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

**-K Chandu**

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

**Phase-4: Development Part 2**

**Topic: AI-driven exploration and prediction project by Performing exploratory data analysis, Feature engineering and Predictive modeling.**

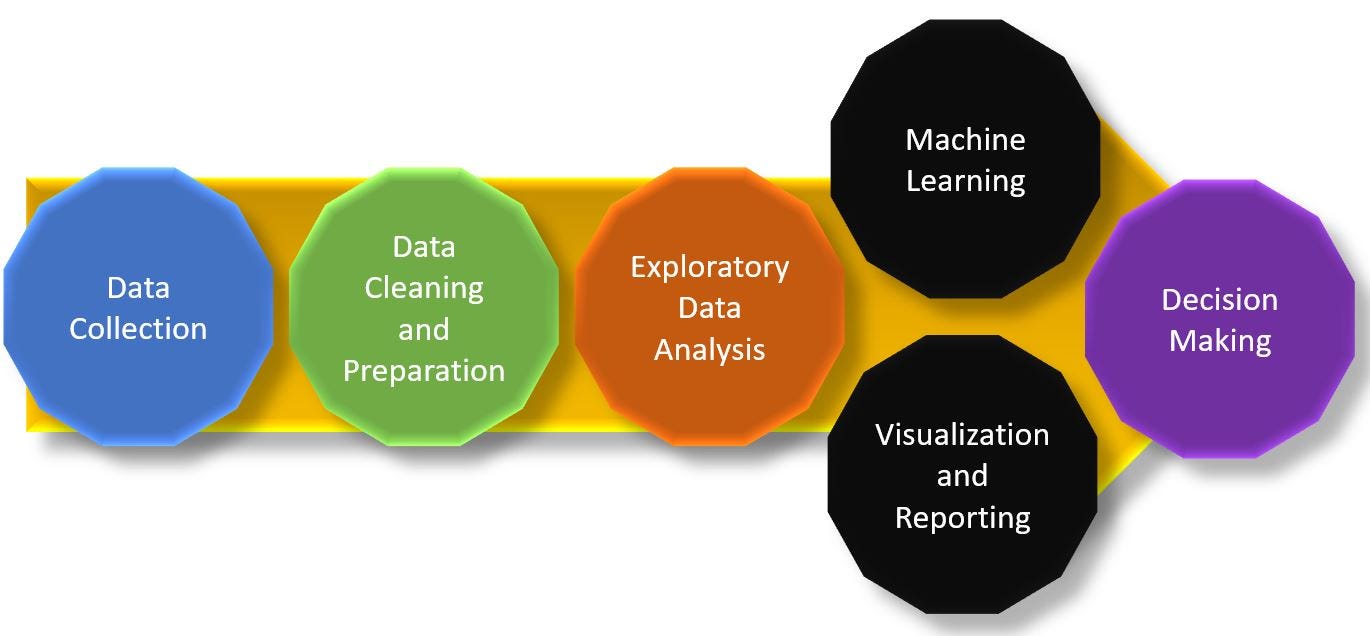
The development phase of AI-driven exploration and prediction project for company registration trends with the Registrar of Companies (RoC) is where you transition from data preparation to building and evaluating predictive models. During this phase, you will leverage the preprocessed dataset to create machine learning models that can provide insights and predictions regarding company registration trends.

This phase is a crucial step in harnessing the power of artificial intelligence to make informed decisions and forecasts in the business and regulatory domains. Central to the development phase is the utilization of your preprocessed dataset as a cornerstone for constructing machine learning models. These models are designed to decipher patterns, uncover trends, and make predictions regarding company registration trends. The choice of models depends on the nature of your prediction task. If you aim to predict numerical outcomes, such as the quantity of new company registrations, regression models will be the avenue of choice. In contrast, if your objective is to forecast categorical outcomes, like whether registrations will increase or decrease, classification models become the instrument of preference.

# Step-by-Step Exploratory Data Analysis (EDA) using Python

## Introduction to EDA

The main objective of this article is to cover the steps involved in Data pre-processing, Feature Engineering, and different stages of Exploratory Data Analysis, which is an essential step in any research analysis. Data pre-processing, Feature Engineering, and EDA are fundamental early steps after data collection. Still, they are not limited to where the data is simply visualized, plotted, and manipulated, without any assumptions, to assess the quality of the data and building models. This article will guide you through data pre-processing, feature engineering, and EDA using Python.



