DOOARS ACADEMY OF TECHNOLOGY & MANAGEMENT

“The Project” project i.e. Online Food Order Delivery Prediction using Machine Learning in python.

‘I’ i.e. each individual members of the project who has given their effort to make the project a successful one. So I am thankful to our members who make the project a grand success their participation in project. And also we all the team members is thankful to our mentor Mr. Akash Dhar , Head of the Bachelors’ Of Computer Application Mr. Sanjib Kumar Dutta , and other faculties of the BCA department their guidance to make our project a grand success we learn the leadership styles and traits throughout the entire project .

Our Team Members whose Contribution in the entire project is negotiable. The member of the list is given below.

Signature of the members			
slno	Name	Roll	Signatures
1	Soumyadeep Saha Chaudhuri	19901220001	
2	Saikat Dhar	19901220003	
3	Anamika Sarkar	19901220004	
4	Srija Bhattacharjee	19901220005	
5	Taniya Nandi	19901220006	
6	Joydeb Barai	19901220007	
7	Uday Dey	19901220008	
8	Arijit Roy	19901220009	
9	Shubhankar Paul	19901220010	
10	Aritri Mitra	19901220011	

Thank you to the all members of your contribution

Dooars Academy of Technology &Management

Abstract

ONLINE FOOD ORDER DELIVERY PREDICTION USING MACHINE LEARNING (PYTHON) is a python machine learning project for use in the food delivery and other CRM related industry. This system will allow the ecommerce based (B2C) to increase scope of business by reducing the labor cost involved. Interface for efficient processing. Our proposed system is an online food order delivery market predicting and customer review system that enables ease for the customers. It overcomes the disadvantages of the organization or we can say it helps to S.W.O.T analysis of any company .

The system's functional requirements (FR) and non-functional requirements (NFR) are derived from the software requirements specification (SRS). The requirement specification is challenging in classification process of FR and NFR requirements. To overcome these issues, the work contains various significant contributions towards SRS. . In addition to this, the test pad-based quality study to determine accuracy, quality, and condition providence to the classification of non-functional requirements (NFR) is also carried out. The resulted classification accuracy was implemented in the jupyter notebook by the help of Anaconda Environment.

Acknowledgement

We would like to express our profound gratitude to **Mr. Sanjib Kumar Dutta**(Head of the Department) and other faculties, of **Bachelor's Of Computer Application** (BCA) department, and **Mr. Sumit Bhattacharya (Director)**of **Dooars Academy Of Technology &Management** Affiliated **MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY** (M.K.A.U.T) for their contributions to the completion of our major project in the final semester .

project titled " **ONLINE FOOD ORDER DELIVERY PREDICTION USING MACHINE LEARNING (PYTHON)**".

We would like to express our special thanks to our mentor **Mr. Akash Dhar** for his time and efforts he provided throughout the year. Your useful advice and suggestions were really helpful to us during the project's completion. In this aspect, We are eternally grateful to you.

Thank you

I N D E X

INTRODUCTION

Pandas Library:

What is Pandas?

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.

Why Use Pandas?

Pandas allows us to analyze big data and make conclusions based on statistical theories.

Pandas can clean messy data sets, and make them readable and relevant.

Relevant data is very important in data science.

What Can Pandas Do?

Pandas gives you answers about the data. Like:

- Is there a correlation between two or more columns?
- What is average value?
- Max value?
- Min value?

Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called *cleaning* the data.

Read CSV Files

A simple way to store big data sets is to use CSV files (comma separated files).

CSV files contains plain text and is a well know format that can be read by everyone including Pandas.

In our examples we will be using a CSV file called 'FinalResponse.csv'.

Example:

```
import pandas as pd

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

print(df.to_string())
```

What is a Data Frame?

A Pandas DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

Example

Create a simple Pandas DataFrame:

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)
```

Before using the response in our machine learning we have to processed the response (CSV File) i.e. Data Cleaning which is used in the field of data mining by the help of pandas library.

Data Cleaning

Data cleaning means fixing bad data in our data set.

Bad data could be:

- Empty cells
- Data in wrong format
- Wrong data
- Duplicates

Empty Cells

Empty cells can potentially give you a wrong result when you analyze data.

Remove Rows

One way to deal with empty cells is to remove rows that contain empty cells.

This is usually OK, since data sets can be very big, and removing a few rows will not have a big impact on the result.

Example

Return a new Data Frame with no empty cells:

```
import pandas as pd

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

new_df = df.dropna()

print(new_df.to_string())
```

If you want to change the original DataFrame, use the `inplace = True` argument:

Example

Remove all rows with NULL values:

```
import pandas as pd

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

df.dropna(inplace = True)

print(df.to_string())
```

Data of Wrong Format

Cells with data of wrong format can make it difficult, or even impossible, to analyze data.

To fix it, you have two options: remove the rows, or convert all cells in the columns into the same format.

By taking a look at our test data set, we can assume that row 11 and 12 are duplicates.

To discover duplicates, we can use the `duplicated()` method.

The `duplicated()` method returns a Boolean values for each row:

Example

Returns `True` for every row that is a duplicate, othwerwise `False`:

```
print(df.duplicated())
```

Pandas - Data Correlations

Finding Relationships

A great aspect of the Pandas module is the `corr()` method.

The `corr()` method calculates the relationship between each column in your data set.

The examples in this page uses a CSV file called: 'FinalResponse.csv'.

Example

Show the relationship between the columns:

```
df.corr()
```

Result Explained

The Result of the `corr()` method is a table with a lot of numbers that represents how well the relationship is between two columns.

The number varies from -1 to 1.

1 means that there is a 1 to 1 relationship (a perfect correlation), and for this data set, each time a value went up in the first column, the other one went up as well.

0.9 is also a good relationship, and if you increase one value, the other will probably increase as well.

-0.9 would be just as good relationship as 0.9, but if you increase one value, the other will probably go down.

0.2 means NOT a good relationship, meaning that if one value goes up does not mean that the other will.

What is a good correlation? It depends on the use, but I think it is safe to say you have to have at least `0.6` (or `-0.6`) to call it a good correlation.

Perfect Correlation:

We can see that "Age and Age" and "Member and Member" got the number `1.000000`, which makes sense, each column always has a perfect relationship with itself.

Good Correlation:

"Age" and "Member" got a very good correlation, and we can predict that the longer you work out, the more calories you burn, and the other way around: if you burned a lot of calories, you probably had a long work out.

Bad Correlation:

It means that we cannot predict the max pulse by just looking at the duration of the work out, and vice versa.

Pandas Plotting:

Pandas uses the `plot()` method to create diagrams.

Pyplot

Most of the Matplotlib utilities lies under the `pyplot` submodule, and are usually imported under the `plt` alias:

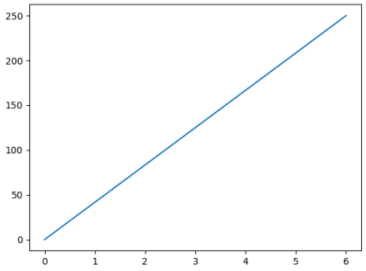
```
import matplotlib.pyplot as plt
```

EXAMPLE

Draw a line in a diagram from position (0,0) to position (6,250):

