PHASE - V

STOCK PRICE PREDICTION

CONTENTS:

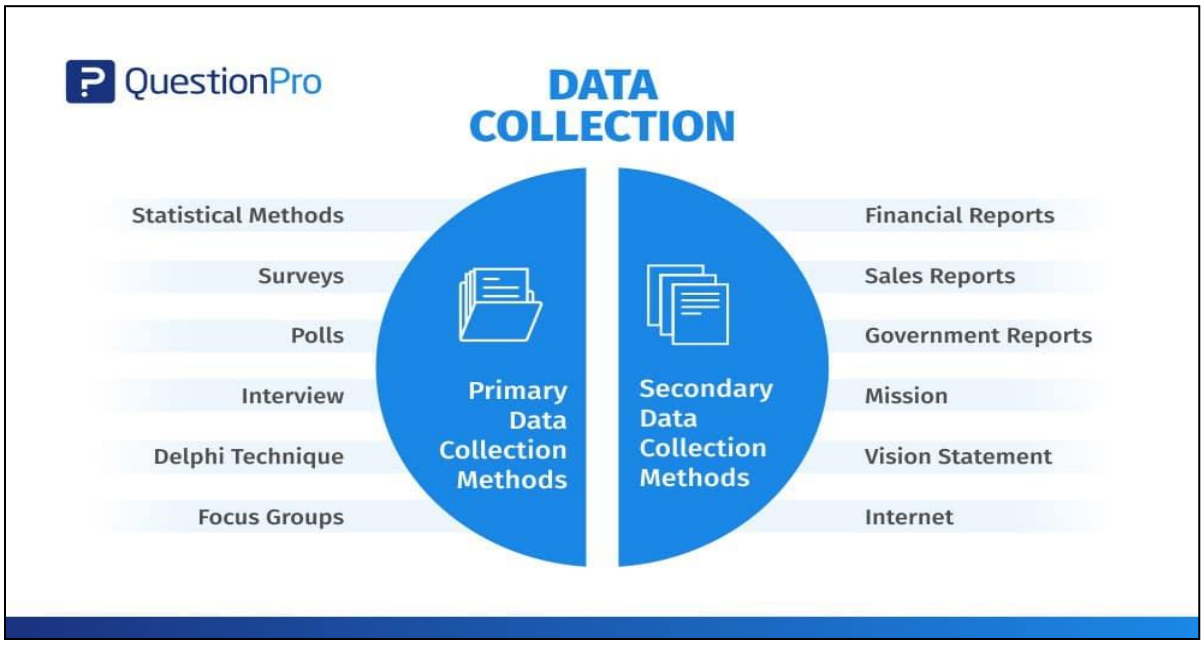
PROBLEM DEFINITION:

Understanding the project objectives and requirements from a domain perspective then converting this knowledge into a data science problem definition with a preliminary plan designed to achieve the objectives. Data science projects are often structured around the specific needs of an industry sector (as shown below) or even tailored and built for a single organization. A successful data science project starts from a well-defined question or need.

DESIGN THINKING:

Data collection:

The process of gathering and analyzing accurate data from various sources to find answers to research problems, trends, and probabilities, etc., to evaluate possible outcomes is known as Data Collection.



Methods of data collection:

- Surveys, quizzes, and questionnaires.
- Interviews.
- Focus groups.
- Direct observations
- Documents and records (and other types of secondary data, which won't be our focus here.

Data collection uses:

The data collection uses provides the information that's needed to answer questions, analyze business performance or other outcomes, and predict future trends, actions and scenarios. In businesses, data collection happens on multiple levels.

Data collection design:

Last updated on Aug 17, 2023. Data collection instruments are the tools and methods you use to gather and record information for your research or evaluation project. They can include surveys, questionnaires, interviews, focus groups, observations, tests, and more

DATA PREPROCESSING:

Data processing, manipulation of data by a computer. It includes the conversion of raw data to machine-readable form, flow of data through the CPU and memory to output devices, and formatting or transformation of output. Any use of computers to perform defined operations on data can be included under data processing.

The four main stages of data processing cycle are:

- Data collection.
- Data input.
- Data processing.
- Data output.

A very simple example of a data processing system is the process of maintaining a check register. Transactions checks and deposits are recorded as they occur and the transactions are summarized to determine a current balance.

Data processing is an essential component of modern computing and communication.

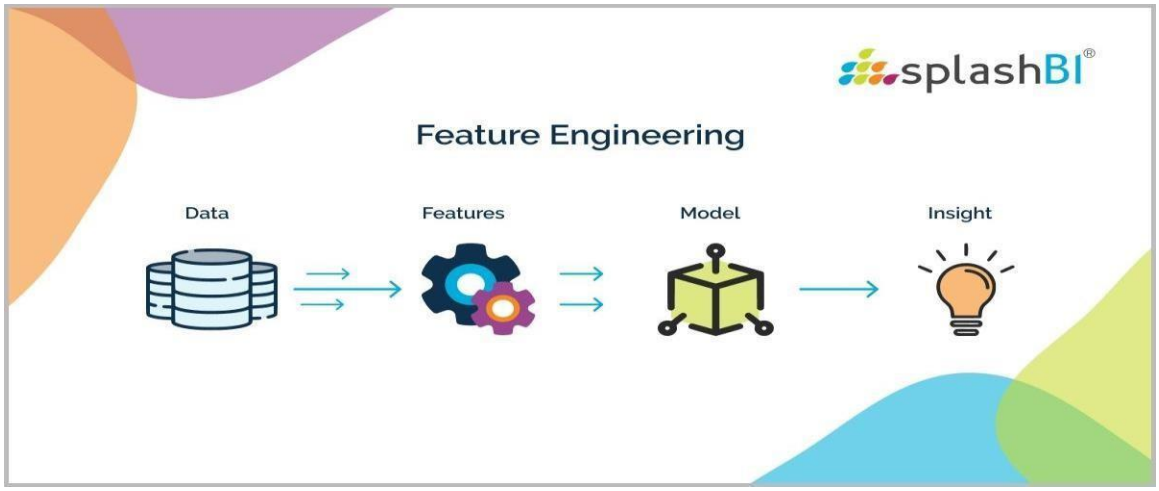
It involves the manipulation, analysis, storage, and retrieval of data in order to produce. Read More. The future of data processing will be driven by advances in technology, such as artificial intelligence and machine learning.
Without data processing, companies limit their access to the very data that can hone their competitive edge and deliver critical business insights. That's why it's crucial for all companies to understand the necessity of processing all their data, and how to go about it.

Data processing cycle:

The data processing cycle is the set of operations used to transform data into useful information. The intent of this processing is to create actionable information that can be used to enhance a business.

FEATURE ENGINEERING:

Feature engineering involves a set of techniques that enable us to create new features by combining or transforming the existing ones. These techniques help to highlight the most important patterns and relationships in the data, which in turn helps the machine learning model to learn from the data more effectively.



Feature engineering refers to manipulation addition, deletion, combination, mutation of your data set to improve machine learning model training, leading to better performance and greater accuracy. Effective feature engineering is based on sound knowledge of business problems and the available data sources. Feature engineering enables you to build more complex models than you could with only raw data. It also allows you to build interpret able models from any amount of data. Feature selection will help you limit these features to a manageable number

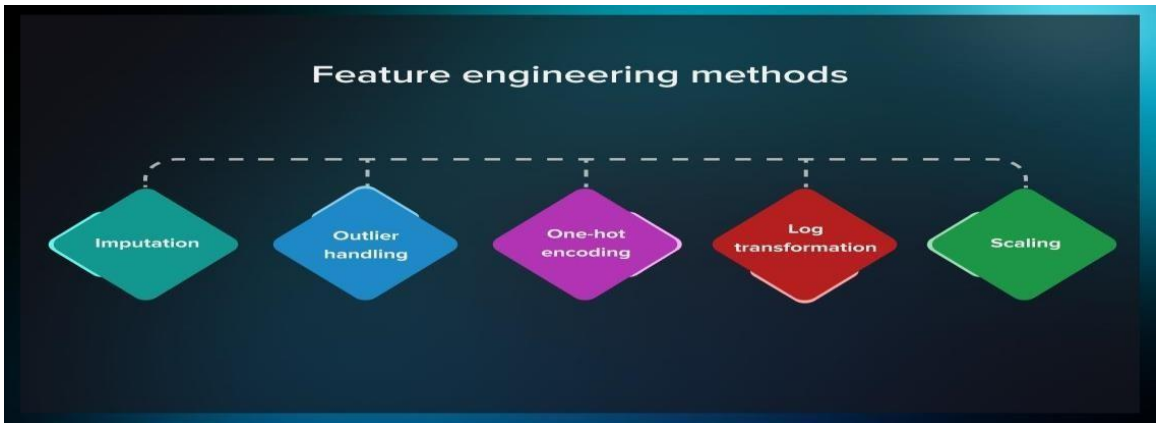
Types of feature Engineering:

•7 of the Most Used Feature Engineering Techniques. Hands-on Feature Engineering with Scikit-Learn, TensorFlow, Pandas and Spicy.
•Encoding. Feature encoding is a process used to transform categorical data into

numerical values that can be understood by ML algorithms.
•Feature Hashing.
•Binning / Bucketizing.
•Transformer.