### **Step 1: Import Python Libraries**

The first step involved in ML using python is understanding and playing around with our data using libraries.

Import all libraries which are required for our analysis, such as Data Loading, Statistical analysis, Visualizations, Data Transformations, Merge and Joins, etc.

**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')

### **Step 2: Reading Dataset**

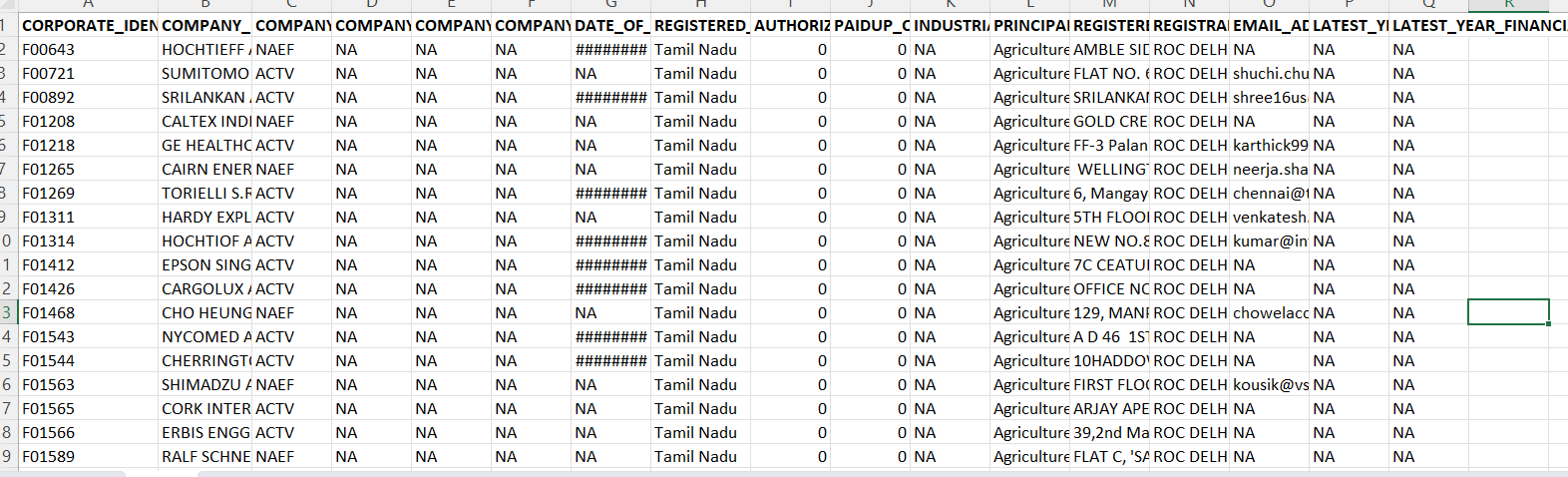
The Pandas library offers a wide range of possibilities for loading data into the pandas DataFrame from files like JSON, .csv, .xlsx, .sql, .pickle, .html, .txt, images etc.

Most of the data are available in a tabular format of CSV files. It is trendy and easy to access. Using the**read\_csv()** function, data can be converted to a pandas DataFrame.

**Code:**

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

**Given Dataset:**



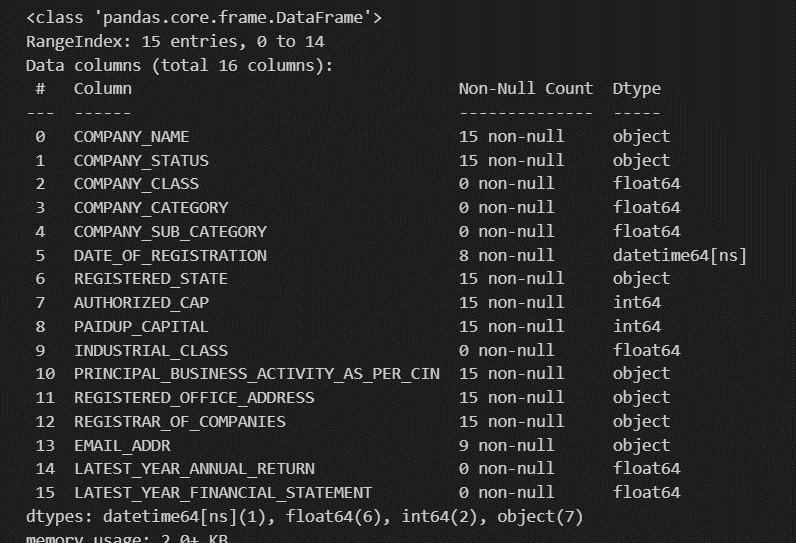
### **Step 3: Data Reduction**

Some columns or variables can be dropped if they do not add value to our analysis.

