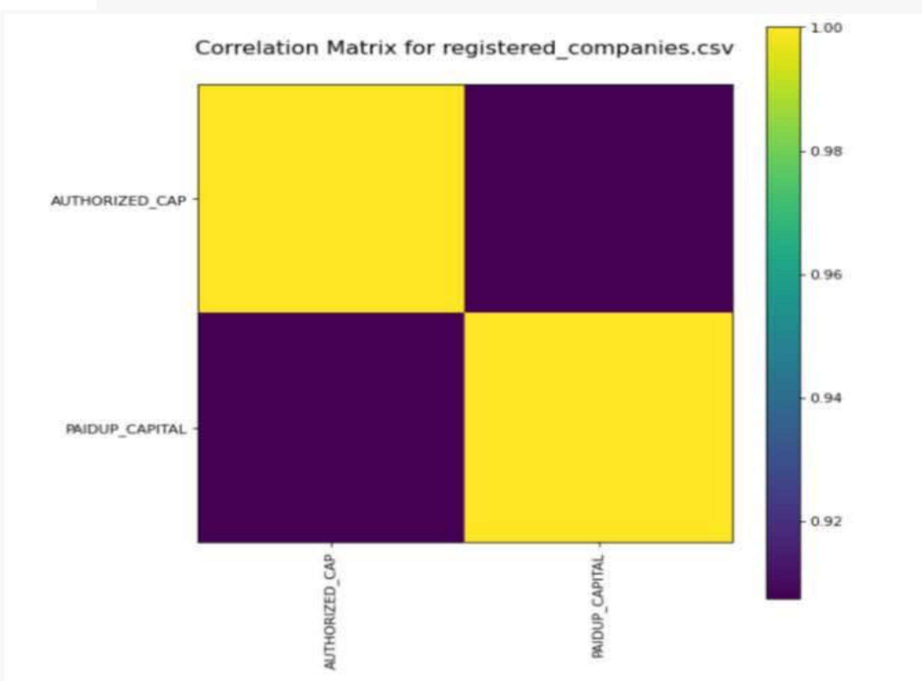
Out[4]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f0b67b8d750>



Feature Correlation Analysis:

```
In[5]: plotCorrelationMatrix(df1, 8)
Out[5]:
```



Visualization:

In [6]:

```
df2 = df
df = df.sample(n=10000)
```

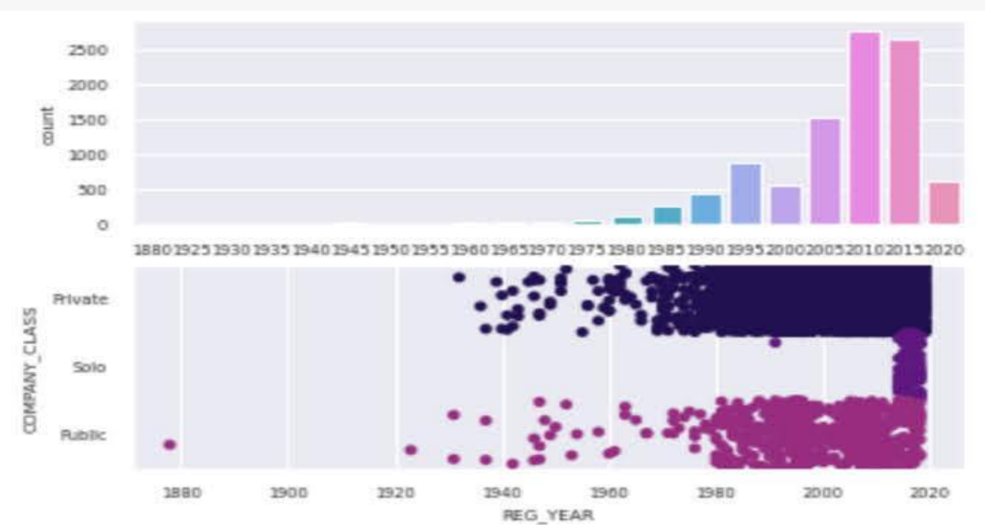
In [7]:

```
sns.set_theme(context='notebook', style='darkgrid',
palette='magma',font='sans-serif',font_scale=0.6,
color_codes=True,rc=None)
```

In [8]:

```
f, ax = plt.subplots(2)
#Counting all the number fo companies by REG_YEAR
sns.countplot(x="REG_YEAR_5BIN",data=df, ax = ax[0])
#Year of registration by COMPANY_CLASS

sns.stripplot(x="REG_YEAR",y="COMPANY_CLASS",data=df, jitter=0.5,ax=
ax[1])
```