***Feature Engineering methods*:**

Feature engineering is the process of transforming raw data into features that are suitable for machine learning models. In other words, it is the process of selecting, extracting, and transforming the most relevant features from the available data to build more accurate and efficient machine learning models



MODEL SELECTION:

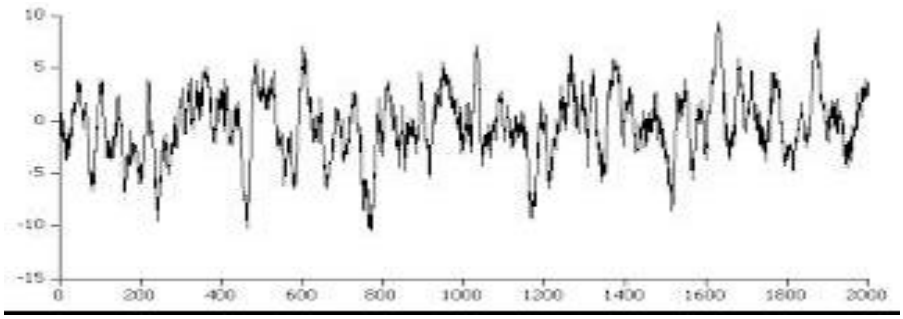
***Time Series Forecasting*:**

Time series forecasting is a data science task that is critical to a variety of activities within any business organization. Time series forecasting is a useful tool that can help to understand how historical data influences the future. This is done by looking at past data, defining the patterns, and producing short or long-term predictions.

There are four general components that a time series forecasting model is comprised of:

Trend: Increase or decrease in the series of data over longer a period.
Seasonality: Fluctuations in the pattern due to seasonal determinants over a period such as a day, week, month, season.

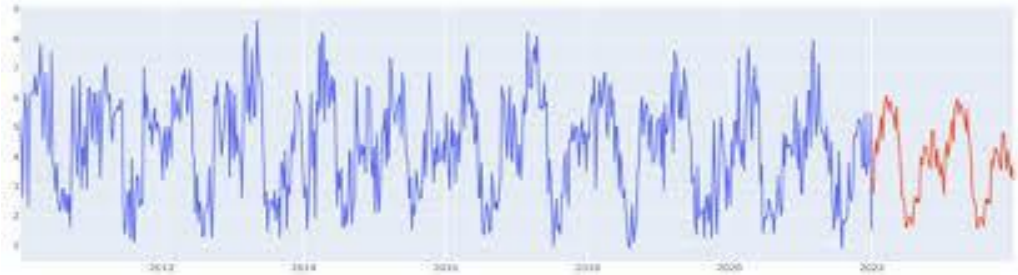
Cyclical variations: Occurs when data exhibit rises and falls at irregular intervals



Random or irregular variations: Instability due to random factors that do not repeat in the pattern.

Auto regressive *(AR):* An auto regressive (AR) model predicts future behavior based on past behavior. It's used for forecasting when there is some correlation between values in a time series and the values that precede and succeed them.

Auto regressive Integrated Moving Average *(ARIMA*): Auto Regressive Integrated Moving Average, ARIMA, models are among the most widely used approaches for time



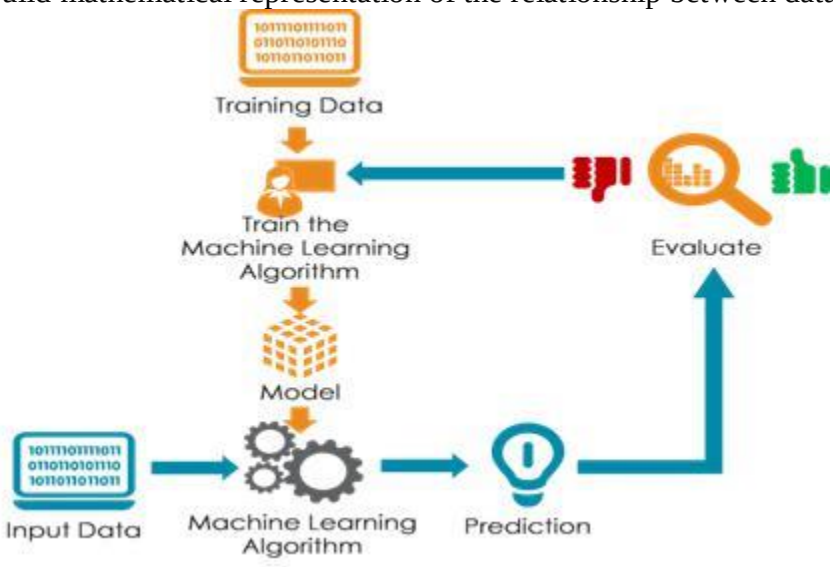
series forecasting. It is a class of models that 'explains' a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values.

Seasonal Auto regressive Integrated Moving Average *(SARIMA):* Seasonal auto regressive integrated moving average (SARIMA) models extend basic ARIMA models and allow for the incorporation seasonal patterns.

MODEL TRAINING:

Model training is the phase in the data science development lifecycle where practitioners
try to fit the best combination of weights and bias to a machine learning algorithm to minimize a loss function over the prediction range. The p urp ose of model training is to

build mathematical representation of the relationship between data features and a target



label (in supervised learning) or among the features themselves (unsupervised learning). Loss functions are a critical aspect of model training since they define how to optimize the machine learning algorithms. Depending on the objective, type of data and algorithm, data science practitioner use different type of loss functions. One of the

popular examples of loss functions is Mean Square Error (MSE).

Why is it Important?
Model training is the key step in machine learning that results in a model ready to be validated, tested, and deployed. The performance of the model determines the quality of the applications that are built using it. Quality of training data and the training algorithm are both important assets during the model training phase. Typically, training data is split for training, validation, and testing. The training algorithm is chosen based on the end use

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

case. There are several tradeoff points in deciding the best algorithm–model complexity, interpretability, performance, compute requirements, etc. All these aspects of model training make it both an involved and important process in the overall machine learning development cycle.

EVALUATION:
Maybe the most popular and simple error metric is MAE:

MAE: The Mean Absolute Error is defined as:

While the MAE is easily interpret able (each residual contributes proportionally to the total amount of error), one could argue that using the sum of the residuals is not the best choice, as we could want to highlight especially whether the model incur in some large errors.

MSE & RMSE*:* For those cases, maybe MSE (Mean Squared Error) or RMSE (Root Mean Squared Error) are a better choice. Here the error grows quadratic ally and therefore
extreme values penalize the metric to a greater extent.

MSE = Σ $(y_i - p_i)^2 n$,

RMSE = Square root of MSE.

The main problem with scale dependent metrics is that they are not suitable to compare errors from different sources.
In our case, the capacity of the power plants would determine the magnitude of the errors and therefore comparing them between facilities would not make much sense. This is something we should try to avoid when choosing the metric.

ABOUT DATASETS:

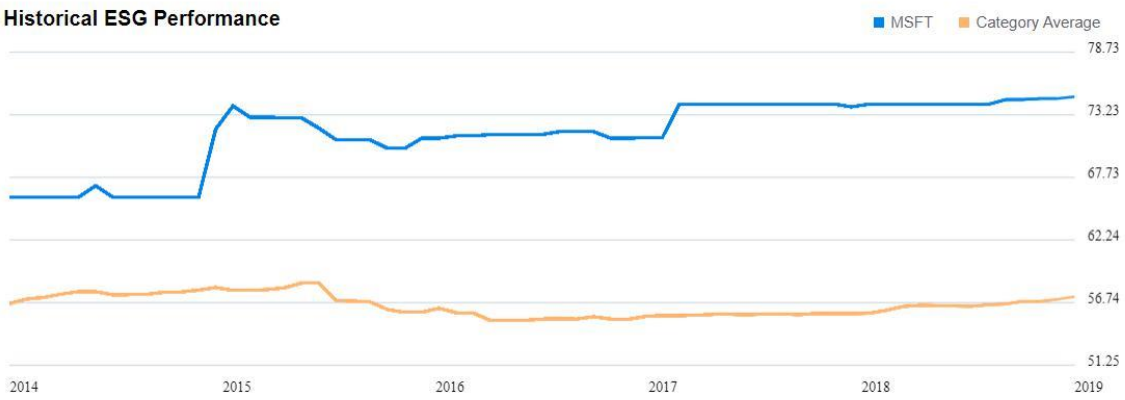
Kaggle is one of the largest communities of data scientists and machine learning practitioners in the world, and its platform hosts thousands of datasets covering a wide range
of topics and industries. With so many options to choose from, it can be difficult to know where to start or what datasets are worth exploring. That's where this dataset comes in. By
scraping information about the top 10,000 datasets on Kaggle, we have created a single

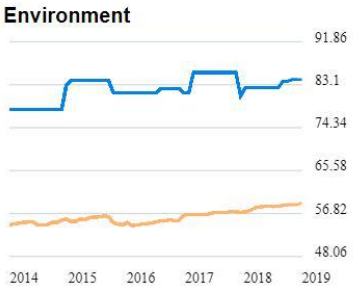
source of truth for the most popular and useful datasets on the platform. This dataset is not
just a list of names and numbers, but a valuable tool for data enthusiasts and professionals alike, providing insights into the latest trends and techniques in data science and machine Learning

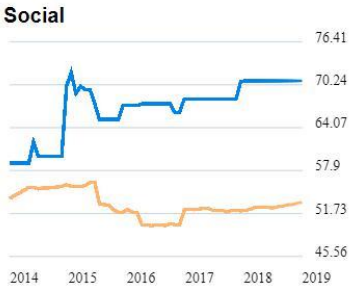
www.kaggle.com

Column Descriptions:

- **datasets _name** - Name of the datasets
- **Author_name** - Name of the author
- **Author_id** - Kaggle id of the author
- **No_of_files** - Number of files the author has uploaded
- ***size*** - Size of all the files
- **Type_of_file** - Type of the files such as csv, json etc.
- **Upvoste** - Total upvotes of the dataset
- **Medals** - Medal of the dataset
- **Usability** - Usability of the dataset
- **Date** - Date in which the dataset is uploaded.
- **Day** - Day in which the dataset is uploaded.
- **Time** - Time in which the dataset is uploaded.
- **Dataset_link** - Kaggle link of the dataset







WHAT IS PANDA IN PYTHON?