**Code:**

data = data.drop(['CORPORATE\_IDENTIFICATION\_NUMBER'], axis = 1)

data.info()



### **Step 4: EDA Univariate Analysis**

Analyzing/visualizing the dataset by taking one variable at a time:

Data visualization is essential; we must decide what charts to plot to better understand the data. In this article, we visualize our data using Matplotlib and Seaborn libraries.Matplotlib is a Python 2D plotting library used to draw basic charts we use Matplotlib.Seaborn is also a python library built on top of Matplotlib that uses short lines of code to create and style statistical plots from Pandas and NumpyUnivariate analysis can be done for both Categorical and Numerical variables.Categorical variables can be visualized using a Count plot, Bar Chart, Pie Plot, etc.Numerical Variables can be visualized using Histogram, Box Plot, Density Plot, etc.In our example, we have done a Univariate analysis using Histogram and  Box Plot for continuous Variables.In the below fig, a histogram and box plot is used to show the pattern of the variables, as some variables have skewness and outliers.

**Code:**

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);

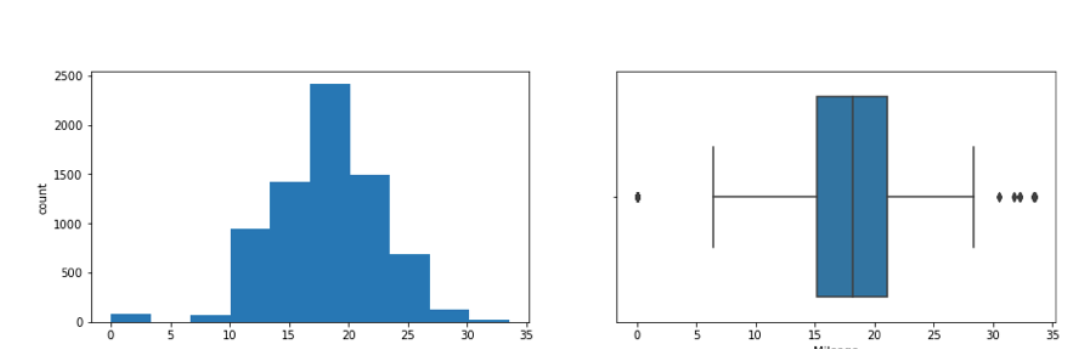
sns.countplot(ax = axes[1, 0], x = 'PAIDUP\_CAPITAL', data = data, color = 'blue',order = data['PAIDUP\_CAPITAL'].value\_counts().index);

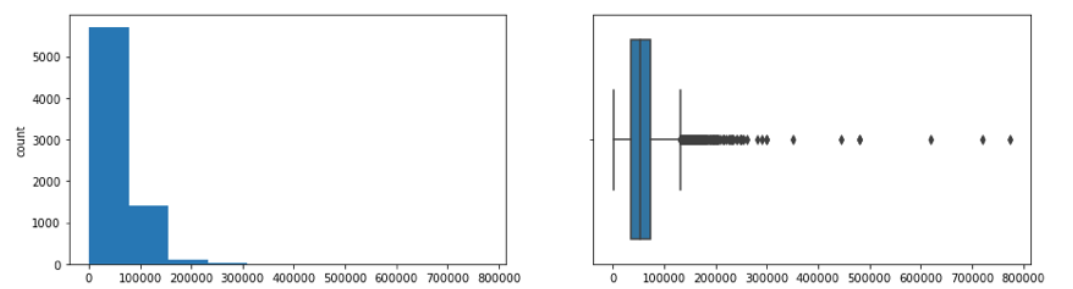
sns.countplot(ax = axes[1, 1], x = 'AUTHORIZED\_CAP', data = data, color = 'blue',order = data['AUTHORIZED\_CAP'].value\_counts().index);