```
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([0, 6])
ypoints = np.array([0, 250])

plt.plot(xpoints, ypoints)
plt.show()
```

Result:



Plotting x and y points

The `plot()` function is used to draw points (markers) in a diagram.

By default, the `plot()` function draws a line from point to point.

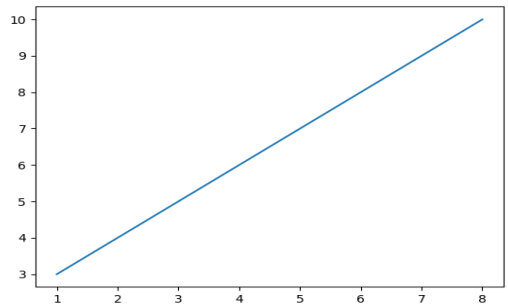
The function takes parameters for specifying points in the diagram.

Parameter 1 is an array containing the points on the **x-axis**.

Parameter 2 is an array containing the points on the **y-axis**.

If we need to plot a line from (1, 3) to (8, 10), we have to pass two arrays [1, 8] and [3, 10] to the plot fun

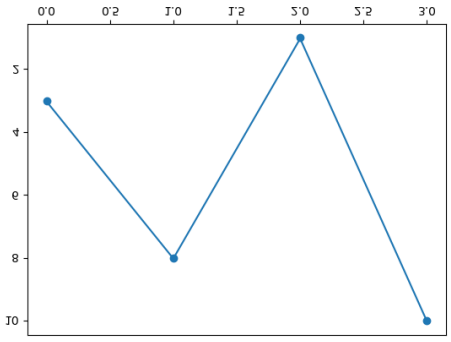
```
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([1, 8])
ypoints = np.array([3, 10])

plt.plot(xpoints, ypoints)
plt.show()
```



Markers

You can use the keyword argument `marker` to emphasize each point with a specified marker:

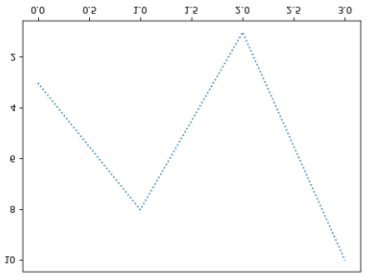
```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, marker = 'o')
plt.show()
```



Linestyle

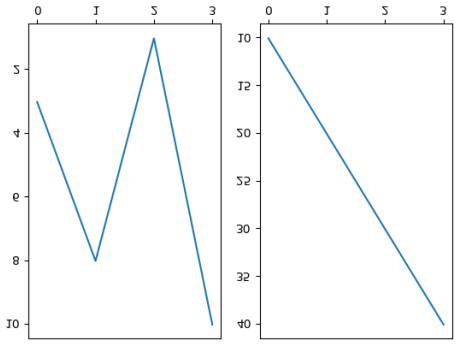
You can use the keyword argument `linestyle`, or shorter `ls`, to change the style of the plotted line:

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, linestyle = 'dotted')
plt.show()
```



Subplot

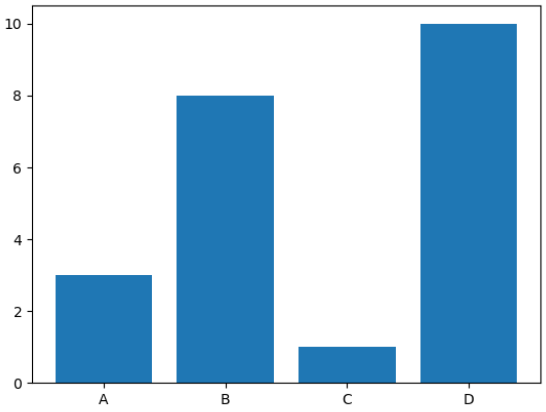
```
import matplotlib.pyplot as plt
import numpy as np

#plot 1:
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

plt.subplot(1, 2, 1)
plt.plot(x,y)

#plot 2:
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(1, 2, 2)
plt.plot(x,y)

plt.show()
```



Creating Bars

With Pyplot, you can use the `bar()` function to draw bar graphs:

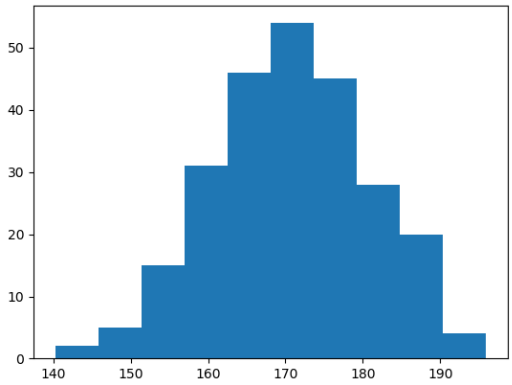
```
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.bar(x,y)
plt.show()
```



Histogram

A histogram is a graph showing *frequency* distributions.

It is a graph showing the number of observations within each given interval.



Histogram Explained

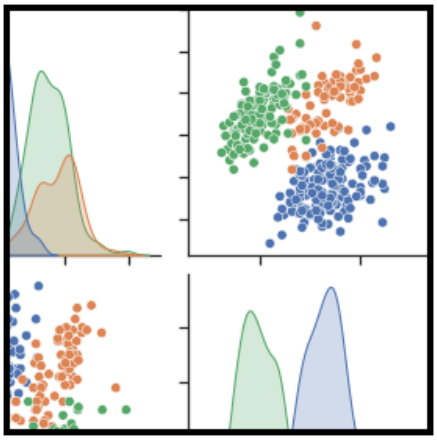
We use the array from the `numpy.random.normal()` method, with 100000 values, to draw a histogram with 100 bars.

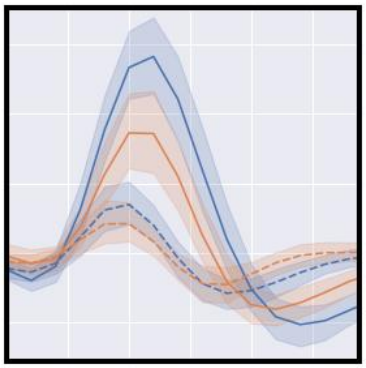
We specify that the mean value is 5.0, and the standard deviation is 1.0.

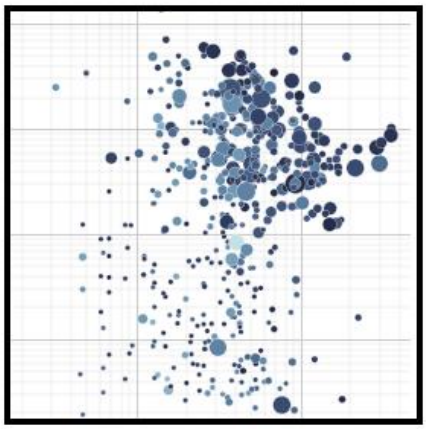
Meaning that the values should be concentrated around 5.0, and rarely further away than 1.0 from the mean.

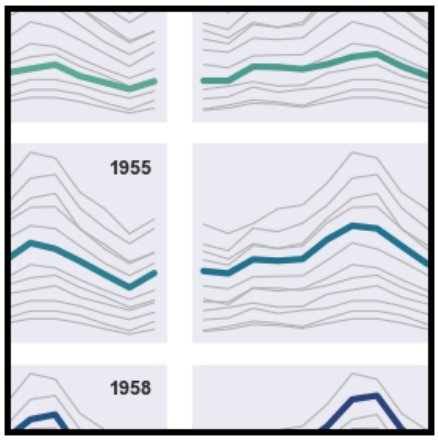
And as you can see from the histogram, most values are between 4.0 and 6.0, with a top at approximately 5.0.