Pandas is an open-source Python library developed by Wes McKinney in 2008. It is used in data science, data analysis, and other machine-learning activities. It is very fast and provides

many tools for effectively handling large amounts of data. It is built on the Numpy library.

Why using pandas in Python?

Pandas strengthens Python by giving the popular programming language the capability to work with spreadsheet-like data enabling fast loading, aligning, manipulating, and merging,
in addition to other key functions. PANDAS is short for Pediatric Autoimmune Neuropsychiatric Disorders Associated with Streptococcal Infections. A child may be diagnosed with PANDAS when: Obsessive-compulsive disorder (OCD), tic disorder, or both
suddenly appear following a streptococcal (strep) infection, such as strep throat or scarlet fever.

What is Pandas and its types?

Pandas works with many different types of data sets such as comma-separated values (CSV)
files, Excel files, extensible markup language (XML) files, JavaScript object notation (JSON)
files and relational database tables. Data read from these sources are returned as Pandas data
types known as DataFrame and Series. Pandas a Python library? Pandas is a Python library
for data analysis. Started by Wes McKinney in 2008 out of a need for a powerful and flexible
quantitative analysis tool, pandas has grown into one of the most popular Python libraries.

Who uses Python pandas?

Data Scientists:

The pandas package is the most important tool at the disposal of Data Scientists and Analysts
working in Python today. The powerful machine learning and glamorous visualization tools
may get all the attention, but pandas is the backbone of most data projects. How pandas are
used in Python? Pandas is a Python library used for working with data sets. It has functions
for analyzing, cleaning, exploring, and manipulating data. The name "Pandas" has a reference
to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

Method: Using pip

It's a package installation tool that simplifies the installation of Python modules and frameworks.Pip will be installed with Python by default if you have a later version of Python available (greater than Python 3.5.x). If you're using an earlier version of Python, you'll need to install pipbefore you can install Pandas. The simplest method to accomplish
this is to update to the most recent version of Python, which can be found at this link.

Step 1: Launch Command Prompt

To open the start menu, use the Windows key on your keyboard or click the Start button. For example, when you type "cmd" the Command Prompt app should display in the start

menu, and once you can view the command prompt app, launch the app. Alternatively, you
may hit the Windows key + r to bring up the "RUN" box, where you can input “cmd” and then press enter. Itwill also launch the Command prompt.

Step 2: Enter the command

After you open the command prompt, the following step is to enter the needed command to begin the pip installation. For example, enter the command shown below. This willstart the pip installation. After downloading the necessary files, Pandas will be set to operate onyour computer. You can employ Pandas in your Python projects once the installation has been completed.

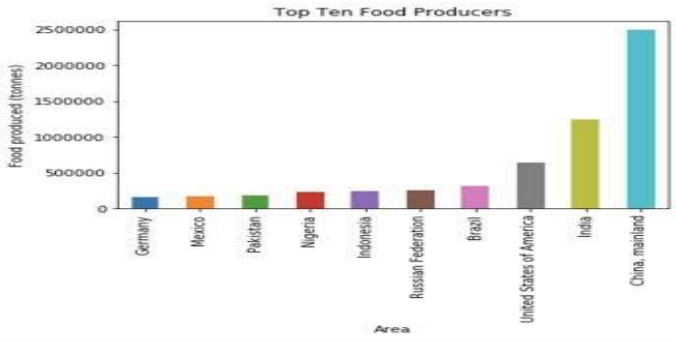
What is in a dataset?

A data set is an ordered collection of data. As we know, a collection of information obtained
through observations, measurements, study, or analysis is referred to as data. It could include
information such as facts, numbers, figures, names, or even basic descriptions of objects.

What is dataset with example?

A data set is a collection of numbers or values that relate to a particular subject. For example,
the test scores of each student in a particular class is a data set. The number of fish eaten by each dolphin at an aquarium is a data set

```
In [79]: # Select just data relevant to Food production
         plot_data = data[data['Element'] == 'Food']
         # Group the data by the Production Area and sum the values for 2013.
         plot_data = plot_data.groupby('Area')['Y2013'].sum()
         # Sort the values in ascending order, and plot
         # the last 10 values in a barchart
         plot_data.sort_values()[-10:].plot(kind='bar')
         plt.title("Top Ten Food Producers")
         plt.ylabel("Food produced (tonnes)")
Out[79]: Text(0,0.5,'Food produced (tonnes)')
```



.

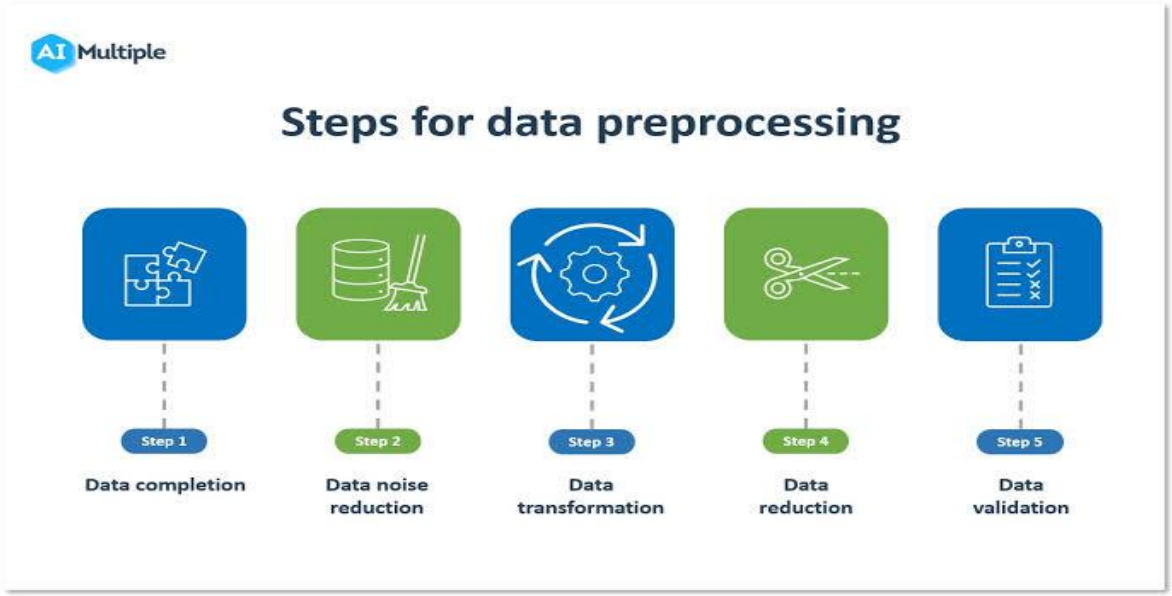
Steps to Constructing Your Data set

1. Collect the raw data.
2. Identify feature and label sources.
3. Select a sampling strategy.
4. Split the data.

Dataset index	Dataset	Number of molecules	Number of descriptors
ACT1	3A4	50,000	9491
ACT2	CB1	11640	5877
ACT3	DPP4	8327	5203
ACT4	HIVINT	2421	4306
ACT5	HIVPROT	4311	6274
ACT6	LOGD	50,000	8921
ACT7	METAB	2092	4595
ACT8	NK1	13482	5803
ACT9	OX1	7135	4730
ACT10	OX2	14875	5790
ACT11	PGP	8603	5135
ACT12	PPB	11622	5470
ACT13	RAT_F	7821	5698
ACT14	TDI	5559	5945
ACT15	THROMBIN	6924	5552

PREPROCESS DATASHEETS:

At the heart of Machine Learning is to process data. Your **machine learning tools are as good as the quality of your data**. This blog deals with the various steps of **cleaning data**. Your data needs to go through a few steps before it is could be used for making predictions.



Steps involved in data preprocessing :

1. Importing the required Libraries
2. Importing the data set
3. Handling the missing data.
4. Encoding Categorical Data.
5. Splitting the data set into test set and training set.
6. Feature Scaling.

Step 1: Importing the required Libraries

To follow along you will need to download this dataset : **Data.csv**

Every time we make a new model, we will require to import Numpy and Pandas. Numpy is a Library which contains Mathematical functions and is used for scientific computing while Pandas is used to import and manage the data sets.

```
import pandas as pd
import numpy as np
```

here we are importing the pandas and Numpy library and assigning a shortcut “pd” and “np” respectively.

Step 2: Importing the Dataset

Data sets are available in .csv format. A CSV file stores tabular data in plain text. Each line of the file is a data record. We use the read_csv method of the pandas library to read a local CSV file as a **dataframe**.

```
dataset = pd.read_csv('Data.csv')
```

After carefully inspecting our dataset, we are going to create a matrix of features in our dataset (X) and create a dependent vector (Y) with their respective observations. To read the columns, we will use iloc of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 3].values
```

Step 3: Handling the Missing Data

An example of Missing data and Imputation

The data we get is rarely homogenous. Sometimes data can be missing and it needs to be handled so that it does not reduce the performance of our machine learning model.

To do this we need to replace the missing data by the Mean or Median of the entire column. For this we will be using the sklearn.preprocessing Library which contains a class called Imputer which will help us in taking care of our missing data.