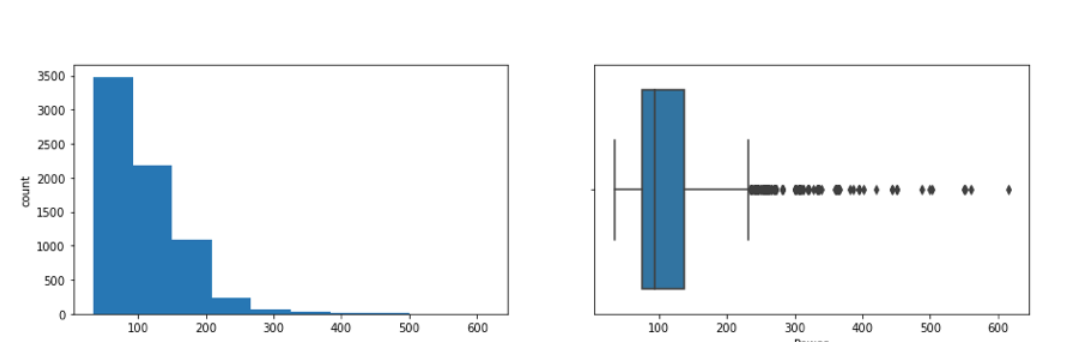
axes[1][1].tick\_params(labelrotation=45);

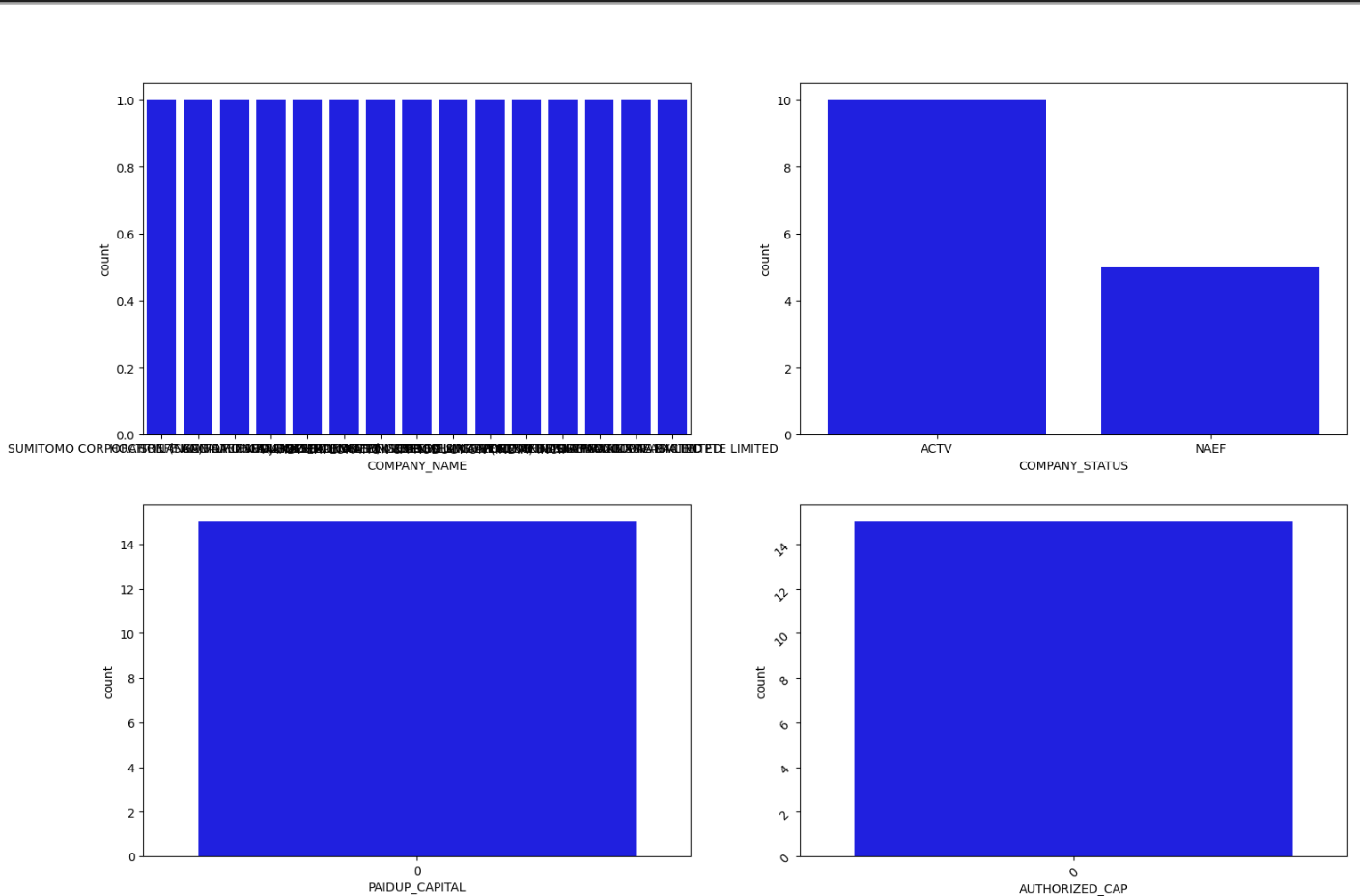
axes[2][0].tick\_params(labelrotation=90);