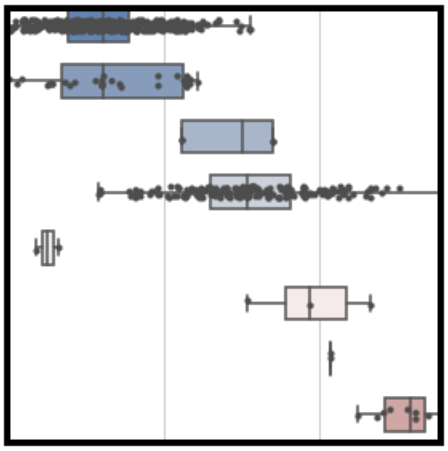
seaborn: statistical data visualization

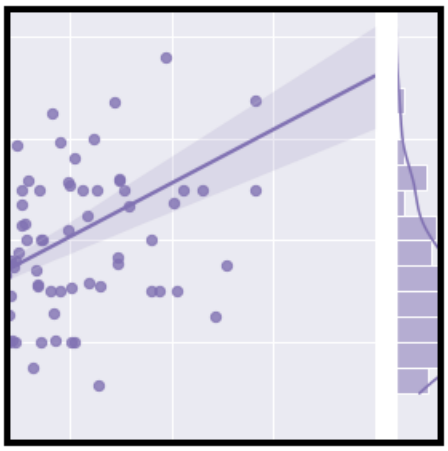







1955
1958





Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Overview of seaborn plotting functions

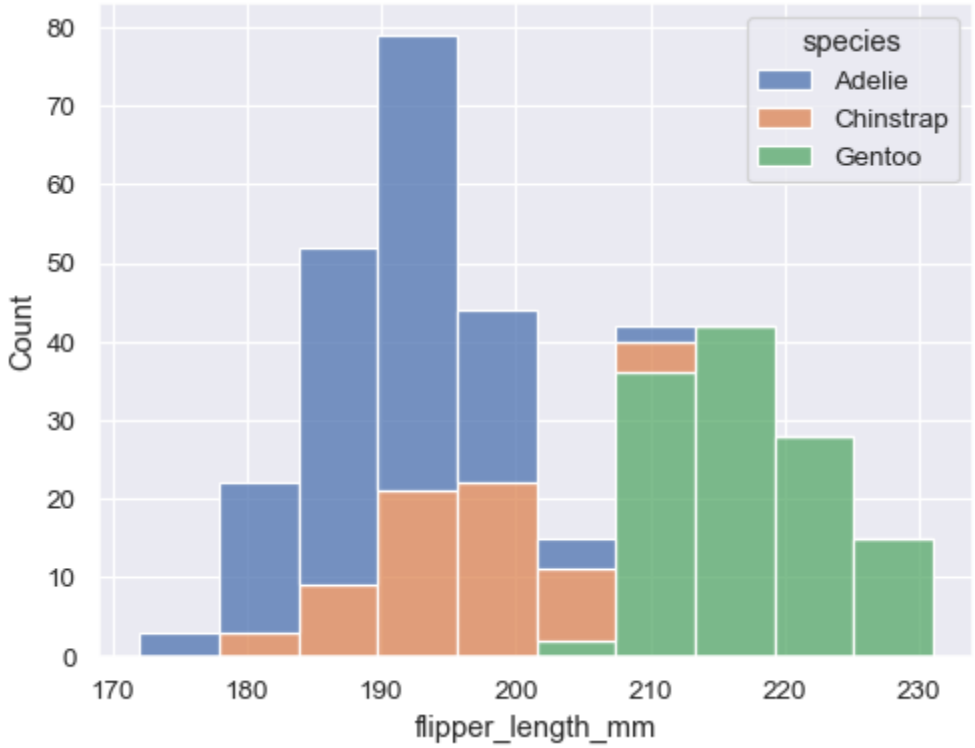
Most of your interactions with seaborn will happen through a set of plotting functions. Later chapters in the tutorial will explore the specific features offered by each function. This chapter will introduce, at a high-level, the different kinds of functions that you will encounter.

Similar functions for similar tasks

The seaborn namespace is flat; all of the functionality is accessible at the top level. But the code itself is hierarchically structured, with modules of functions that achieve similar visualization goals through different means. Most of the docs are structured around these modules: you'll encounter names like "relational", "distributional", and "categorical".

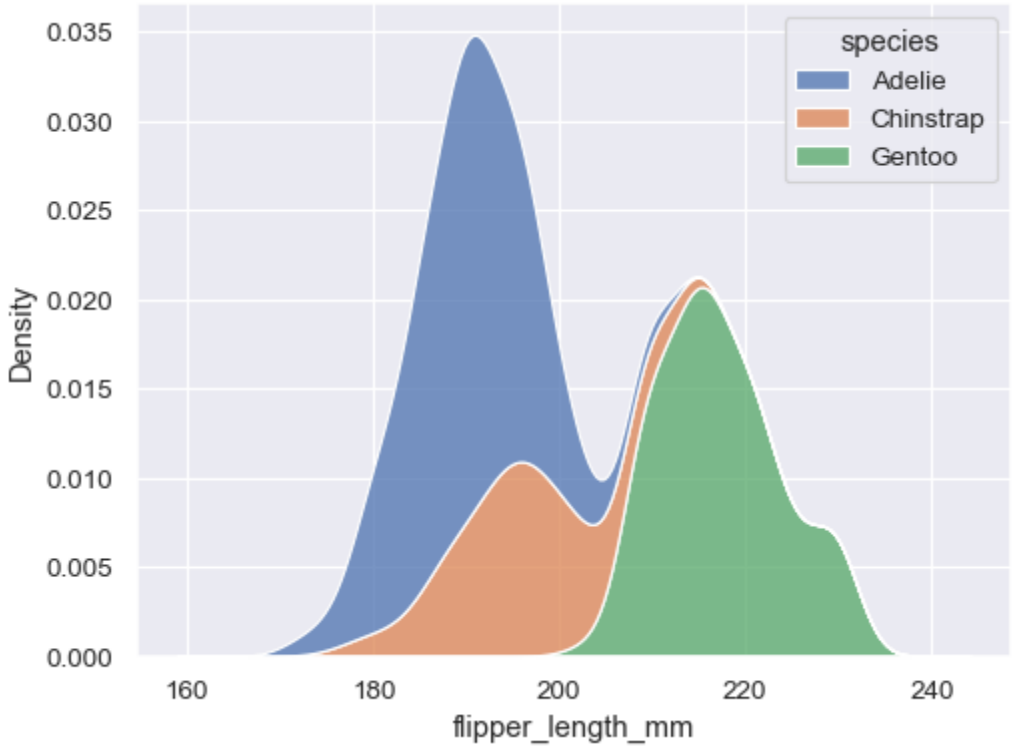
For example, the distributions module defines functions that specialize in representing the distribution of datapoints. This includes familiar methods like the histogram:

```
penguins = sns.load_dataset("penguins")
sns.histplot(data=penguins, x="flipper_length_mm", hue="species",
multiple="stack")
```



Along with similar, but perhaps less familiar, options such as kernel density estimation:

```
sns.kdeplot(data=penguins, x="flipper_length_mm", hue="species",
multiple="stack")
```