```
from sklearn.preprocessing import Imputer
imputer = Imputer(missing_values = "NaN", strategy = "mean", axis = 0)
```

Our object name is **imputer.** The Imputer class can take parameters like :

1. **missing_values** : It is the placeholder for the missing values.All occurrences of missing_values will be imputed. We can give it an integer or “NaN” for it to find missing values.
2. **strategy** : It is the imputation strategy — If “mean”, then replace missing values using the mean along the axis (Column). Other strategies include “median” and “most_frequent”.
3. **axis** : It can be assigned 0 or 1, 0 to impute along columns and 1 to impute along rows.

Now we fit the imputer object to our dataNow we fit the imputer object to our data.

```
```

Now replacing the missing values with the mean of the column by using transform method.

```
X[:, 1:3] = imputer.transform(X[:, 1:3])
```

Step 4: Encoding categorical data

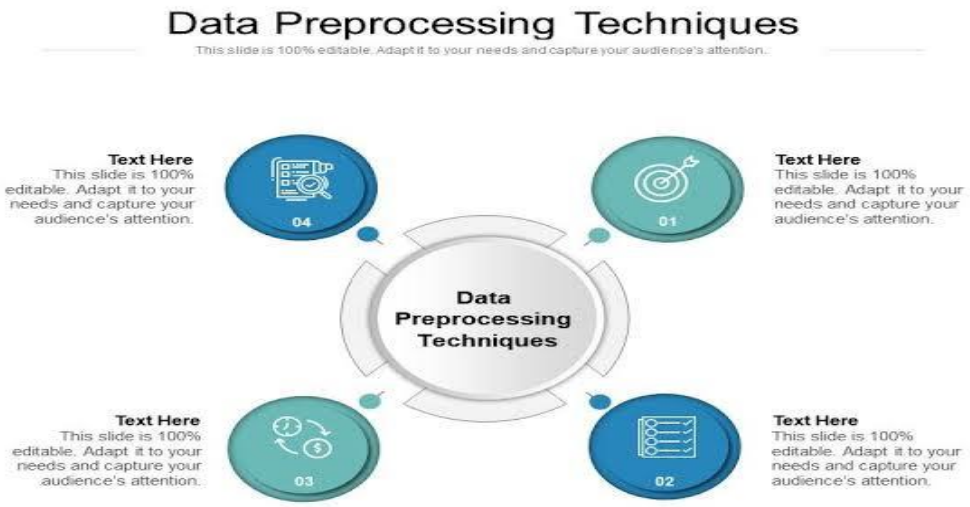
Converting Categorical data into dummy variables. Any variable that is not quantitative is categorical. Examples include Hair color, gender, field of study, college attended, political affiliation, status of disease infection.

But why encoding ?

We cannot use values like “Male” and “Female” in mathematical equations of the model so we need to encode these variables into numbers.

To do this we import “LabelEncoder” class from “sklearn.preprocessing” library and create an object labelencoder_X of the LabelEncoder class. After that we use the fittransform method on the categorical features.

After Encoding it is necessary to distinguish between between the variables in the same column, for this we will use OneHotEncoder class from sklearn.preprocessing library.



One-Hot Encoding

One hot encoding transforms categorical features to a format that works better with classification and regression algorithms. from sklearn.preprocessing import LabelEncoder,
OneHotEncoder labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:, 0])
onehotencoder = OneHotEncoder(categorical_features = [0])
X = onehotencoder.fit_transform(X).
toarray()labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)

Step 5: Splitting the Data set into Training set and Test Set

Now we divide our data into two sets, one for training our model called the **training set** and the other for testing the performance of our model called the **test set**

. The split is generally 80/20. To do this we import the "train_test_split" method of "sklearn.model_selection" library.

```
from sklearn.model_selection import train_test_split
```

Now to build our training and test sets, we will create 4 sets —

1. **X_train** (training part of the matrix of features),
2. **X_test** (test part of the matrix of features),
3. **Y_train** (training part of the dependent variables associated with the X train sets, and therefore also the same indices) ,
4. **Y_test** (test part of the dependent variables associated with the X test sets, and therefore also the same indices).

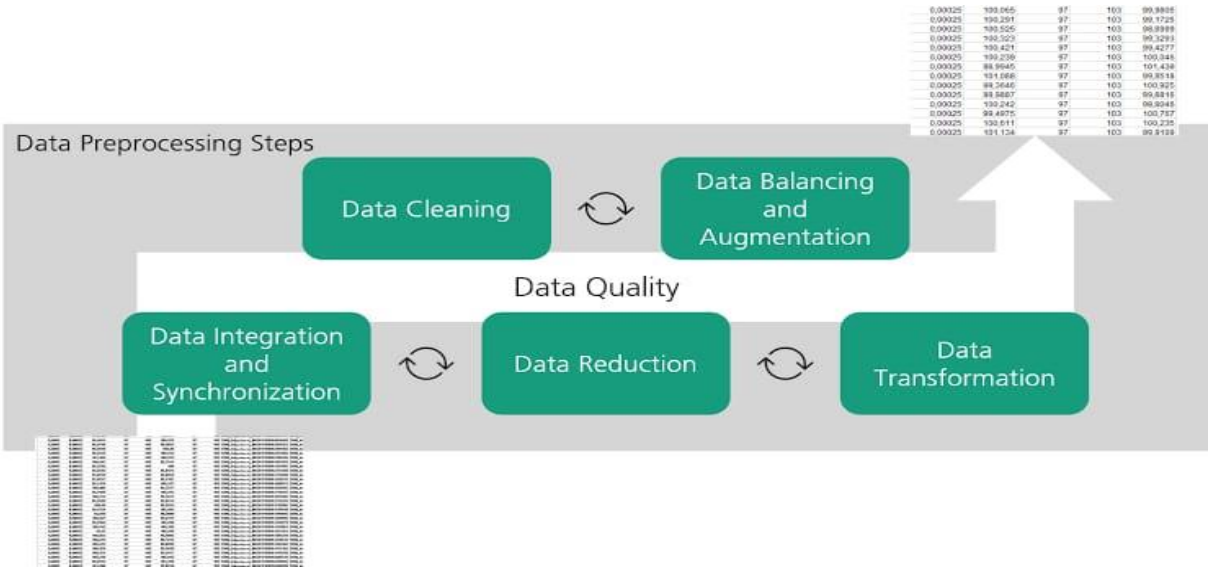
We will assign to them the test_train_split, which takes the parameters — arrays (X and Y), test_size (Specifies the ratio in which to split the data set).