Out[8]:<AxesSubplot:xlabel='REG_YEAR', ylabel='COMPANY_CLASS'>

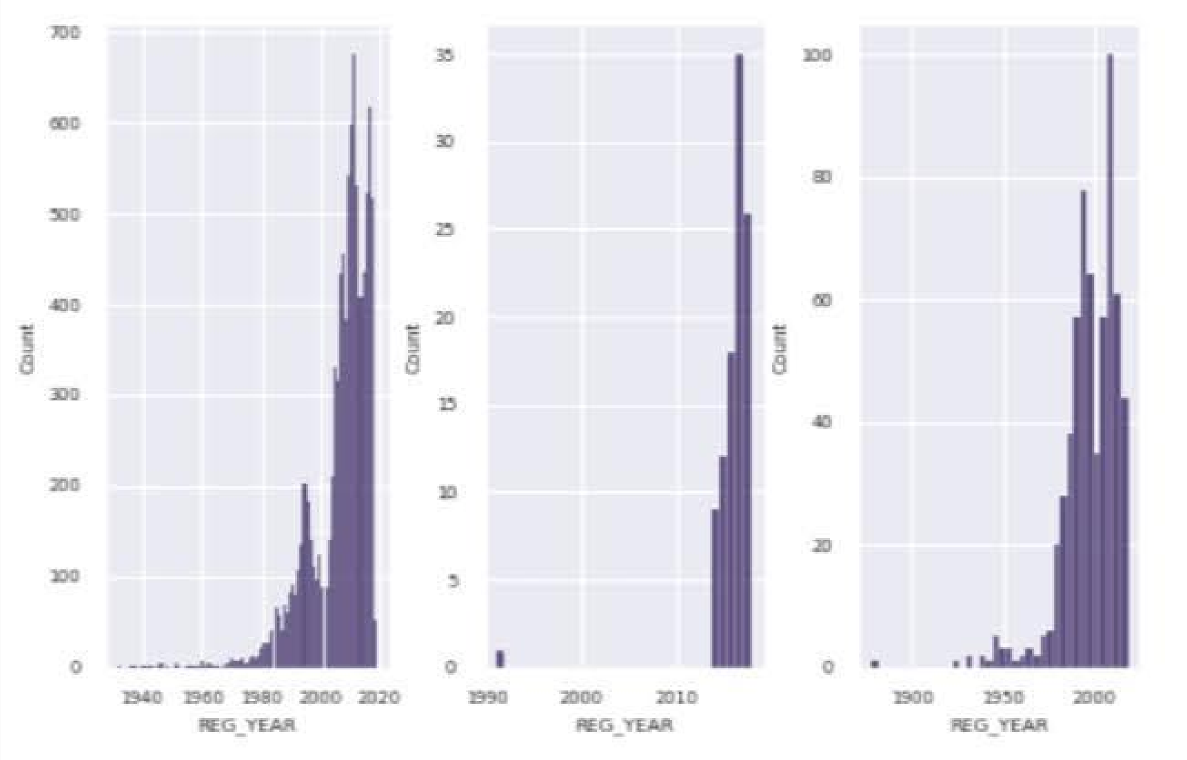


In[9]:

```
f, ax = plt.subplots(1, len(df["COMPANY_CLASS"].unique()))
f.tight_layout()
y=0
print(df["COMPANY_CLASS"].unique())
for x in df["COMPANY_CLASS"].unique():
    sns.histplot(x="REG_YEAR",
                data=df[df["COMPANY_CLASS"]==x],
                ax=ax[y])
    y+=1
```

Out[9]:

```
['Private' 'Solo' 'Public']
```

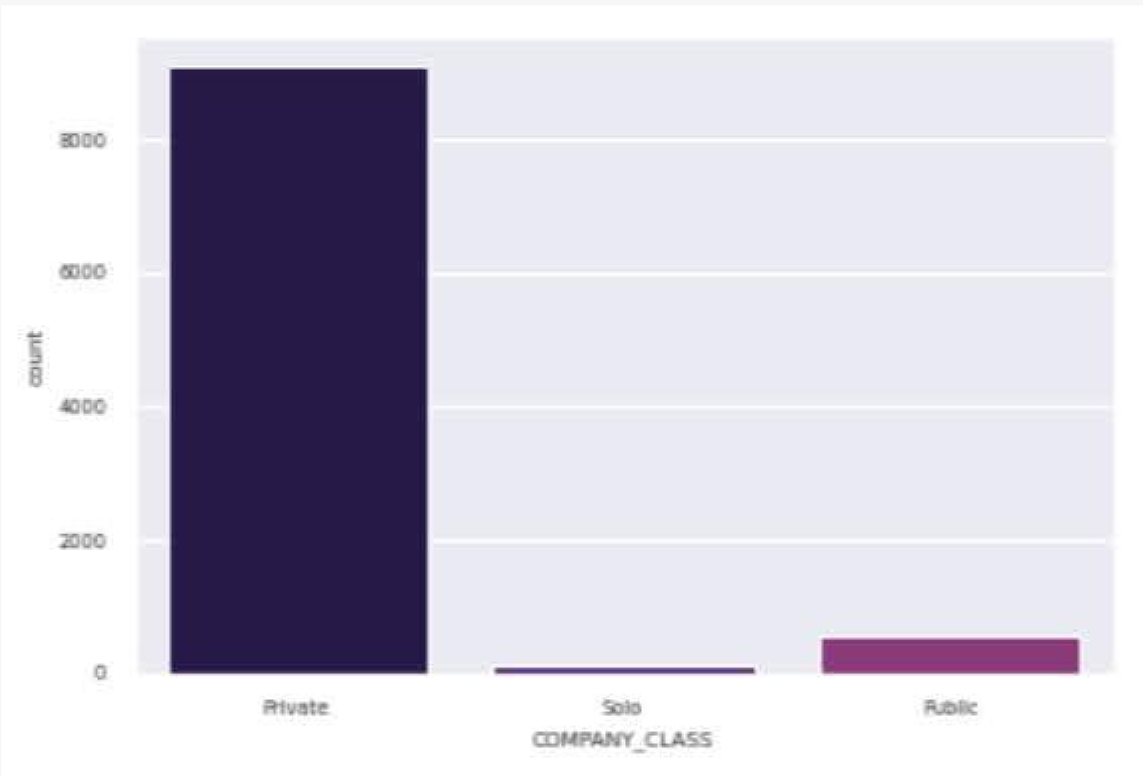


India is seeing a increasing number of new companies being registered which a vast proportion of them being in the 2000s+ \ Solo COMPANY_CLASS catches traction post 2010+ from the first lower plot and can see appearing from 2014 in this sample space on the second rightmost plot.\ The majority of companies are classified as Private (density of the first lower plot & count in second plot).