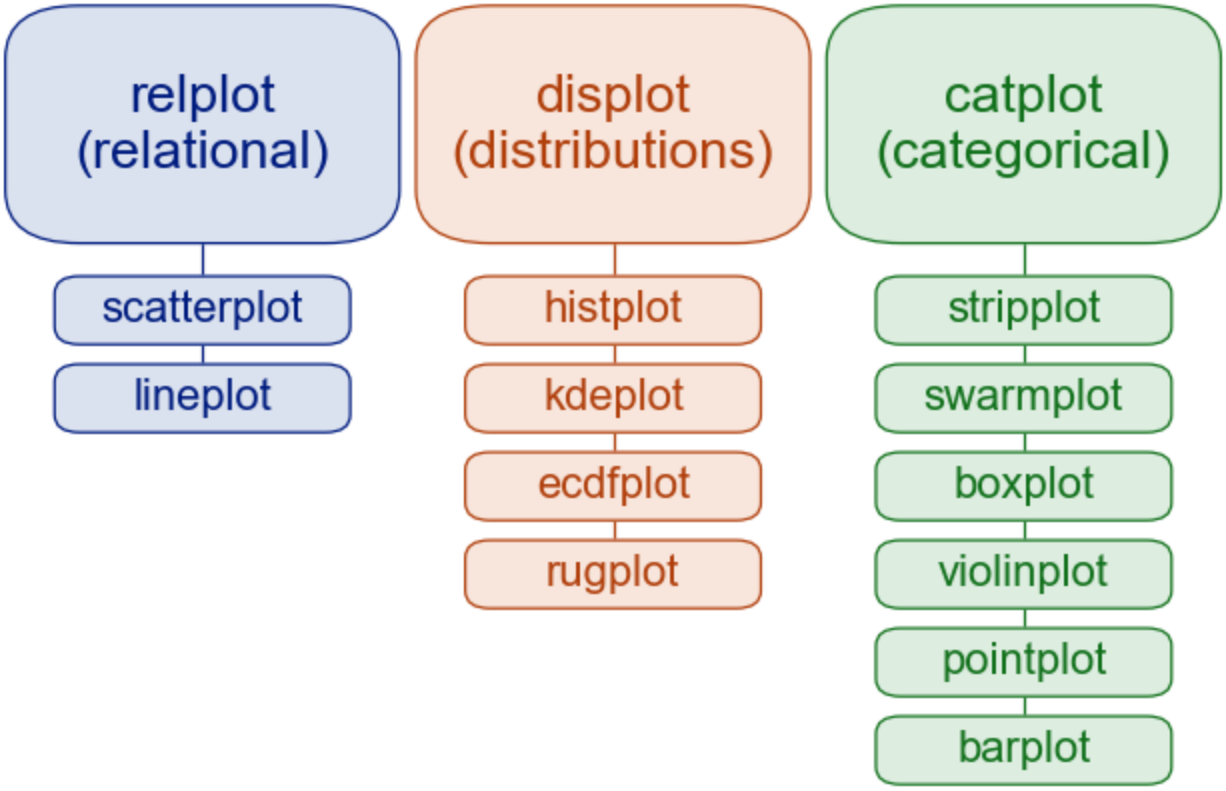


Functions within a module share a lot of underlying code and offer similar features that may not be present in other components of the library (such as multiple="stack" in the examples above). They are designed to facilitate switching between different visual representations as you explore a dataset, because different representations often have complementary strengths and weaknesses.

Figure-level vs. axes-level functions

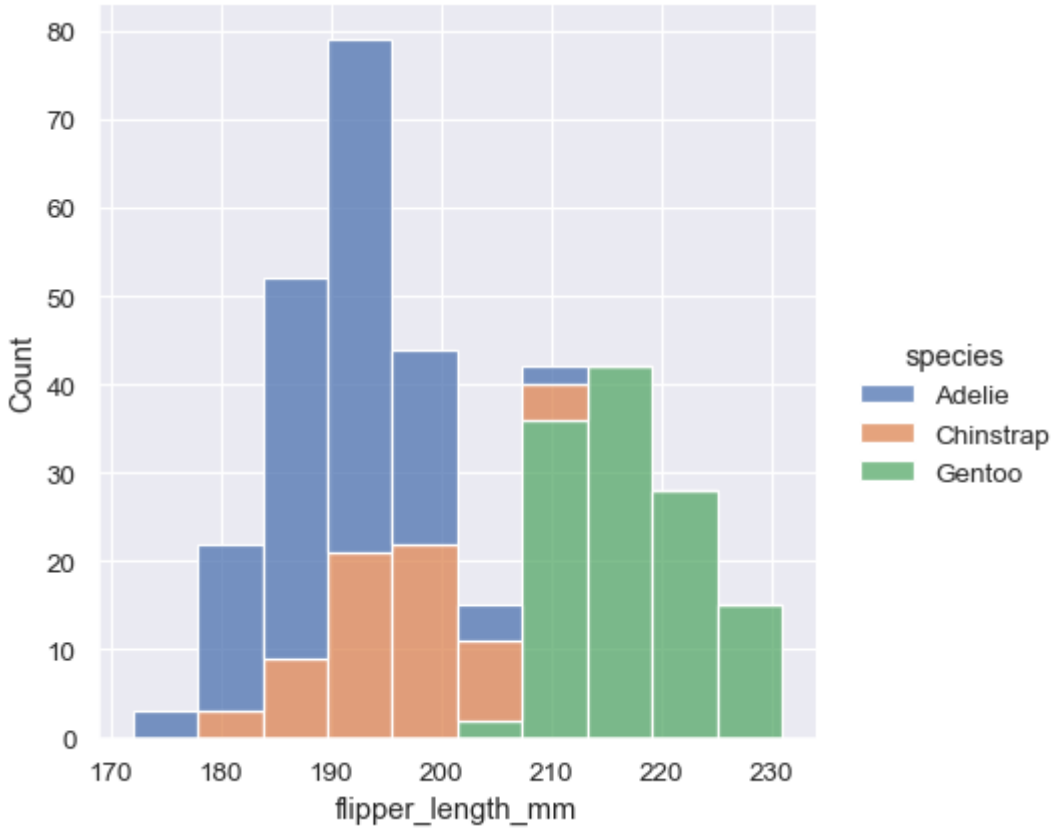
In addition to the different modules, there is a cross-cutting classification of seaborn functions as "axes-level" or "figure-level". The examples above are axes-level functions. They plot data onto a single **matplotlib.pyplot.Axes** object, which is the return value of the function.