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









### **Step 5: EDA Multivariate Analysis**

As the name suggests, [Multivariate analysis](https://www.analyticsvidhya.com/blog/2021/04/exploratory-analysis-using-univariate-bivariate-and-multivariate-analysis-techniques/) looks at more than two variables. Multivariate analysis is one of the most useful methods to determine relationships and analyze patterns for any dataset.

**A heat map is widely been used for Multivariate Analysis**

Heat Map gives the correlation between the variables, whether it has a positive or negative correlation.

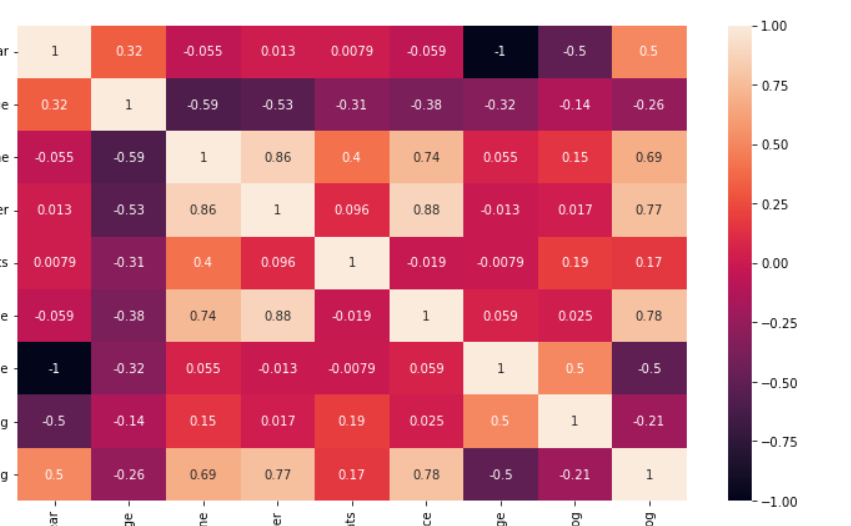
In our example heat map shows the correlation between the variables.