In[10]:

```
sns.countplot(x="COMPANY_CLASS",
              data=df[df["REG_YEAR"] >= 1982])
```

Out[10]:<AxesSubplot:xlabel='COMPANY_CLASS', ylabel='count'>



In[11]:

```
sns.stripplot(x="REG_YEAR",
              y="PRINCIPAL_BUSINESS",
              hue="COMPANY_CLASS",
              data=df, jitter=0.3)
```

Out[18]:

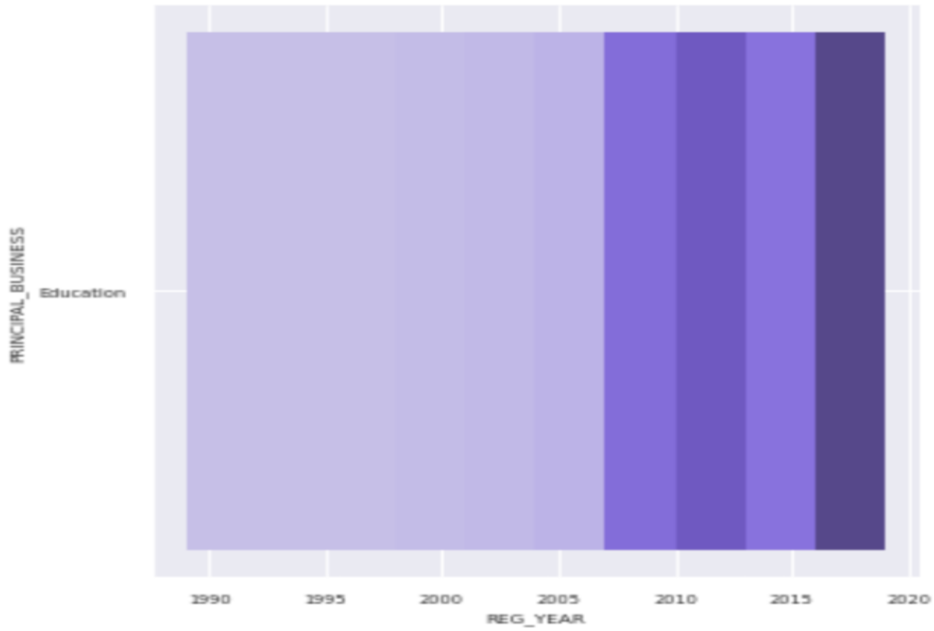
<AxesSubplot:xlabel='REG_YEAR', ylabel='PRINCIPAL_BUSINESS'>

Int[12]:

```
sns.displot(x="REG_YEAR", y="PRINCIPAL_BUSINESS",
data=df[df["PRINCIPAL_BUSINESS"] == "Education"])
```

Out[12]:

<seaborn.axisgrid.FacetGrid at 0x7f4abaaa1890>

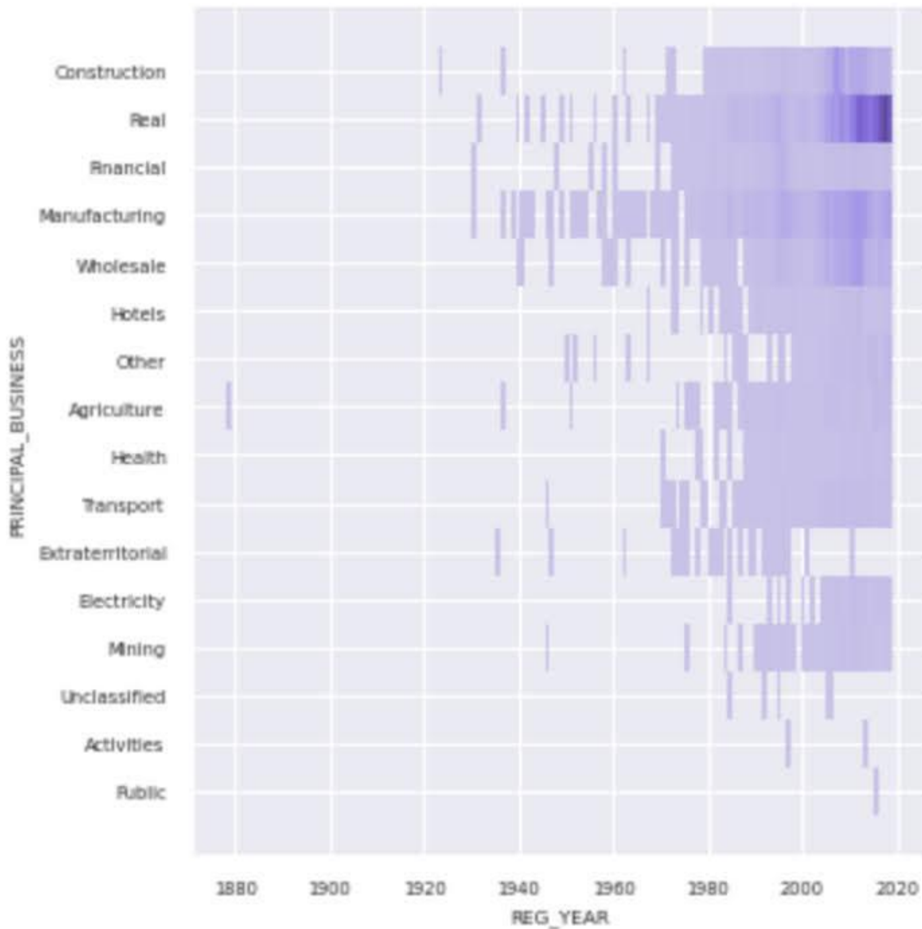


In[13]:

```
sns.displot(x="REG_YEAR", y="PRINCIPAL_BUSINESS",
data=df[df["PRINCIPAL_BUSINESS"] != "Education"])
```