In contrast, figure-level functions interface with matplotlib through a seaborn object, usually a **FacetGrid**, that manages the figure. Each module has a single figure-level function, which offers a unitary interface to its various axes-level functions. The organization looks a bit like this:



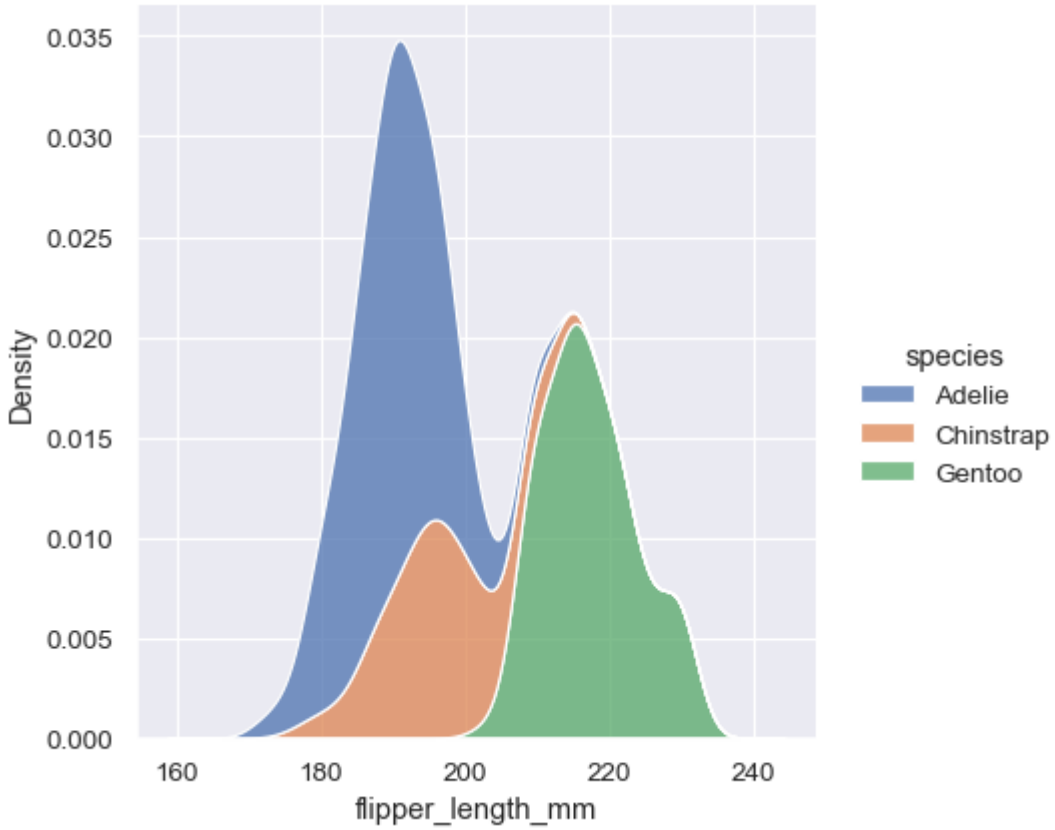
For example, **displot()** is the figure-level function for the distributions module. Its default behavior is to draw a histogram, using the same code as **histplot()** behind the scenes:

```
sns.displot(data=penguins, x="flipper_length_mm", hue="species",
multiple="stack")
```



To draw a kernel density plot instead, using the same code as **kdeplot()**, select it using the kind parameter:

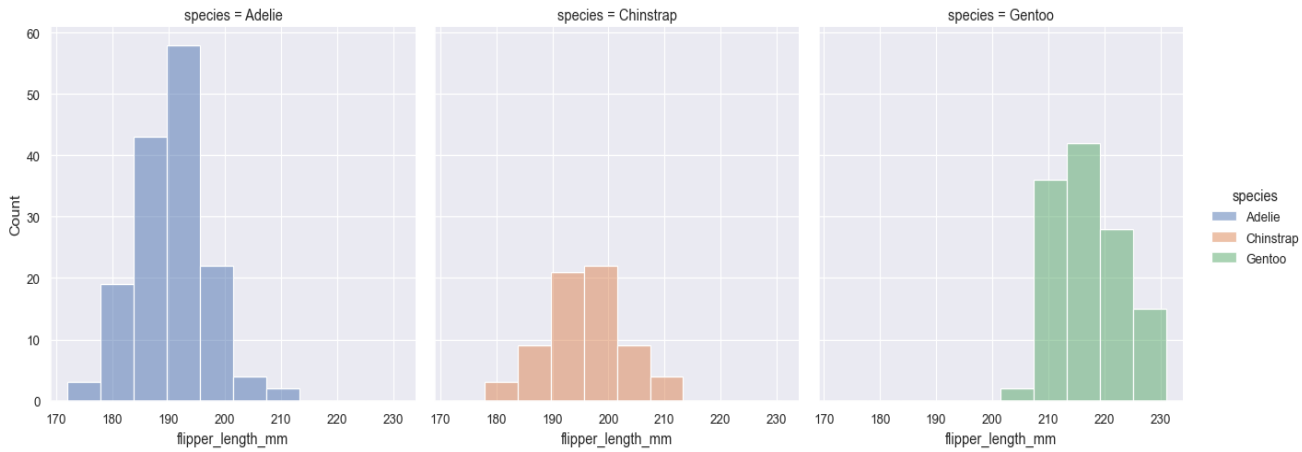
```
sns.displot(data=penguins, x="flipper_length_mm", hue="species",
multiple="stack", kind="kde")
```



You'll notice that the figure-level plots look mostly like their axes-level counterparts, but there are a few differences. Notably, the legend is placed outside the plot. They also have a slightly different shape (more on that shortly).

The most useful feature offered by the figure-level functions is that they can easily create figures with multiple subplots. For example, instead of stacking the three distributions for each species of penguins in the same axes, we can "facet" them by plotting each distribution across the columns of the figure:

```
sns.displot(data=penguins, x="flipper_length_mm", hue="species",
col="species")
```



The figure-level functions wrap their axes-level counterparts and pass the kind-specific keyword arguments (such as the bin size for a histogram) down to the underlying function. That means they are no less flexible, but there is a downside: the kind-specific parameters don't appear in the function signature or docstrings. Some of their features might be less discoverable, and you may need to look at two different pages of the documentation before understanding how to achieve a specific goal.

Axes-level functions make self-contained plots

The axes-level functions are written to act like drop-in replacements for matplotlib functions. While they add axis labels and legends automatically, they don't modify anything beyond the axes that they are drawn into. That means they can be composed into arbitrarily-complex matplotlib figures with predictable results.

The axes-level functions call **matplotlib.pyplot.gca()** internally, which hooks into the matplotlib state-machine interface so that they draw their plots on the "currently-active" axes. But they additionally accept an ax= argument, which integrates with the object-oriented interface and lets you specify exactly where each plot should go:

```
f, axs = plt.subplots(1, 2, figsize=(8, 4),
gridspec_kw=dict(width_ratios=[4, 3]))
sns.scatterplot(data=penguins, x="flipper_length_mm",
y="bill_length_mm", hue="species", ax=axs[0])
sns.histplot(data=penguins, x="species", hue="species", shrink=.8,
alpha=.8, legend=False, ax=axs[1])
f.tight_layout()
```