**Code:**

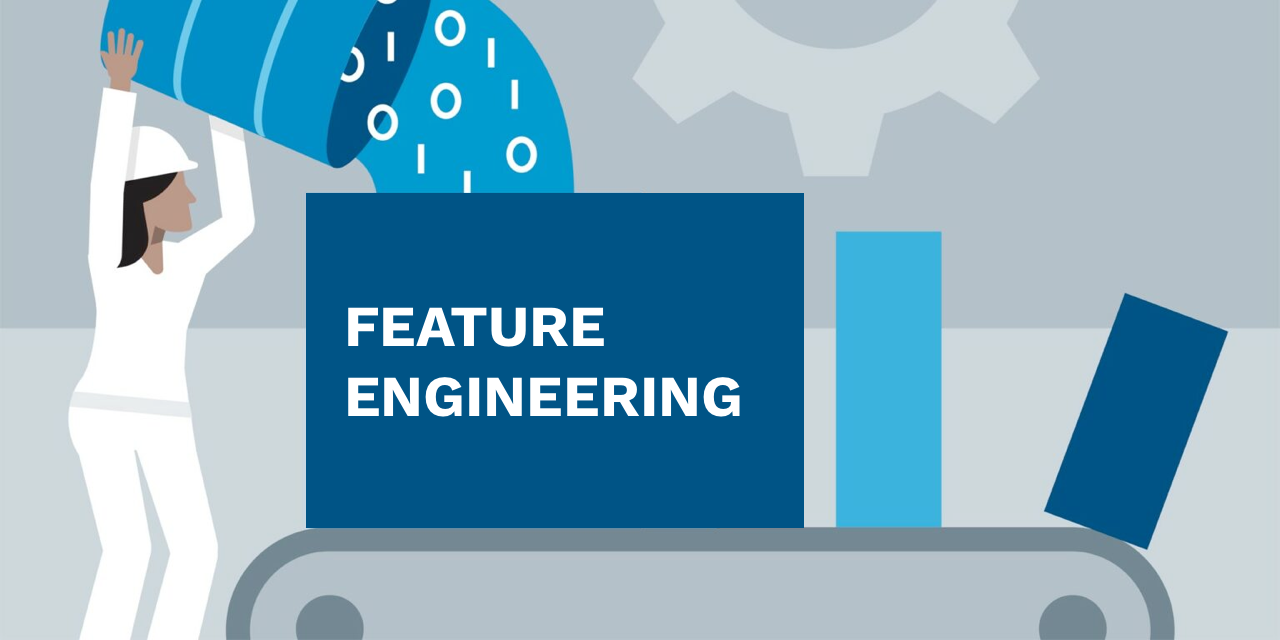
plt.figure(figsize=(12, 7))

sns.heatmap(data.drop(['INDUSTRIAL\_CLASS','CORPORATE\_IDENTIFICATION\_NUMBER'],axis=1).corr(), annot = True, vmin = -1, vmax = 1)

plt.show()



# Feature Engineering

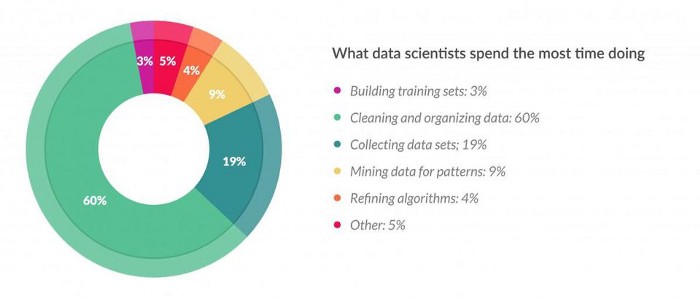


Feature Engineering is one of the beautiful arts which helps you to represent data in the most insightful possible way. It entails a skilled combination of subject knowledge, intuition, and fundamental mathematical skills. You are effectively transforming your data properties into data features when you undertake feature engineering. How you provide your data to your algorithm should effectively denote the relevant structures/properties of the underlying information.

Feature engineering is the process of pre-processing data so that your model/learning algorithm may spend as little time as possible sifting through the noise. Any information that is unrelated to learning or forecasting concerning your final aim is known as noise.

According to a Forbes poll, data scientists spend 80% of their time preparing data:

This measure demonstrates the significance of feature engineering in data science. As a result, I decided to start this guide, which highlights the key strategies of feature engineering and provides brief descriptions of each.



## Iterative steps for Feature Engineering

1. Get deep into the topic, look at a lot of data, and see what you can learn from feature engineering on other challenges.
2. You can make use of automated feature extraction, manual feature construction, or a mixture of the two, depending on your problem.
3. To prepare one or more views for your models to operate on, use various feature importance scores and feature selection approaches.
4. Estimate model accuracy on new data using the features you’ve chosen.

## ****Missing value treatment for Feature Engineering****

Three types of Missing Data are Missing at Random(MAR), Missing Completely at Random(MCAR), Missing Not at Random(MNAR). For brief information go through this.

### **1. Deletion**

Two types: Listwise and Pairwise Deletion.

**Listwise:** Here, we discard observations in which one or more variables are missing. One of the key advantages of this strategy is its simplicity; nevertheless, because the sample size has reduced, it limits the model’s power.

**Pairwise:** We undertake analysis for all situations in which the variables of interest are present in pairwise deletion. This strategy has the advantage of keeping as many examples available for study as possible. One of the method’s drawbacks is that various sample sizes had used for different variables.

When the nature of missing data is “Missing fully at random,” deletion methods are applied; otherwise, non-random missing values can bias the model output.