Out[13]:

<seaborn.axisgrid.FacetGrid at 0x7f4abab1a710>



In[13]:

```
df["AUTHORIZED_CAP"].describe()
```

Out[27]:

```
count     1.000000e+04
mean      7.809848e+07
```

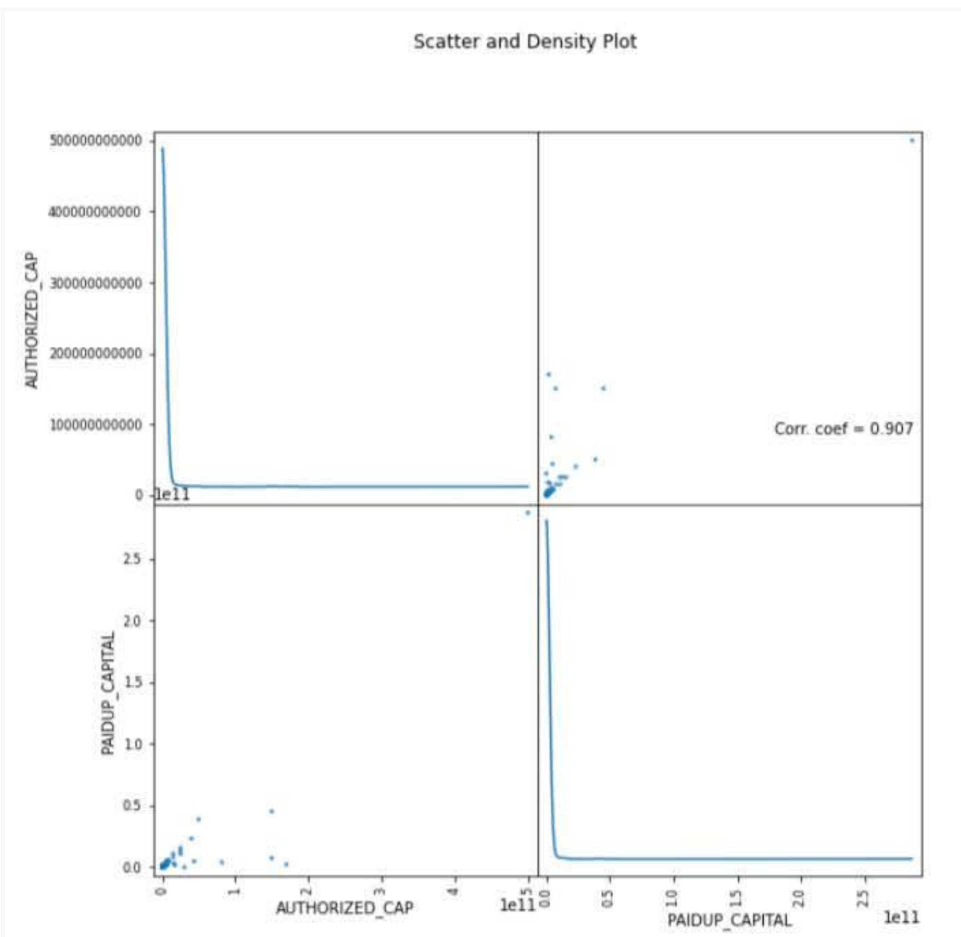
```
std      2.047122e+09
min      0.000000e+00
25%      1.000000e+05
50%      1.000000e+06
75%      4.000000e+06
max      1.500000e+11

Name: AUTHORIZED_CAP, dtype: float64
```

Scatter plot:

`In[14]:`

```
plotScatterMatrix(df1, 9, 10)
```

`Out[14]:`



3. Feature Engineering:

Effective feature engineering is the art of creating or transforming features to improve predictive model performance. This phase includes:

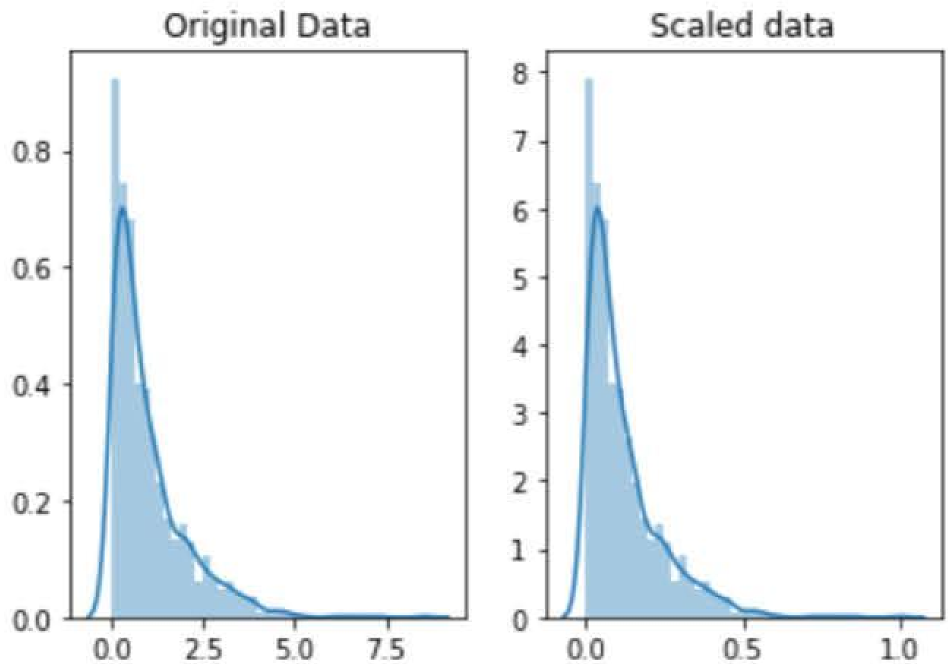
- Feature Selection: Identifying the most relevant features and eliminating irrelevant ones.
- Feature Creation: Crafting new features based on domain knowledge or relationships between existing features.
- Handling Categorical Data: Converting categorical variables into a numerical format, often through one-hot encoding.
- Normalization and Scaling: Preprocessing numerical features to bring them to a common scale.
- Data Transformation: Applying mathematical transformations to better represent the underlying relationships.