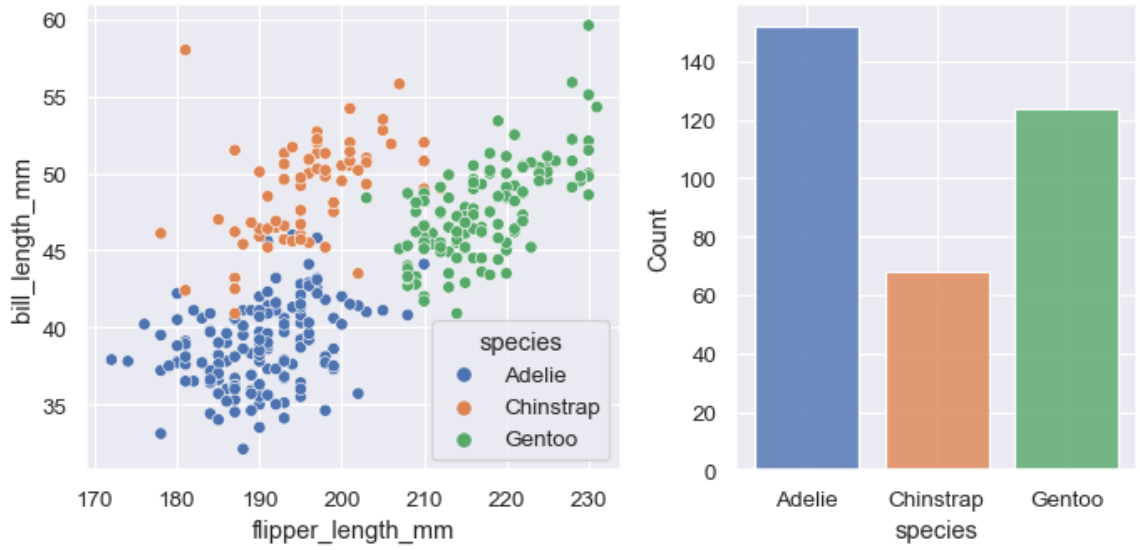
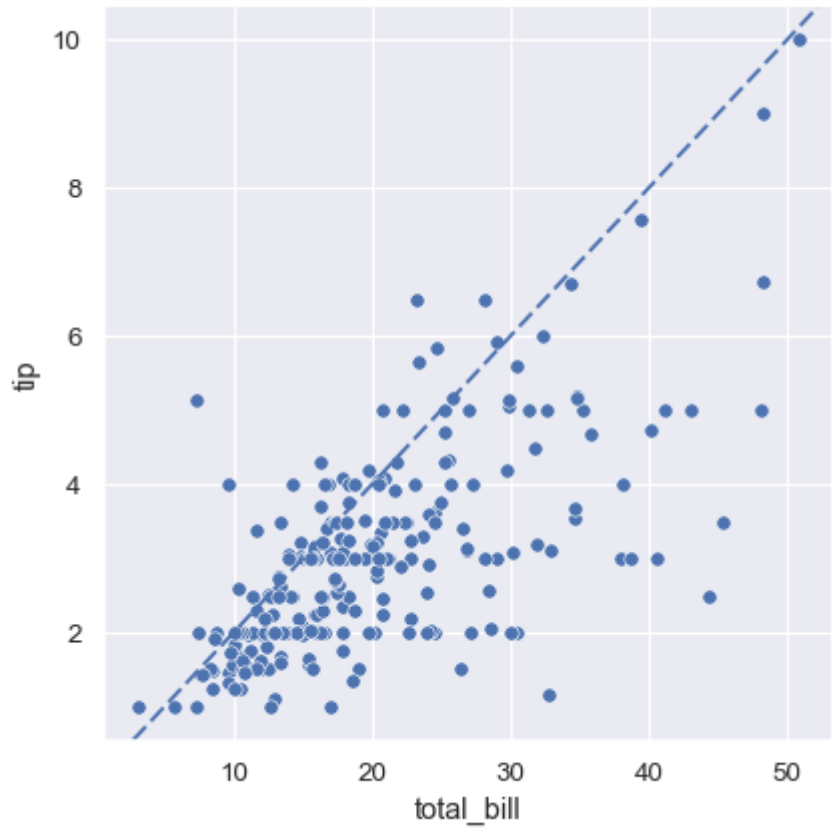


Figure-level functions own their figure

In contrast, figure-level functions cannot (easily) be composed with other plots. By design, they "own" their own figure, including its initialization, so there's no notion of using a figure-level function to draw a plot onto an existing axes. This constraint allows the figure-level functions to implement features such as putting the legend outside of the plot.

Nevertheless, it is possible to go beyond what the figure-level functions offer by accessing the matplotlib axes on the object that they return and adding other elements to the plot that way:

```
tips = sns.load_dataset("tips")
g = sns.relplot(data=tips, x="total_bill", y="tip")
g.ax.axline(xy1=(10, 2), slope=.2, color="b", dashes=(5, 2))
```

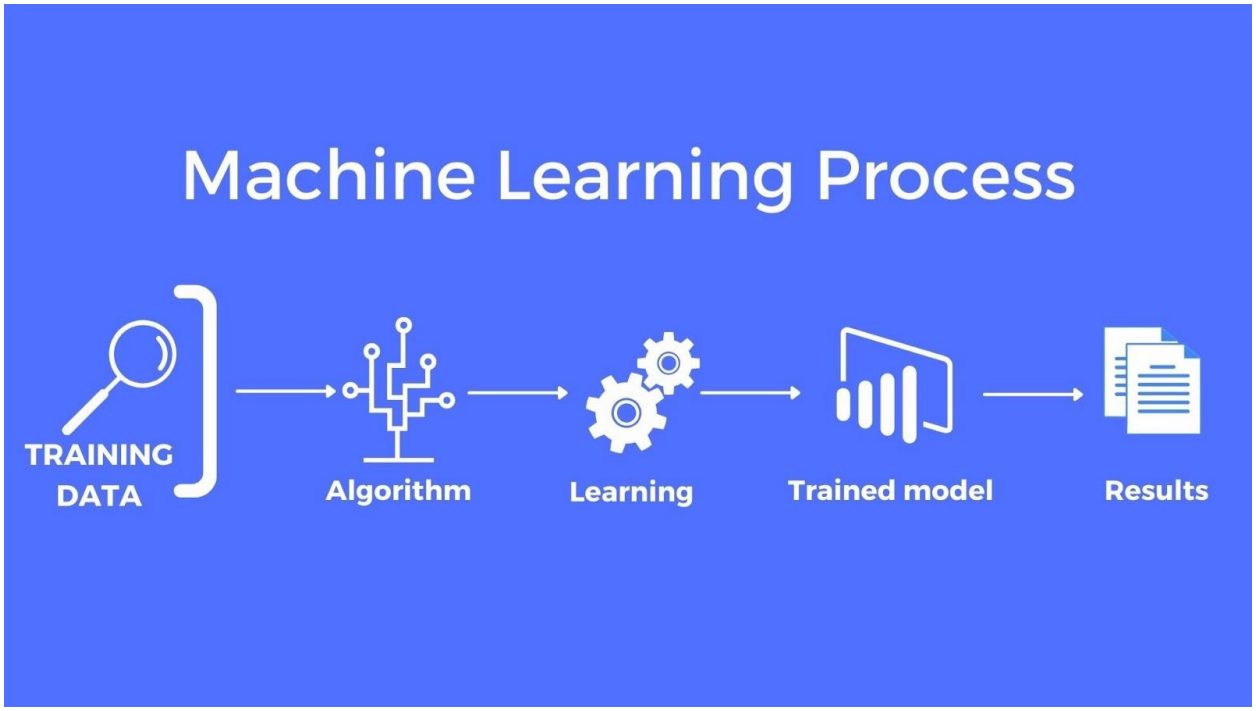


Customizing plots from a figure-level function

The figure-level functions return a **FacetGrid** instance, which has a few methods for customizing attributes of the plot in a way that is "smart" about the subplot organization. For example, you can change the labels on the external axes using a single line of code:

```
g = sns.relplot(data=penguins, x="flipper_length_mm",
y="bill_length_mm", col="sex")
g.set_axis_labels("Flipper length (mm)", "Bill length (mm)")
```

While convenient, this does add a bit of extra complexity, as you need to remember that this method is not part of the matplotlib API and exists only when using a figure-level function.



Machine Learning is making the computer learn from studying data and statistics.

Machine Learning is a step into the direction of artificial intelligence (AI).

Machine Learning is a program that analyses data and learns to predict the outcome.