```
X_train, X_test, Y_train, Y_test = train_test_split( X , Y , test_size = 0.2, random_state = 0)
```

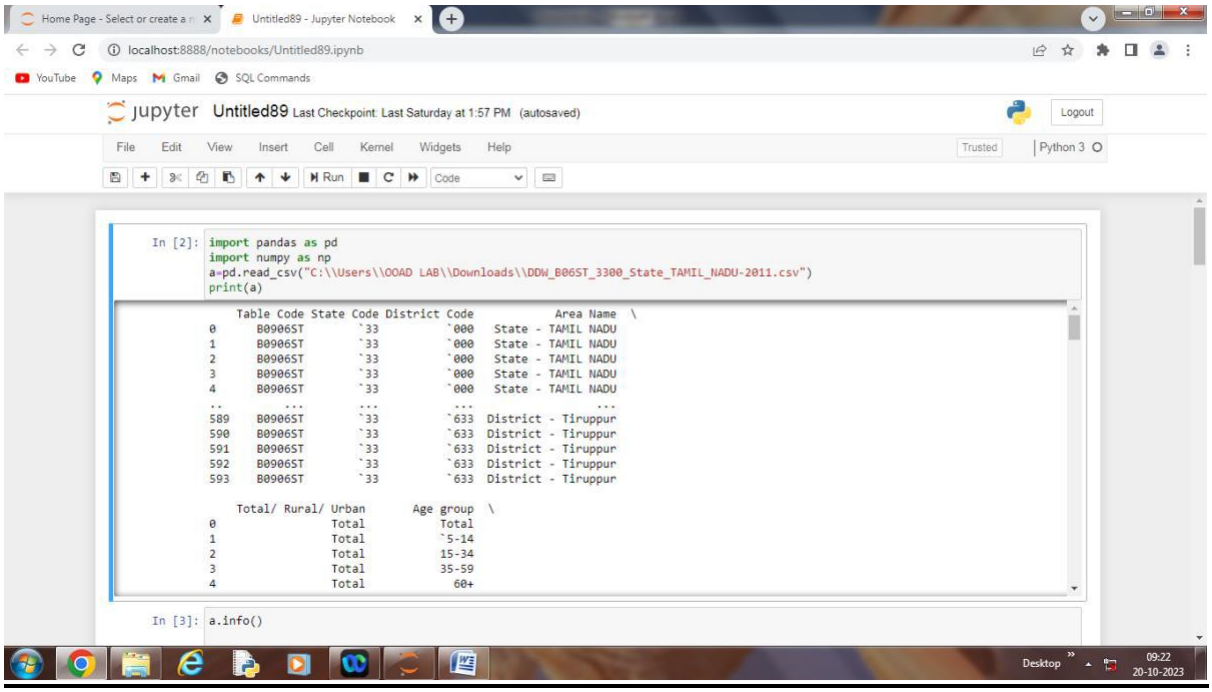
Step 6: Feature Scaling

Most of the machine learning algorithms use the **Euclidean distance** between two data points in their computations . Because of this, **high magnitudes features will weigh more** in the distance calculations **than features with low magnitudes**. To avoid this Feature standardization or Z-score normalization is used. This is done by using "StandardScaler" class of "sklearn.preprocessing".

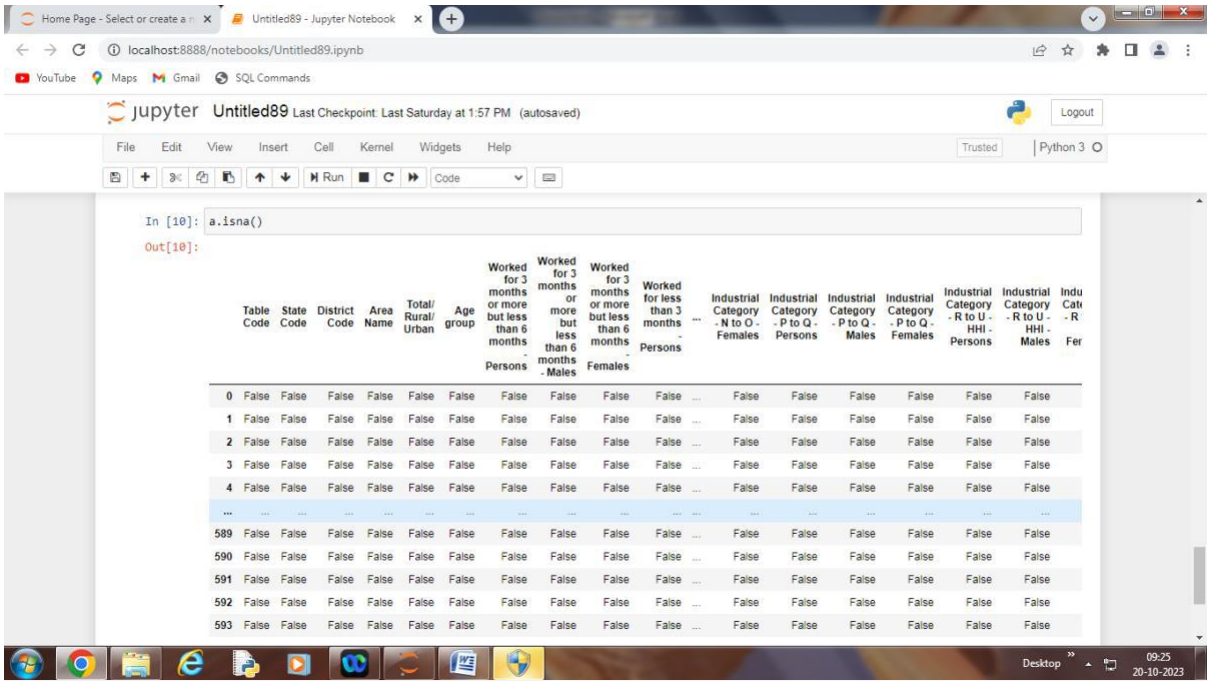
```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
```



➢ LOADING A DATASETS



➢ *PREPROCESSING THE DATASETS*

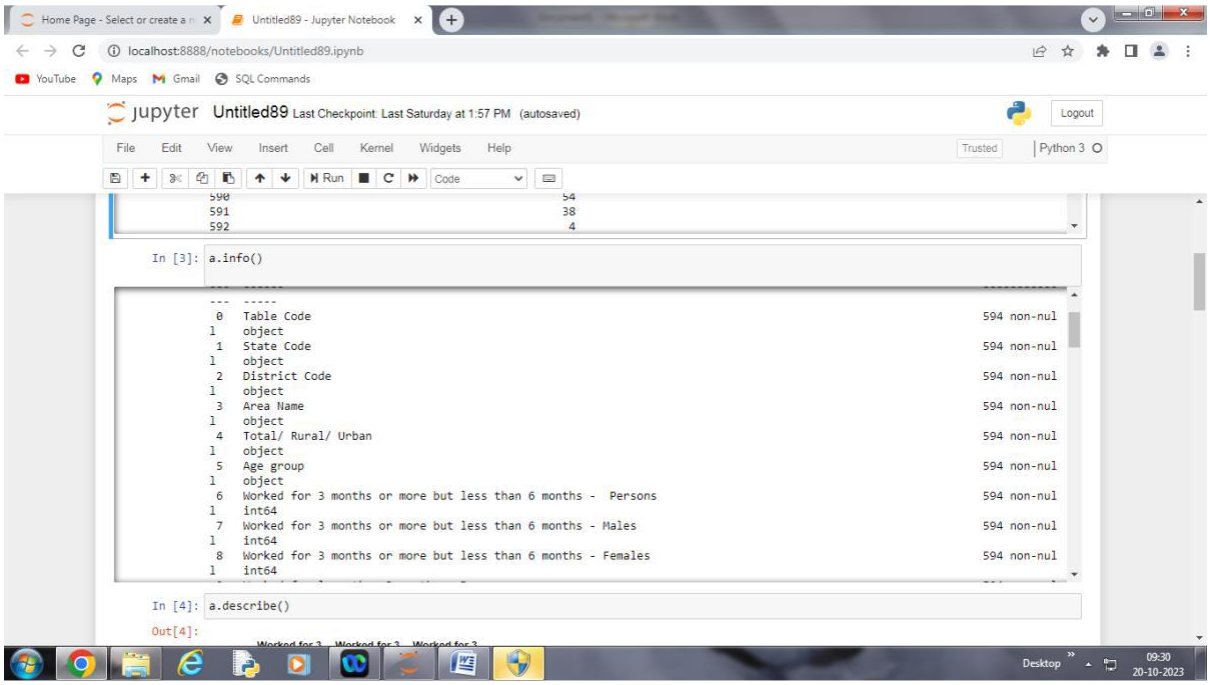


```
In [10]: a.isna()
```

Out[10]:

	Table Code	State Code	District Code	Area Name	Total/ Rural/ Urban	Age group	Worked for 3 months or more but less than 6 months - Persons	Worked for 3 months or more but less than 6 months - Males	Worked for 3 months or more but less than 6 months - Females	Worked for less than 3 months - Persons	...	Industrial Category - N to O - Females	Industrial Category - P to Q - Persons	Industrial Category - P to Q - Males	Industrial Category - P to Q - Females	Industrial Category - R to U - HHI - Persons	Industrial Category - R to U - HHI - Males	Indu Cate - R Fer
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
589	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
590	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
591	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
592	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	
593	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	

➢ *PREFROMING THE DIFFERENT ANALYSIS*



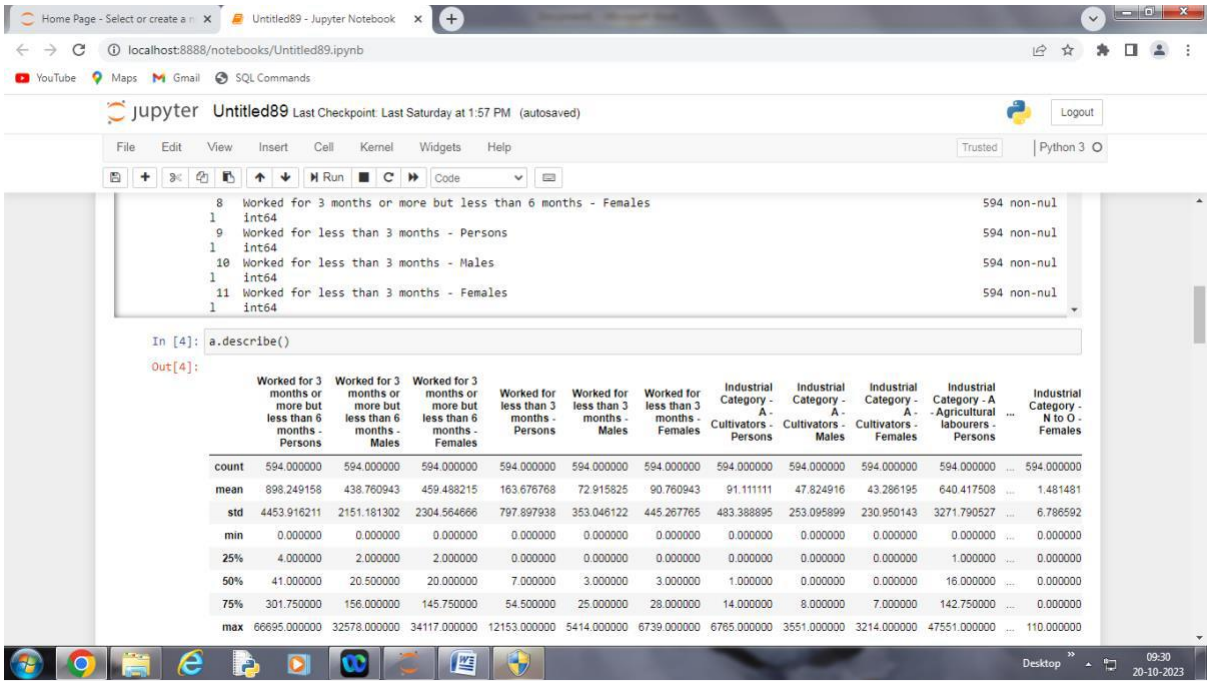
```
590    54
591    38
592     4

In [3]: a.info()