### **2. Mean/Median/Mode**

Imputation is a technique for replacing missing values with estimates. The goal is to use known associations that seem in the valid values of the data set to help estimate the missing values. It is one of the most widely utilized techniques. It entails using the mean, median, or mode to replace missing data for a specific attribute

**Code:**

male =data[data['Gender']=='M']

female =data[data['Gender']=='F']

# mean for male

print(round(male['Manpower'].mean(skipna=True),2))

# mean for female

print(round(female['Manpower'].mean(skipna=True),2))

### **3. Prediction Model**

One of the more sophisticated methods for dealing with missing data is the prediction model. Here, we build a predictive model to estimate values that will fill in for missing data. In our case, we split our data set into two sections: one with no missing values for the variable and one with missing values.

The first data set becomes the model’s training data set, and the second data set with missing values becomes the model’s test data set, and missing values are known as the target variable. Then, based on the other attributes of the training data set, we build a model to predict the target variable and populate missing values in the test data set. To accomplish this, we can employ regression, ANOVA, logistic regression, and other modeling techniques. This method has two disadvantages:

* The estimated values of the model are usually better behaved than the real values.
* If there are no relationships between attributes in the data set and the attribute with missing values, the model will be inaccurate in estimating missing values.

### **4. KNN Imputation**

The missing values of an attribute are imputed using the given number of features that are most similar to the feature whose values are missing in this method of imputation. A distance function helps to determine the similarity of two attributes. It is also known to have specific benefits and drawbacks.

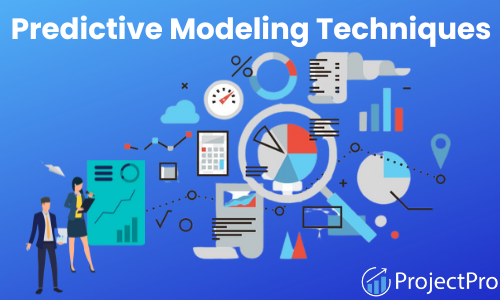
**Pros**

* k-nearest neighbor helps to predict both qualitative and quantitative attributes.
* It is not necessary to create a predictive model for each feature with missing data.
* Features with multiple missing values are simple to handle.
* The data’s correlation structure has been taken into account.

**Cons**

* When analyzing large databases, the KNN algorithm takes a long time. It searches the entire dataset for the most similar instances.
* The choice of k-value is critical. Here, we include attributes that were significantly different from what we require, in the case of a higher k value. A lower k value implies that significant features are missing.

# Predictive Analytics



## Introduction

Predictive analytics powered by AI technology is changing the world as we know it. Nowadays, there is an ocean of data available for every industry. However, finding the gems that answer your business questions is as hard as navigating a real ocean on a raft. The majority of tools available now are either lacking in capacity or speed to process vast swarms of information, let alone draw insights from them. This is where predictive analytics AI-powered platforms are coming into play. These solutions have the potential to make valuable insights obtained from big data accessible to any business. Even companies with limited budgets and time-sensitive campaigns can draw great value from such SaaS tools.

SaaS platforms, in general, are diverse and getting more popular with every passing day. It’s a well-known fact that SaaS companies often [have the best customer success teams](https://thinkmobiles.com/posts/22/). And developers understand that increasing customer success will grow their own revenue. SaaS platforms for predictive analytics have the potential to take your business’ success further.

## AI-Powered Predictive Analytics Platforms Are the Future of Tech

AI-powered tech is making huge waves right now and with good reason. The majority of people might not even notice it, but AI is present in nearly all tech-related aspects of our lives. This presence is sure to grow further as, [according to McKinsey research](https://www.mckinsey.com/featured-insights/artificial-intelligence/five-fifty-real-world-ai?cid=fivefifty-soc-twi-mip-mck-oth-1806&kui=1EJ9eNZ-MV-uuDAyc2-RSA), even at its current level of development, AI can resolve over 400 longstanding business challenges.

SaaS predictive analytics platforms enable SMEs and entrepreneurs to use the processing power of AI. This service is vitally important in the competitive market of today. There can be no doubt that technology will develop further in response to the demand.

However, one needs to remember that these platforms have to integrate with existing tech. There are also legal considerations to keep in mind. All SaaS platforms must have a well-thought-out agreement to protect both the developer and customers from litigation. And this situation will gain new layers of complexity as AI technology becomes more prevalent in the world.