Mean, Median, and Mode

In Machine Learning (and in mathematics) there are often three values that interests us:

- **Mean** - The average value
- **Median** - The mid point value
- **Mode** - The most common value

What is Standard Deviation?

Standard deviation is a number that describes how spread out the values are.

A low standard deviation means that most of the numbers are close to the mean (average) value.

A high standard deviation means that the values are spread out over a wider range.

Data Distribution

Earlier in this tutorial we have worked with very small amounts of data in our examples, just to understand the different concepts.

In the real world, the data sets are much bigger, but it can be difficult to gather real world data, at least at an early stage of a project.

Normal Data Distribution

In the previous chapter we learned how to create a completely random array, of a given size, and between two given values.

In this chapter we will learn how to create an array where the values are concentrated around a given value.

In probability theory this kind of data distribution is known as the *normal data distribution*, or the *Gaussian data distribution*, after the mathematician Carl Friedrich Gauss who came up with the formula of this data distribution.

Regression

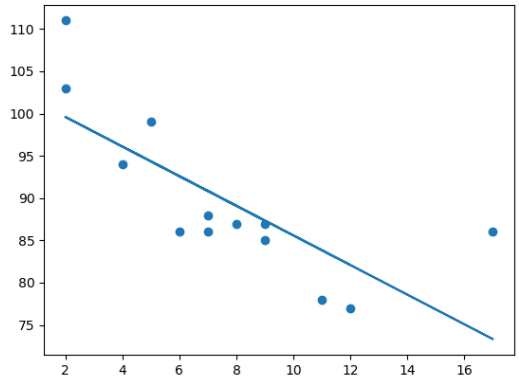
The term regression is used when you try to find the relationship between variables.

In Machine Learning, and in statistical modeling, that relationship is used to predict the outcome of future events.

Linear Regression

Linear regression uses the relationship between the data-points to draw a straight line through all them.

This line can be used to predict future values.



How Does it Work?

Python has methods for finding a relationship between data-points and to draw a line of linear regression. We will show you how to use these methods instead of going through the mathematic formula.

In the example below, the x-axis represents age, and the y-axis represents speed. We have registered the age and speed of 13 cars as they were passing a tollbooth. Let us see if the data we collected could be used in a linear regression:

Multiple Regression

Multiple regression is like linear regression, but with more than one independent value, meaning that we try to predict a value based on **two or more** variables.

Evaluation our model

What is Train/Test ?

Train/Test is a method to measure the accuracy of your model.

It is called Train/Test because you split the data set into two sets: a training set and a testing set.

80% for training, and 20% for testing.

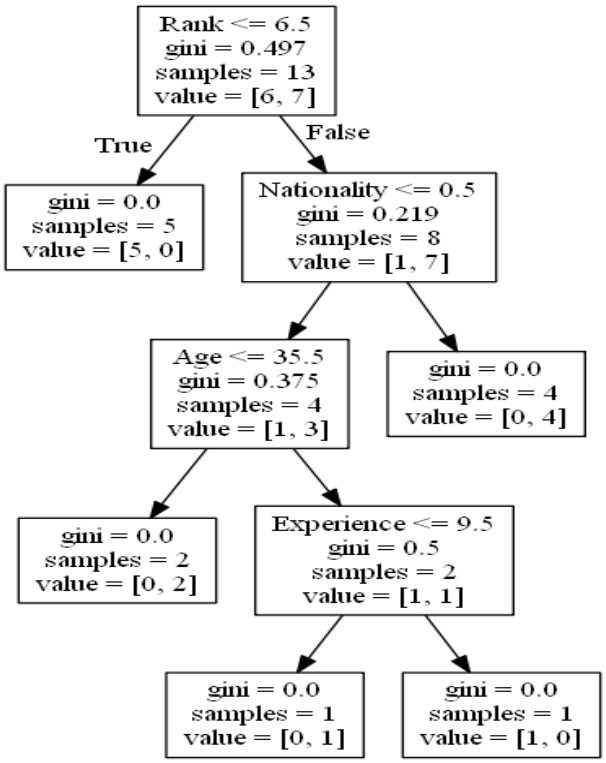
You *train* the model using the training set.

You *test* the model using the testing set.

Train the model means *create* the model.

Test the model means test the accuracy of the model.

Decision Tree



A Decision Tree is a Flow Chart, and can help you make decisions based on previous experience.

In the example, a person will try to decide if he/she should go to a comedy show or not.

Luckily our example person has registered every time there was a comedy show in town, and registered some information about the comedian, and also registered if he/she went or not.

What is a confusion matrix?

It is a table that is used in classification problems to assess where errors in the model were made.

The rows represent the actual classes the outcomes should have been. While the columns represent the predictions we have made. Using this table it is easy to see which predictions are wrong.

Creating a Confusion Matrix

Confusion matrixes can be created by predictions made from a logistic regression.

For now we will generate actual and predicted values by utilizing NumPy:

```
import numpy
```

Naive Bayes

Naive Bayes is classification approach that adopts the principle of class conditional independence from the Bayes Theorem. This means that the presence of one feature does not impact the presence of another in the probability of a given outcome, and each predictor has an equal effect on that result. There are three types of Naïve Bayes classifiers: Multinomial Naïve Bayes, Bernoulli Naïve Bayes, and Gaussian Naïve Bayes. This technique is primarily used in text classification, spam identification, and recommendation systems.

Linear regression

Linear regression is used to identify the relationship between a dependent variable and one or more independent variables and is typically leveraged to make predictions about future outcomes. When there is only one independent variable and one dependent variable, it is known as simple linear regression. As the number of independent variables increases, it is referred to as multiple linear regression. For each type of linear regression, it seeks to plot a line of best fit, which is calculated through the method of least squares. However, unlike other regression models, this line is straight when plotted on a graph.

Logistic regression

While linear regression is leveraged when dependent variables are continuous, logistical regression is selected when the dependent variable is categorical, meaning they have binary outputs, such as "true" and "false" or "yes" and "no." While both regression models seek to understand relationships between data inputs, logistic regression is mainly used to solve binary classification problems, such as spam identification.

Support vector machine (SVM)

A support vector machine is a popular supervised learning model developed by Vladimir Vapnik, used for both data classification and regression. That said, it is typically leveraged for classification problems, constructing a hyperplane where the distance between two classes of data points is at its maximum. This hyperplane is known as the decision boundary, separating the classes of data points (e.g., oranges vs. apples) on either side of the plane.

K-nearest neighbor

K-nearest neighbor, also known as the KNN algorithm, is a non-parametric algorithm that classifies data points based on their proximity and association to other available data. This algorithm assumes that similar data points can be found near each other. As a result, it seeks to calculate the distance between data points, usually through Euclidean distance, and then it assigns a category based on the most frequent category or average.

Its ease of use and low calculation time make it a preferred algorithm by data scientists, but as the test dataset grows, the processing time lengthens, making it less appealing for classification tasks. KNN is typically used for recommendation engines and image recognition.

Random forest

Random forest is another flexible supervised machine learning algorithm used for both classification and regression purposes. The "forest" references a collection of uncorrelated decision trees, which are then merged together to reduce variance and create more accurate data predictions.

supervised learning

Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of labeled datasets to train algorithms that to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately, which occurs as part of the cross validation process. Supervised learning helps organizations solve for a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox.

Working Principal

Supervised learning uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.