Scaling and Normalization:

This means that you're transforming your data so that it fits within a specific scale, like 0-100 or 0-1. You want to scale data when you're using methods based on measures of how far apart data points, like support vector machines, or SVM or k-nearest neighbors, or KNN. With these algorithms, a change of "1" in any numeric feature is given the same importance.

```
In[15]:

# generate 1000 data points randomly drawn from an exponential distribution

original_data = np.random.exponential(size = 1000)

# mix-max scale the data between 0 and 1

scaled_data = minmax_scaling(original_data, columns = [0])

# plot both together to compare

fig, ax=plt.subplots(1,2)

sns.distplot(original_data, ax=ax[0])

ax[0].set_title("Original Data")

sns.distplot(scaled_data, ax=ax[1])

ax[1].set_title("Scaled data")

Out[15]:
```

- Text(0.5,1,'Scaled data')



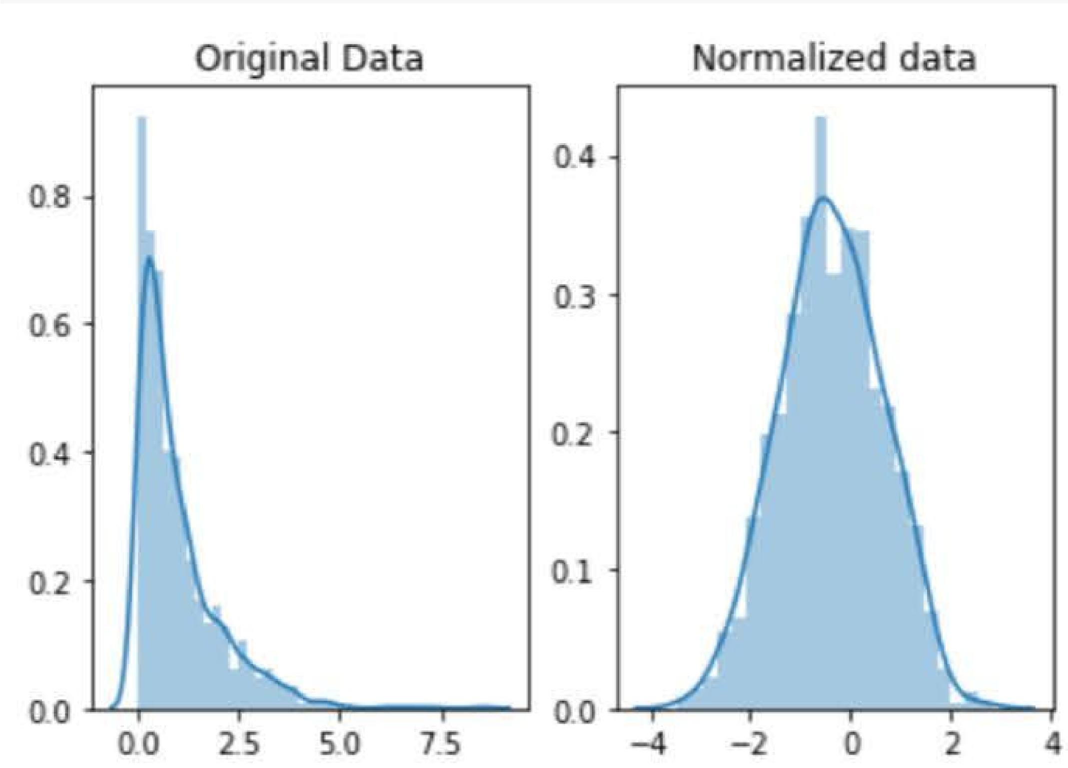
Normalization is a more radical transformation. The point of normalization is to change your observations so that they can be described as a normal distribution.

```
In[16]:

# normalize the exponential data with boxcox

normalized_data = stats.boxcox(original_data)

# plot both together to compare

fig, ax=plt.subplots(1,2)

sns.distplot(original_data, ax=ax[0])

ax[0].set_title("Original Data")
```

```
sns.distplot(normalized_data[0], ax=ax[1])

ax[1].set_title("Normalized data")
```

Out[3]:

```
Text(0.5,1,'Normalized data')
```



RandomForestRegressor:

```
# create instance

forest = RandomForestRegressor(n_estimators=100,

random_state=10)
```

```
params = {"max_depth":[20,21,22,23], "n_estimators":[39, 41,
43, 45]}# Fitting

cv_f = GridSearchCV(forest, params, cv = 10,
n_jobs=10)cv_f.fit(X_train, y_train)

print("Best params:{}".format(cv_f.best_params_))

best_f = cv_f.best_estimator_

prediction

y_tra# in_pred_f = best_f.predict(X_train)

y_val_pred_f = best_f.predict(X_val)

print("MSE train:{}".format(mean_squared_error(y_train,
y_train_pred_f)))

print("MSE test;{}".format(mean_squared_error(y_val,
y_val_pred_f)))

print("R2 score train:{}".format(r2_score(y_train,
y_train_pred_f)))

print("R2 score test:{}".format(r2_score(y_val,
y_val_pred_f)))

Out[16]Best params:{'max_depth': 23, 'n_estimators': 45}
```

```
MSE train:0.0005782719112940582

MSE test;0.004247373155393951

R2 score train:0.9802428621873917

R2 score test:0.8721541441404171
```