 0   Table Code                                                                  594 non-nul
l    object
 1   State Code                                                                  594 non-nul
l    object
 2   District Code                                                               594 non-nul
l    object
 3   Area Name                                                                   594 non-nul
l    object
 4   Total/ Rural/ Urban                                                         594 non-nul
l    object
 5   Age group                                                                   594 non-nul
l    object
 6   Worked for 3 months or more but less than 6 months -  Persons               594 non-nul
l    int64
 7   Worked for 3 months or more but less than 6 months - Males                  594 non-nul
l    int64
 8   Worked for 3 months or more but less than 6 months - Females                594 non-nul
l    int64

In [4]: a.describe()
```

Out[4]:



	Worked for 3 months or more but less than 6 months - Persons	Worked for 3 months or more but less than 6 months - Males	Worked for 3 months or more but less than 6 months - Females	Worked for less than 3 months - Persons	Worked for less than 3 months - Males	Worked for less than 3 months - Females	Industrial Category - A - Cultivators - Persons	Industrial Category - A - Cultivators - Males	Industrial Category - A - Cultivators - Females	Industrial Category - A - Agricultural labourers - Persons	...	Industrial Category - N to O - Females
count	594.000000	594.000000	594.000000	594.000000	594.000000	594.000000	594.000000	594.000000	594.000000	594.000000	...	594.000000
mean	898.249158	438.760943	459.488215	163.676768	72.915825	90.760943	91.111111	47.824916	43.286195	640.417508	...	1.481481
std	4453.916211	2151.181302	2304.564666	797.897938	353.046122	445.267765	483.388895	253.095899	230.950143	3271.790527	...	6.786592
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000
25%	4.000000	2.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	...	0.000000
50%	41.000000	20.500000	20.000000	7.000000	3.000000	3.000000	1.000000	0.000000	0.000000	16.000000	...	0.000000
75%	301.750000	156.000000	145.750000	54.500000	25.000000	28.000000	14.000000	8.000000	7.000000	142.750000	...	0.000000
max	66695.000000	32578.000000	34117.000000	12153.000000	5414.000000	6739.000000	6765.000000	3551.000000	3214.000000	47551.000000	...	110.000000

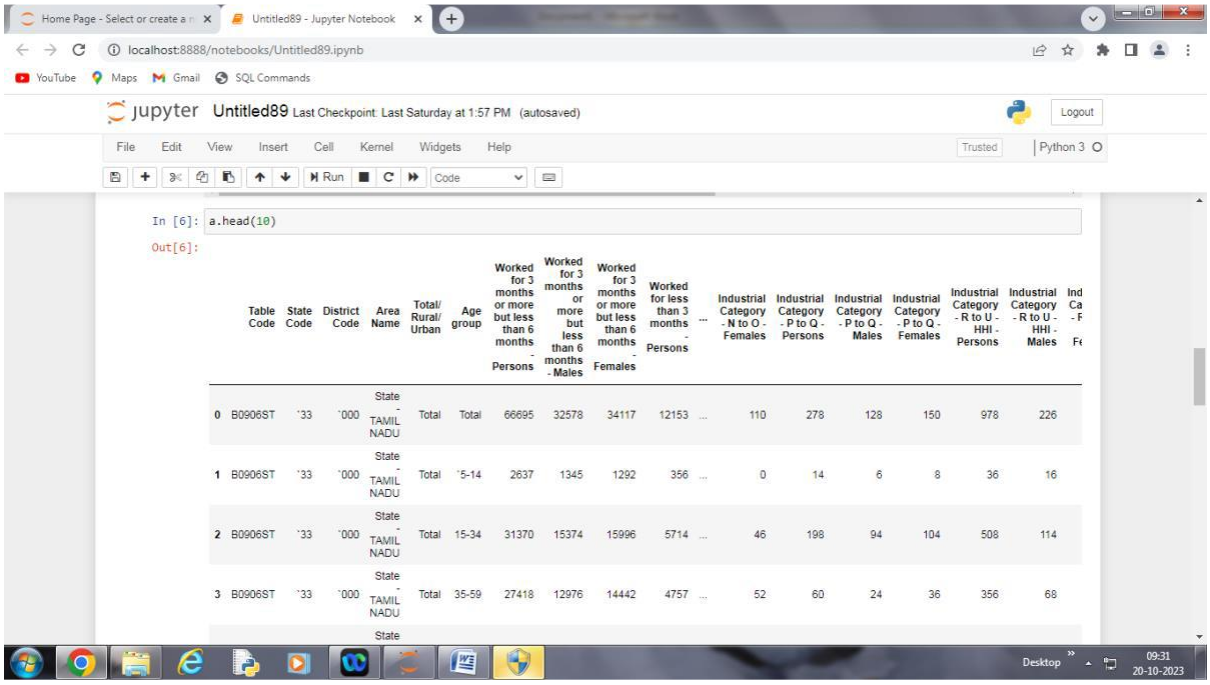
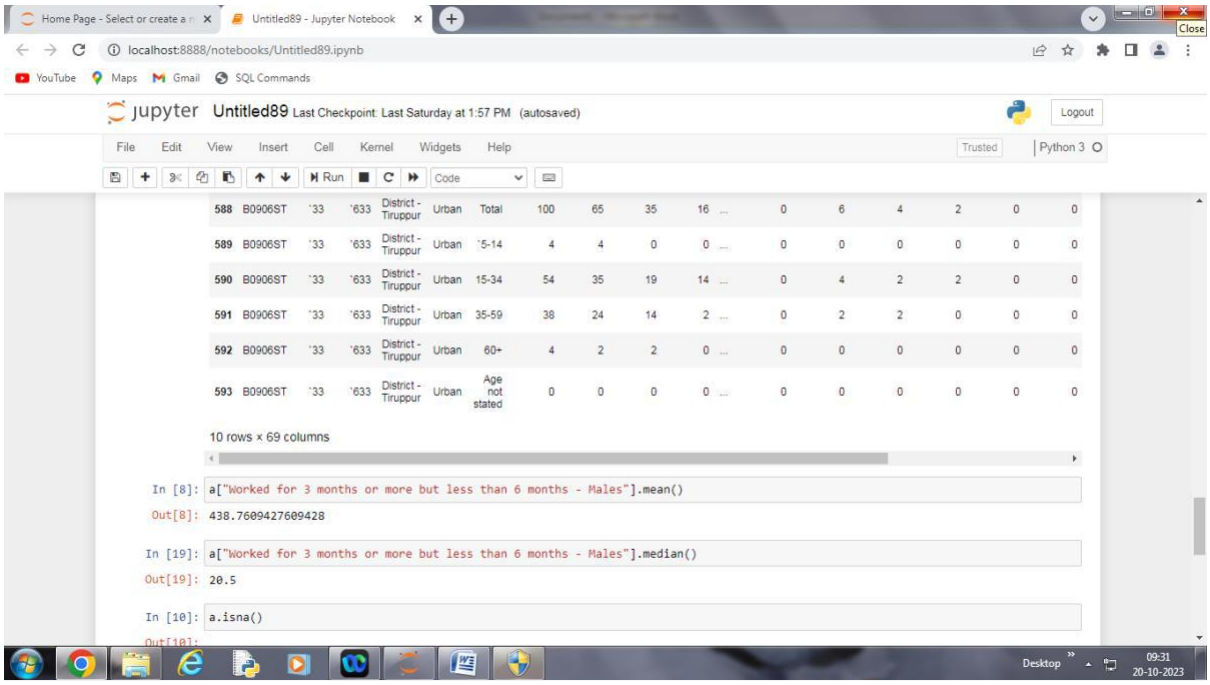


	Table Code	State Code	District Code	Area Name	Total/ Rural/ Urban	Age group	Worked for 3 months or more but less than 6 months - Persons	Worked for 3 months or more but less than 6 months - Males	Worked for 3 months or more but less than 6 months - Females	Worked for less than 3 months - Persons	...	Industrial Category - N to O - Females	Industrial Category - P to Q - Persons	Industrial Category - P to Q - Males	Industrial Category - P to Q - Females	Industrial Category - R to U - HHI - Persons	Industrial Category - R to U - HHI - Males	Ind Ca - F Fe
0	B0906ST	'33	'000	State - TAMIL NADU	Total	Total	66695	32578	34117	12153	...	110	278	128	150	978	226	
1	B0906ST	'33	'000	State - TAMIL NADU	Total	'5-14	2637	1345	1292	356	...	0	14	6	8	36	16	
2	B0906ST	'33	'000	State - TAMIL NADU	Total	15-34	31370	15374	15996	5714	...	46	198	94	104	508	114	
3	B0906ST	'33	'000	State - TAMIL NADU	Total	35-59	27418	12976	14442	4757	...	52	60	24	36	356	68	
				State														



FEATURE ENGINEERING

Feature engineering refers to the process of using domain knowledge to select and transform the most relevant variables from raw data when creating a predictive model using machine learning or statistical modeling.

What is Feature Engineering?

```
# Number of contractions (can't, won't, don't, haven't, etc.) in text
import re

def contraction_count(sent):
    count = 0
    count += re.subn(r"won\'t", '', sent)[1]
    count += re.subn(r"can\'t", '', sent)[1]
    count += re.subn(r"n\'t", '', sent)[1]
    count += re.subn(r"\'re", '', sent)[1]
    count += re.subn(r"\'s", '', sent)[1]
    count += re.subn(r"\'d", '', sent)[1]
    count += re.subn(r"\'ll", '', sent)[1]
    count += re.subn(r"\'t", '', sent)[1]
    count += re.subn(r"\'ve", '', sent)[1]
    count += re.subn(r"\'m", '', sent)[1]
    return count


df["excerpt_num_contractions"] = df["excerpt"].apply(contraction_count)
df[["excerpt","excerpt_num_contractions"]].head()
```

	excerpt	excerpt_num_contractions
2089	Alice looked at the jury-box, and saw that, in...	0
2806	Artificial intelligence (AI) is intelligence e...	0
1146	A gruff squire on horseback with shiny top boo...	0
1110	But that hadn't helped Washington.\nThe Americ...	2
196	The principal business of the people of this c...	0

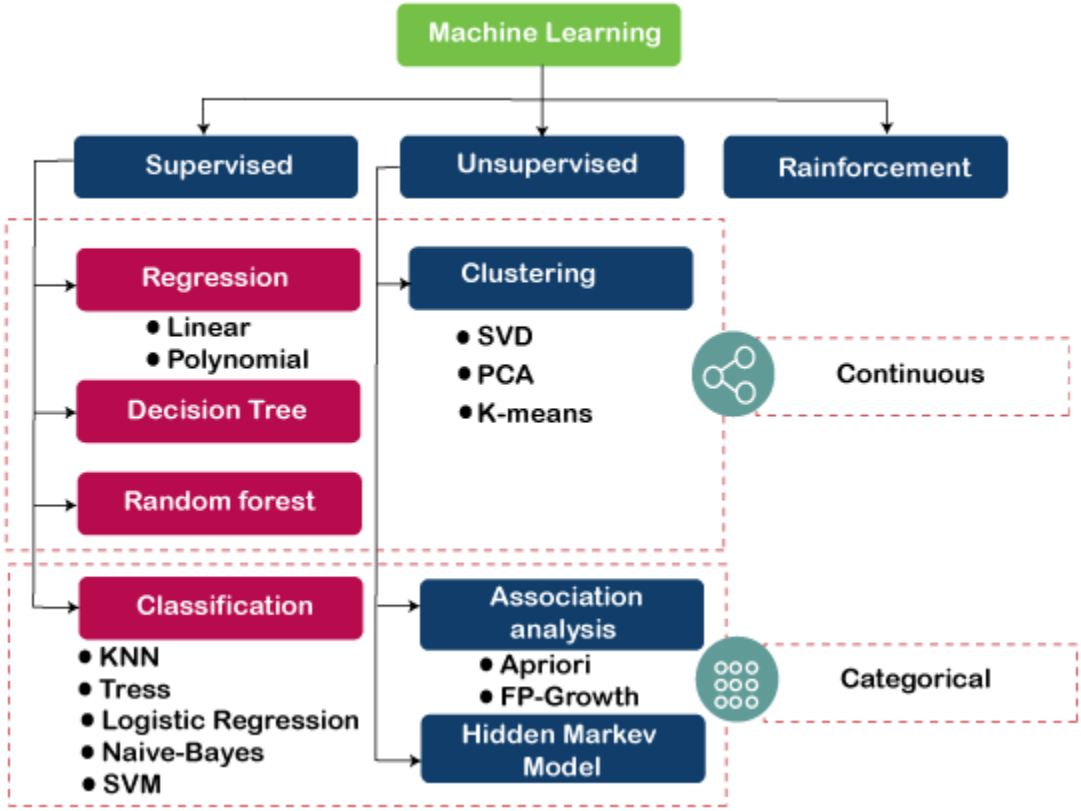
MACHINE LEARNING ALGORITHM:

Machine Learning algorithms are the programs that can learn the hidden patterns from the data, predict the output, and improve the performance from experiences on their own. Different algorithms can be used in machine learning for different tasks, such as simple linear regression that can be used **for prediction problem**s like **stock market prediction,** and **the KNN algorithm can be used for classification problems**.

TYPES OF MACHINE LEARNING

- **Supervised Learning Algorithms**
- **Unsupervised Learning Algorithms**

- **Reinforcement Learning algorithm**



List of Popular Machine Learning Algorithm

- **Linear Regression Algorithm**
- **Logistic Regression Algorithm**
- **Decision Tree**
- **SVM**
- **Naïve Bayes**
- **KNN**
- **K-Means Clustering**
- **Random Forest**
- **Apriori**
- **PCA**

1. Linear Regression

Linear regression is one of the most popular and simple machine learning algorithms that is used for predictive analysis. Here, **predictive analysis** defines prediction of something, and linear regression makes predictions for continuous numbers such as **salary, age, etc.**

It shows the linear relationship between the dependent and independent variables, and shows how the dependent variable(y) changes according to the independent variable (x).

It tries to best fit a line between the dependent and independent variables, and this best fit line is knowns as the regression line.

The equation for the regression line is:

$$y= a_0+ a*x+ b$$

Here, y= dependent variable

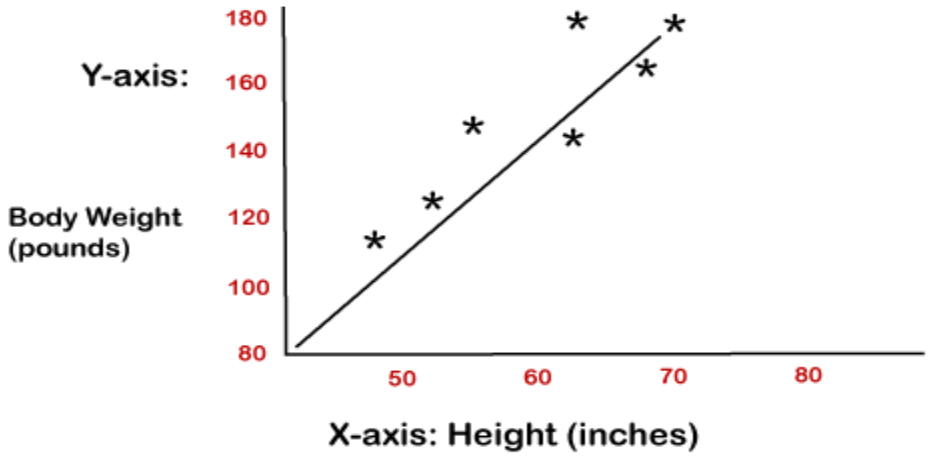
x= independent variable

a_0 = Intercept of line.

Linear regression is further divided into two types:

- **Simple Linear Regression:** In simple linear regression, a single independent variable is used to predict the value of the dependent variable.
- **Multiple Linear Regression:** In multiple linear regression, more than one independent variables are used to predict the value of the dependent variable.

The below diagram shows the linear regression for prediction of weight according to heig



2. Logistic Regression

Logistic regression is the supervised learning algorithm, which is used to **predict the categorical variables or discrete values**. It can be used for the classification problems in machine learning, and the output of the logistic regression algorithm can be either Yes or NO, 0 or 1, Red or Blue, etc.

Logistic regression is similar to the linear regression except how they are used, such as Linear regression is used to solve the regression problem and predict continuous values, whereas Logistic regression is used to solve the Classification problem and used to predict the discrete values.

Instead of fitting the best fit line, it forms an S-shaped curve that lies between 0 and 1. The S-shaped curve is also known as a logistic function that uses the concept of the threshold. Any value above the threshold will tend to 1, and below the threshold will tend to 0. Read more..

3. Decision Tree Algorithm

A decision tree is a supervised learning algorithm that is mainly used to solve the classification problems but can also be used for solving the regression problems. It can work with both categorical variables and continuous variables. It shows a tree-like structure that includes nodes and branches, and starts with the root node that expand on further branches till the leaf node. The **internal node** is used to represent the **features of the dataset, branches show the decision rules,** and **leaf nodes represent the outcome of the problem.**

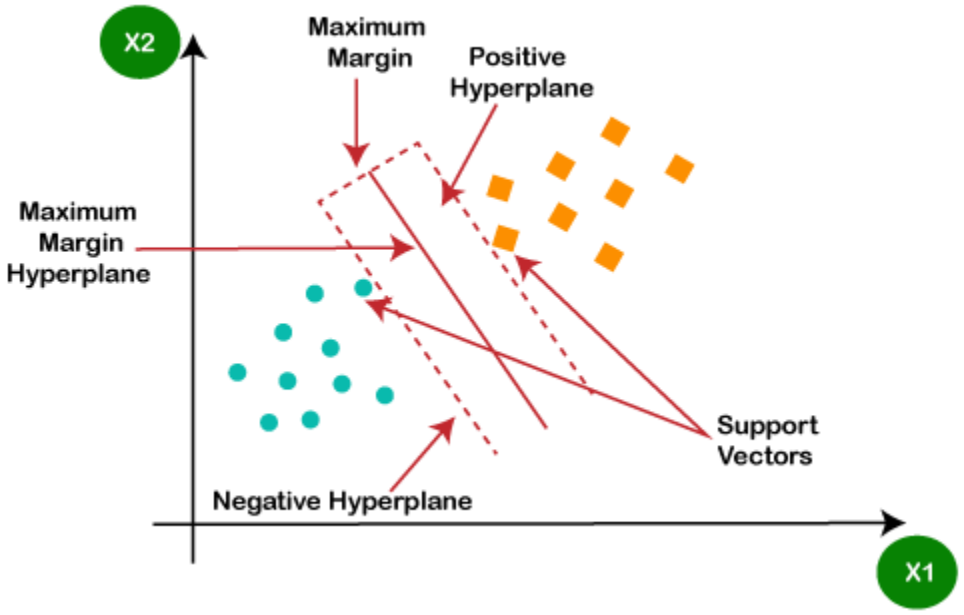
Some real-world applications of decision tree algorithms are identification between cancerous and non-cancerous cells, suggestions to customers to buy a car, etc. Read more..

4. Support Vector Machine Algorithm

A support vector machine or SVM is a supervised learning algorithm that can also be used for classification and regression problems. However, it is primarily used for classification problems. The goal of SVM is to create a hyperplane or decision boundary that can segregate datasets into different classes.

The data points that help to define the hyperplane are known as **support vectors**, and hence it is named as support vector machine algorithm.

Some real-life applications of SVM are **face detection, image classification, Drug discovery**, etc. Consider the below diagram:



As we can see in the above diagram, the hyperplane has classified datasets into two different classes. Read more..

5. Naïve Bayes Algorithm:

Naïve Bayes classifier is a supervised learning algorithm, which is used to make predictions based on the probability of the object. The algorithm named as Naïve Bayes as it is based on **Bayes theorem**, and follows the naïve assumption that says' variables are independent of each other.

The Bayes theorem is based on the conditional probability; it means the likelihood that event(A) will happen, when it is given that event(B) has already happened. The equation for Bayes theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Naïve Bayes classifier is one of the best classifiers that provide a good result for a given problem. It is easy to build a naïve bayesian model, and well suited for the huge amount of dataset. It is mostly used for **text classification**. Read more..

6. K-Nearest Neighbour (KNN)

K-Nearest Neighbour is a supervised learning algorithm that can be used for both classification and regression problems. This algorithm works by assuming the similarities between the new data point and available data points. Based on these similarities, the new data points are put in the most similar categories. It is also known as the lazy learner algorithm as it stores all the available datasets and classifies each new case with the help of K-neighbours. The new case is assigned to the nearest class with most similarities, and any distance function measures the distance between the data points. The distance function can be **Euclidean, Minkowski, Manhattan, or Hamming distance**, based on the requirement. Read more..

7. K-Means Clustering

K-means clustering is one of the simplest unsupervised learning algorithms, which is used to solve the clustering problems. The datasets are grouped into K different clusters based on similarities and dissimilarities, it means, datasets with most of the commonalties remain in one cluster which has very less or no commonalities between other clusters. In K-means, K-refers to the number of clusters, and **means** refer to the averaging the dataset in order to find the centroid.

It is a centroid-based algorithm, and each cluster is associated with a centroid. This algorithm aims to reduce the distance between the data points and their centroids within a cluster.

This algorithm starts with a group of randomly selected centroids that form the clusters at starting and then perform the iterative process to optimize these centroids' positions.

It can be used for spam detection and filtering, identification of fake news, etc. Read more..

8. Random Forest Algorithm

Random forest is the supervised learning algorithm that can be used for both classification and regression problems in machine learning. It is an ensemble learning technique that provides the predictions by combining the multiple classifiers and improve the performance of the model.

It contains multiple decision trees for subsets of the given dataset, and find the average to improve the predictive accuracy of the model. A random-forest should contain 64-128 trees. The greater number of trees leads to higher accuracy of the algorithm.

To classify a new dataset or object, each tree gives the classification result and based on the majority votes, the algorithm predicts the final output.

Random forest is a fast algorithm, and can efficiently deal with the missing & incorrect data. Read more..

9. Apriori Algorithm

Apriori algorithm is the unsupervised learning algorithm that is used to solve the association problems. It uses frequent itemsets to generate association rules, and it is designed to work on the databases that contain transactions. With the help of these association rule, it determines how strongly or how weakly two objects are connected to each other. This algorithm uses a breadth-first search and Hash Tree to calculate the itemset efficiently.

The algorithm process iteratively for finding the frequent itemsets from the large dataset.

The apriori algorithm was given by the **R. Agrawal and Srikant** in the year 1994. It is mainly used for market basket analysis and helps to understand the products that can be bought together. It can also be used in the healthcare field to find drug reactions in patients. Read more..

10. Principle Component Analysis

Principle Component Analysis (PCA) is an unsupervised learning technique, which is used for dimensionality reduction. It helps in reducing the dimensionality of the dataset that contains many features correlated with each other. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. It is one of the popular tools that is used for exploratory data analysis and predictive modeling.

PCA works by considering the variance of each attribute because the high variance shows the good split between the classes, and hence it reduces the dimensionality

WHAT IS MODEL TRAINING?

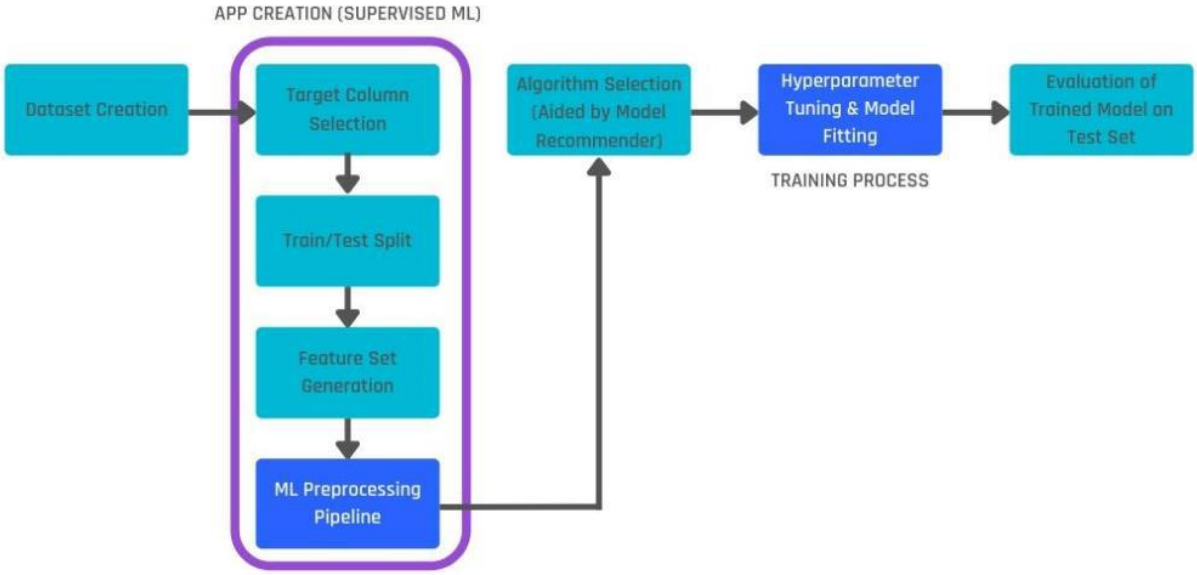
Model training is the phase in the data science development lifecycle where practitioners try to fit the best combination of weights and bias to a machine learning algorithm to minimize a loss function over the prediction range.

STEPS TO TRAINING MACHINE LEARNING MODEL

•Step 1: Begin with existing data. Machine learning requires us to have existing data—not the data our application will use when we run it, but data to learn from. ...

•Step 2: Analyze data to identify patterns. ...

•Step 3: Make predictions



In [10]:
```
# prepare data frame for splitting data into train and test datasets