4. Predictive Modeling:

The heart of the project is predictive modeling, where the focus shifts to building machine learning or AI models to make predictions. The steps in this phase include:

- Data Splitting: Dividing the dataset into training, validation, and test sets to evaluate model performance.
- Model Selection: Choosing the appropriate algorithm(s) for the specific prediction task, considering factors such as data type and complexity.
- Model Training: Training the selected model using the training dataset and optimizing its parameters.
- Model Evaluation: Assessing model performance using relevant evaluation metrics (e.g., accuracy, F1 score, RMSE).
- Hyperparameter Tuning: Fine-tuning the model by adjusting hyperparameters to optimize performance.

- Interpretability and Explainability: Employing techniques to understand how the model makes predictions, making results more transparent.
- Model Validation: Validating the model on the test dataset to ensure it generalizes well to unseen data.

```
from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout

from tensorflow.keras.callbacks import EarlyStopping

In [16]:

model = Sequential()

model.add(Dense(units= 7, activation='relu'))

#model.add(Dropout(0.5))

#model.add(Dense(units= 14, activation='relu'))

#model.add(Dropout(0.5))

model.add(Dense(units= 3, activation='relu'))

model.add(Dense(units=1,kernel_initializer='uniform',
activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

```
#early_stopping = EarlyStopping(monitor='val_loss', mode='min',
verbose=1, patience=40)

model.fit(x=X_train_scaled_2, y=y_train_scaled_2, epochs=1000,
batch_size=200,
```

```
validation_data=(val_test_X, val_test_y)) #callbacks=early_stopping)
```

Predictive modeling:

In [17]:

```
models = pd.DataFrame(columns=["Model","MAE","MSE","RMSE","R2 S
core","RMSE (Cross-Validation)"])
```

Linear Regression:

In [18]:

```
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)
predictions = lin_reg.predict(X_test)
mae, mse, rmse, r_squared = evaluation(y_test, predictions)
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
```

```
print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(lin_reg)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "LinearRegression","MAE": mae, "MSE": mse, "RM

SE": rmse, "R2 Score": r_squared, "RMSE (Cross-Validation)": rmse_cross

_val}models = models.append(new_row, ignore_index=True)
```

Out[18]:

```
MAE: 23567.890565943395

MSE: 1414931404.6297863

RMSE: 37615.57396384889

R2 Score: 0.8155317822983865

------------------------------

RMSE Cross-Validation: 36326.451444669496
```

Ridge Regression:

In [19]:

```
ridge = Ridge()ridge.fit(X_train, y_train)predictions = ridge.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)
```

```
print("MAE:", mae)

print("MSE:", mse)

print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(ridge)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "Ridge","MAE": mae, "MSE": mse, "RMSE": rmse,

"R2 Score": r_squared, "RMSE (Cross-Validation)": rmse_cross_val}model

s = models.append(new_row, ignore_index=True)

Out[19]:

MAE: 23435.50371200822

MSE: 1404264216.8595588

RMSE: 37473.513537691644

R2 Score: 0.8169224907874508

------------------------------

 RMSE Cross-Validation: 35887.852791598336
```

Lasso Regression:

In [20]:

```
lasso = Lasso()lasso.fit(X_train, y_train)predictions = lasso.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)

print("MAE:", mae)

print("MSE:", mse)

print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(lasso)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "Lasso","MAE": mae, "MSE": mse, "RMSE": rmse,

"R2 Score": r_squared, "RMSE (Cross-Validation)": rmse_cross_val}model

s = models.append(new_row, ignore_index=True)
```

Out[20]:

```
MAE: 23560.45808027236

MSE: 1414337628.502095

RMSE: 37607.680445649596

R2 Score: 0.815609194407292
```

```
------------------------------

 RMSE Cross-Validation: 35922.76936876075
```

Elastic Net:

In [21]:

```
elastic_net = ElasticNet()elastic_net.fit(X_train, y_train)predictions = elasti

c_net.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)

print("MAE:", mae)

print("MSE:", mse)

print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(elastic_net)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "ElasticNet","MAE": mae, "MSE": mse, "RMSE": r

mse, "R2 Score": r_squared, "RMSE (Cross-Validation)": rmse_cross_val}

models = models.append(new_row, ignore_index=True)
```

Out[21]:

MAE: 23792.743784996732

MSE: 1718445790.1371393

RMSE: 41454.14080809225

R2 Score: 0.775961837382229

RMSE Cross-Validation: 38449.00864609558

Support Vector Machines:

In [22]:

```
svr = SVR(C=100000)svr.fit(X_train, y_train)predictions = svr.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)

print("MAE:", mae)

print("MSE:", mse)

print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(svr)
```

```
print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "SVR","MAE": mae, "MSE": mse, "RMSE": rmse, " R2 Score": r_squared, "RMSE (Cross-Validation)": rmse_cross_val}models = models.append(new_row, ignore_index=True)
```

Out[22]:

```
MAE: 17843.16228084976

MSE: 1132136370.3413317

RMSE: 33647.234215330864

R2 Score: 0.852400492526574

------------------------------

RMSE Cross-Validation: 30745.475239075837
```

Random Forest Regressor:

In [23]:

```
random_forest = RandomForestRegressor(n_estimators=100)random_forest. fit(X_train, y_train)predictions = random_forest.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)

print("MAE:", mae)

print("MSE:", mse)
```

```
print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(random_forest)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "RandomForestRegressor","MAE": mae, "MSE": ms

e, "RMSE": rmse, "R2 Score": r_squared, "RMSE (Cross-Validation)": rms

e_cross_val}models = models.append(new_row, ignore_index=True)
```

Out[23]:

```
MAE: 18115.11067351598

MSE: 1004422414.0219476

RMSE: 31692.623968708358

R2 Score: 0.869050886899595

------------------------------

RMSE Cross-Validation: 31138.863315259332
```

XGBoost Regressor:

In [24]:

```
xgb = XGBRegressor(n_estimators=1000, learning_rate=0.01)xgb.fit(X_trai
```

```
n, y_train)predictions = xgb.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)

print("MAE:", mae)

print("MSE:", mse)

print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(xgb)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "XGBRegressor","MAE": mae, "MSE": mse, "RMS

E": rmse, "R2 Score": r_squared, "RMSE (Cross-Validation)": rmse_cross_

val}models = models.append(new_row, ignore_index=True)

Out[24]:

MAE: 17439.918396832192

MSE: 716579004.5214689

RMSE: 26768.993341578403

R2 Score: 0.9065777666861116

------------------------------
```

RMSE Cross-Validation: 29698.84961808251

Polynomial Regression (Degree=2)

In [25]:

```
poly_reg = PolynomialFeatures(degree=2)X_train_2d = poly_reg.fit_transfo

rm(X_train)X_test_2d = poly_reg.transform(X_test)

lin_reg = LinearRegression()lin_reg.fit(X_train_2d, y_train)predictions = li

n_reg.predict(X_test_2d)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)

print("MAE:", mae)

print("MSE:", mse)

print("RMSE:", rmse)

print("R2 Score:", r_squared)

print("-"*30)rmse_cross_val = rmse_cv(lin_reg)

print("RMSE Cross-Validation:", rmse_cross_val)

new_row = {"Model": "Polynomial Regression (degree=2)","MAE": mae, " MSE": mse,
"RMSE": rmse, "R2 Score": r_squared, "RMSE (Cross-Validat

ion)": rmse_cross_val}models = models.append(new_row, ignore_index=True)
```

Out[25]:

```
MAE: 2382228327828308.5

MSE: 1.5139911544182342e+32

RMSE: 1.230443478758059e+16

R2 Score: -1.9738289005226644e+22

------------------------------

RMSE Cross-Validation: 36326.451444669496
```

5. Deployment and Application:

If the model meets the desired performance criteria, it can be deployed in a real-world application or decision-making process. Deployment might involve integrating the model into existing systems or applications, enabling it to make real-time predictions and decisions.

6. Monitoring and Maintenance:

To ensure the model continues to perform well over time, monitoring and maintenance are crucial. This involves regularly checking for data drift, model performance degradation, and concept drift, and retraining the model as needed.

Throughout this process, ethical considerations, biases, and privacy should be carefully addressed to ensure that the AI-driven exploration and prediction project adheres to ethical and legal standards. The project's ultimate goal is to empower organizations and individuals with data-driven insights and predictions to improve decision-making and achieve their goals.

Various features to perform Exporatory Data Analysis(EDA):

- **Descriptive Statistics:**
    - Measures of central tendency (mean, median, mode).
    - Measures of dispersion (variance, standard deviation, range).
    - Quantiles (e.g., quartiles, percentiles).
- **Data Distribution Characteristics:**
    - Histograms: Visual representation of data distribution.
    - Box plots: Show summary statistics and identify outliers.
    - Kernel density plots: Estimate probability density functions.
- **Correlation Analysis:**

    - Correlation coefficients (e.g., Pearson, Spearman) to measure relationships between numerical variables.
    - Correlation matrices and heatmaps for visualizing correlations.
- **Categorical Data Exploration:**
    - Frequency tables: Show the distribution of categories within categorical variables.
    - Bar charts and pie charts: Visualize categorical data distribution.
- Multivariate Analysis:
    - Scatter plots: Reveal relationships between pairs of numerical variables.
    - Pair plots (scatterplot matrices): Display pairwise relationships for multiple variables.
- **Time-Series Analysis:**
    - Line plots: Visualize time-series data.
    - Decomposition: Separate time-series data into trend, seasonality, and noise components.
- **Geospatial Visualization:**
    - Geographic heatmaps: Plot data points on a map.
    - Choropleth maps: Visualize spatial data using color-coding.
- **Outlier Detection:**
    - Identification of outliers through visualization and statistical methods.
    - Box plots and scatter plots are commonly used for this purpose.

- **Data Transformation:**
    - Log transformation: Useful for data that exhibits exponential growth or extreme skewness.
    - Z-score standardization: Transform data to have mean=0 and standard deviation=1.
- **Data Quality Assessment:**
    - Handling missing data: Identify missing values and assess their impact.
    - Duplicate data detection: Identify and remove duplicate records.
- **Dimensionality Reduction:**
    - Principal Component Analysis (PCA): Reduce dimensionality by extracting important components.
    - t-SNE (t-Distributed Stochastic Neighbor Embedding): Visualize high-dimensional data in lower dimensions.
- **Data Splitting:**
    - Divide data into training, validation, and test sets for model evaluation.
- **Visualization Tools:**
    - Use libraries such as Matplotlib, Seaborn, Plotly, and others for creating interactive visualizations.
    - Dashboard tools like Tableau or Power BI can be useful for advanced visualization.
- **Pattern Detection**:

    - Identify trends, cycles, and repeating patterns in time-series and sequential data.
    - Clustering techniques like K-means or DBSCAN can reveal data groupings.
- Comparative Analysis:
    - Compare multiple groups or subsets within the data to understand differences.

Various feature to perform feature engineering:

- **Binning or Bucketing:**
    - Convert continuous numerical variables into discrete bins or buckets.
    - Useful for converting age or income ranges into categories.
- **Log Transformations:**
    - Apply logarithmic transformations to features to handle data that follows an exponential distribution.
    - Common for variables like income, population, or other highly skewed data.
- **Scaling and Normalization:**
    - Scale numerical features to a common range (e.g., 0 to 1) to ensure that they contribute equally to the model.
    - Common techniques include Min-Max scaling and Z-score normalization.

- **Interaction Features:**
    - Create new features by combining or interacting with existing ones.
    - For example, multiplying "Length" and "Width" to create an "Area" feature.
- **Polynomial Features:**
    - Generate polynomial features by squaring or cubing numerical variables to capture non-linear relationships.
    - Useful when relationships aren't linear
- **Date and Time Features:**
    - Extract components from date and time variables, such as year, month, day of the week, or time of day.
    - Helps capture seasonality and cyclical patterns.

Various feature to perform predictive modeling:

- **Numerical Features:**
    - Continuous numerical variables, such as age, income, temperature, or counts.
    - Often used in regression and some classification tasks.
- **Categorical Features:**
    - Discrete variables with distinct categories, like gender, city, or product type.
    - These are typically one-hot encoded or label encoded for modeling.
- **Textual Features:**
    - Features derived from text data, such as word counts, TF-IDF scores, or word embeddings.
    - Common in natural language processing (NLP) and text classification tasks.
- **Time-Series Features:**
    - Temporal data, including timestamps, dates, and time intervals.
    - Used in time-series forecasting and analysis.
- **Geospatial Features:**
    - Features related to geographical data, such as latitude, longitude, or postal codes.
    - Valuable for location-based applications and spatial analysis.
- **Cyclical Features:**
    - Variables representing cyclical patterns, like time of day, day of the week, or seasons.

- Useful in modeling seasonal trends.

- **Interaction Features:**
  - Features created by combining two or more variables or performing operations (e.g., addition, multiplication, division) on them.
  - Captures relationships between variables.
- **Derived Features:**
  - Features that are derived from domain knowledge or expertise, potentially complex mathematical calculations, or other transformations.
  - Can include ratios, indices, or scores specific to the problem.
- **Count Features:**
  - Represent counts of events or occurrences, such as the number of purchases, website visits, or customer interactions.
  - Common in recommendation systems and customer behavior analysis.
- **Temporal Features:**
  - Features that capture time-related patterns, like moving averages, exponential smoothing, or time lags.
  - Used in time-series forecasting and analysis.

CONCLUSION:

In conclusion, building an AI-driven exploration and prediction project that encompasses exploratory data analysis (EDA), feature engineering, and predictive modeling is a powerful and data-driven approach to extracting valuable insights and making informed decisions. This process represents a structured journey toward uncovering hidden patterns, creating informative features, and achieving accurate predictions. Here are key takeaways from this endeavor:

- **Data-Driven Decision Making**: The project is rooted in the principle that data is a valuable asset for making informed decisions. By exploring the data, engineering features, and building predictive models, organizations and individuals can leverage data-driven insights for a wide range of applications.
- **Exploratory Data Analysis (EDA):** EDA is the foundation of the project, providing a thorough understanding of the dataset. EDA involves data cleaning, visualization, and statistical analysis to reveal data characteristics, patterns, and relationships. This phase sets the stage for the subsequent feature engineering and modeling.
- **Feature Engineering:** The art of creating, selecting, and transforming features is instrumental in improving model performance. By refining the dataset through feature engineering, we ensure that the models can effectively capture the underlying patterns in the data. This phase requires domain knowledge and creativity.
- **Predictive Modeling:** The heart of the project, predictive modeling, uses machine learning algorithms to make accurate predictions. Model selection, training, evaluation, and

validation are integral to this phase. The choice of models and techniques is tailored to the specific predictive task and data characteristics.

- **Iterative Process**: The project is iterative by nature. Exploratory data analysis often leads to new insights that inform feature engineering choices. Feature engineering can reveal patterns that guide predictive modeling. Continuous refinement based on feedback and results is essential to achieving the best outcomes.
- **Transparency and Interpretability:** Ensuring that models are transparent and interpretable is crucial for understanding how predictions are made. Techniques for model interpretability and explainability should be applied as needed.
- **Ethical Considerations**: Throughout the project, ethical considerations are paramount. This includes addressing issues of bias, fairness, privacy, and data security to ensure that the project aligns with ethical and legal standards.
- **Continuous Improvement:** Building AI-driven exploration and prediction projects is not a one-time endeavor. Models require ongoing monitoring, maintenance, and updates to adapt to changing data patterns and evolving business needs.
- **Empowering Decision-Makers:** Ultimately, the project empowers decision-makers with the tools and insights to make better-informed choices. Whether it's predicting market trends, identifying anomalies, or forecasting future outcomes, the project's results can have a substantial impact on decision-making processes.

In today's data-rich landscape, AI-driven exploration and prediction projects have the potential to revolutionize various domains, including business, healthcare, finance, and more. By embracing the power of data and artificial intelligence, organizations and individuals can harness the benefits of data-driven decision-making and stay ahead in a data-driven world.