features = []
features = df_churn_pd.drop(['CHURNRISK'], axis=1)

label_churn = pd.DataFrame(df_churn_pd, columns = ['CHURNRISK'])
label_encoder = LabelEncoder()
label = df_churn_pd['CHURNRISK']

label = label_encoder.fit_transform(label)
print("Encoded value of Churnrisk after applying label encoder : " + str(label))
```

```
Encoded value of Churnrisk after applying label encoder : [2 1 1 ... 2 1 1]
```

MODEL EVALUATION

Model evaluation is a crucial aspect of machine learning, allowing us to assess how well our models perform on unseen data. In this step-by-step guide, we will explore the process of model evaluation using Python. By following these steps and leveraging Python's powerful libraries, you'll gain valuable insights into your model's performance and be able to make informed decisions. Let's dive in and evaluate our machine learning models!

Step 1: Prepare the Data

The first step in model evaluation is to prepare your data. Split your dataset into training and test sets using the train_test_split function from the scikit-learn library. This ensures that we have separate data for training and evaluating our model.

```
from sklearn.model_selection import train_test_split
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Step 2: Train the Model

Next, select an appropriate model for your task and train it using the training set. For example, let's train a logistic regression model using scikit-learn:

```
from sklearn.linear_model import LogisticRegression
# Create an instance of the model
model = LogisticRegression()
# Train the model
model.fit(X_train, y_train)
```

Step 3: Evaluate on the Test Set

Now, it's time to evaluate our model on the test set. Use the trained model to make predictions on the test data and compare them to the actual labels. Calculate evaluation metrics such as accuracy_score to measure the model's performance.

```
from sklearn.metrics import accuracy_score
# Make predictions on the test set
y_pred = model.predict(X_test)
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Step 4: Perform Cross-Validation (Optional)

To obtain a more robust evaluation, you can perform cross-validation. This technique involves splitting the data into multiple folds and training/evaluating the model on different combinations. Here's an example using cross_val_score from scikit-learn:

```
from sklearn.model_selection import cross_val_score
# Perform cross-validation
scores = cross_val_score(model, X, y, cv=5)
# Calculate the average performance across all folds
```

```
mean_accuracy = scores.mean()
print("Mean Accuracy:", mean_accuracy)
```

Step 5: Assess Model's Performance

Analyze the evaluation metrics obtained from the previous steps to assess the model's performance. Consider the context of your problem and compare the results against your desired performance level or any baseline models. This analysis will provide insights into the strengths and weaknesses of your model.

Step 6: Iterate and Improve (if needed)

Based on the assessment, you may need to iterate and improve your model. Consider collecting more data, refining features, trying different algorithms, or tuning hyperparameters. Repeat the evaluation process until you achieve the desired performance.