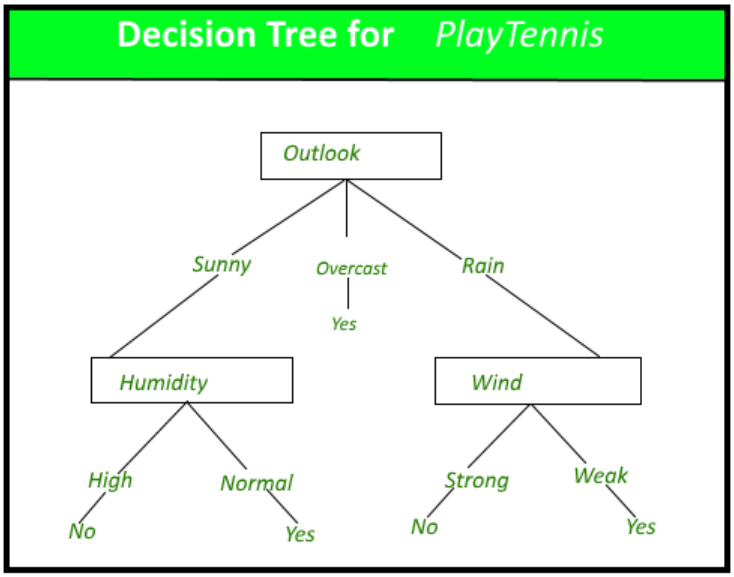
Supervised learning can be separated into two types of problems when data mining—classification and regression:

- Classification uses an algorithm to accurately assign test data into specific categories. It recognizes specific entities within the dataset and attempts to draw some conclusions on how those entities should be labeled or defined. Common classification algorithms are linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbor, and random forest, which are described in more detail below.
- Regression is used to understand the relationship between dependent and independent variables. It is commonly used to make projections, such as for sales revenue for a given business. Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

Our project is based on supervised learning . so, we need to use certain algorithms likes ensemble technique Random Forest which helps to give a feasible solution

Decision Tree

Decision trees are a popular machine learning algorithm that can be used for both regression and classification tasks. They are easy to understand, interpret, and implement, making them an ideal choice for beginners in the field of machine learning.



It is an example of decision tree

A decision tree for the concept PlayTennis.

Construction of Decision Tree: A tree can be “learned” by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. The construction of a decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery. Decision trees can handle high-dimensional data. In general decision tree, classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge on classification.

Advantages:

1. Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
2. A decision tree does not require normalization of data.
3. A decision tree does not require scaling of data as well.
4. Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
5. A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.

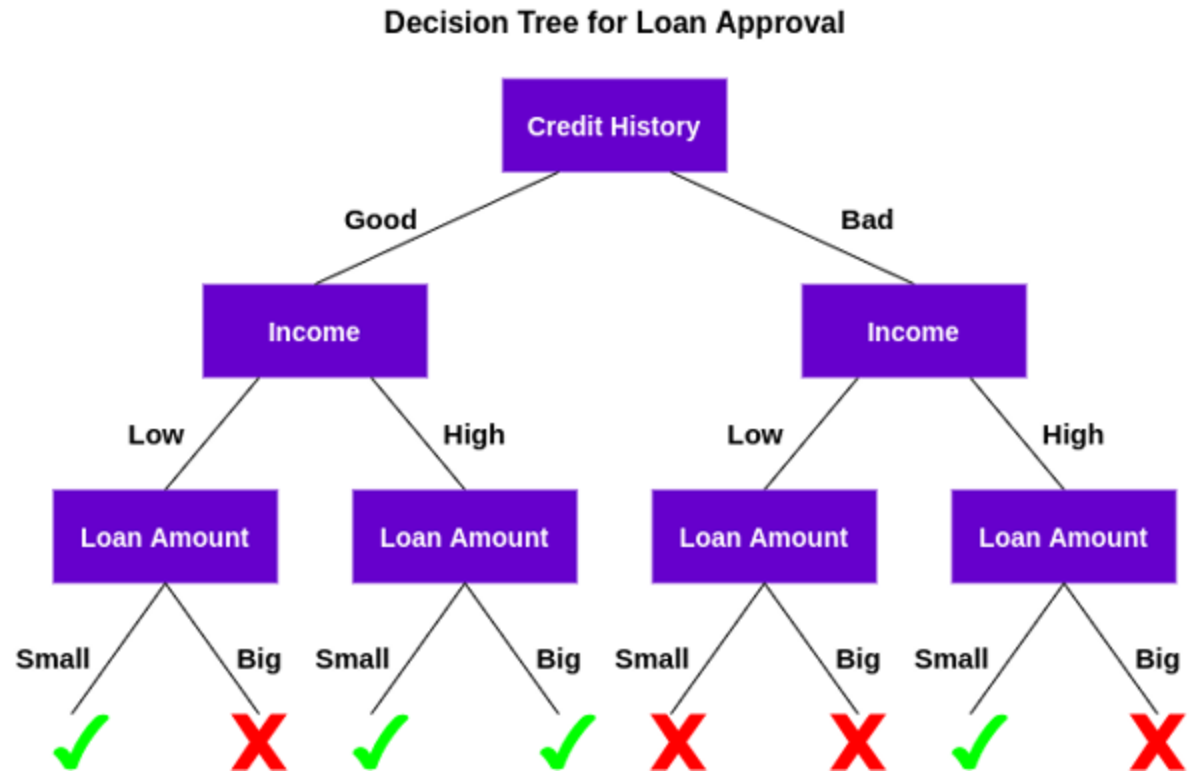
Disadvantage:

1. A small change in the data can cause a large change in the structure of the decision tree causing instability.
2. For a Decision tree sometimes calculation can go far more complex compared to other algorithms.
3. Decision tree often involves higher time to train the model.
4. Decision tree training is relatively expensive as the complexity and time has taken are more.
5. The Decision Tree algorithm is inadequate for applying regression and predicting continuous values.

these two methods, and answer the key question – which machine learning algorithm should you go with?

What Are Decision Trees?

A decision tree is a supervised machine-learning algorithm that can be used for both classification and regression problems. Algorithm builds its model in the structure of a tree along with decision nodes and leaf nodes. A decision tree is simply a series of sequential decisions made to reach a specific result. Here's an illustration of a decision tree in action (using our above example):



Let's understand how this tree works.

First, it checks if the customer has a good credit history. Based on that, it classifies the customer into two groups, i.e., customers with good credit history and customers with bad credit history. Then, it checks the income of the customer and again classifies him/her into two groups. Finally, it checks the loan amount requested by the customer. Based on the outcomes from checking these three features, the decision tree decides if the customer's loan should be approved or not.

The features/attributes and conditions can change based on the data and complexity of the problem, but the overall idea remains the same. So, a decision tree makes a series of

decisions based on a set of features/attributes present in the data, which in this case were credit history, income, and loan amount.

Now, you might be wondering:

Why did the decision tree check the credit score first and not the income?

This is known as feature importance, and the sequence of attributes to be checked is decided on the basis of criteria like the **Gini Impurity Index** or **Information Gain.** The explanation of these concepts is outside the scope of our article here, but you can refer to either of the below resources to learn all about decision trees:

- Tree-Based Algorithms: A Complete Tutorial from Scratch (in R & Python)
- Getting Started with Decision Trees (Free Course)

Note: The idea behind this article is to compare decision trees and random forests. Therefore, I will not go into the details of the basic concepts, but I will provide the relevant links in case you wish to explore them further.

What Is Random Forest?

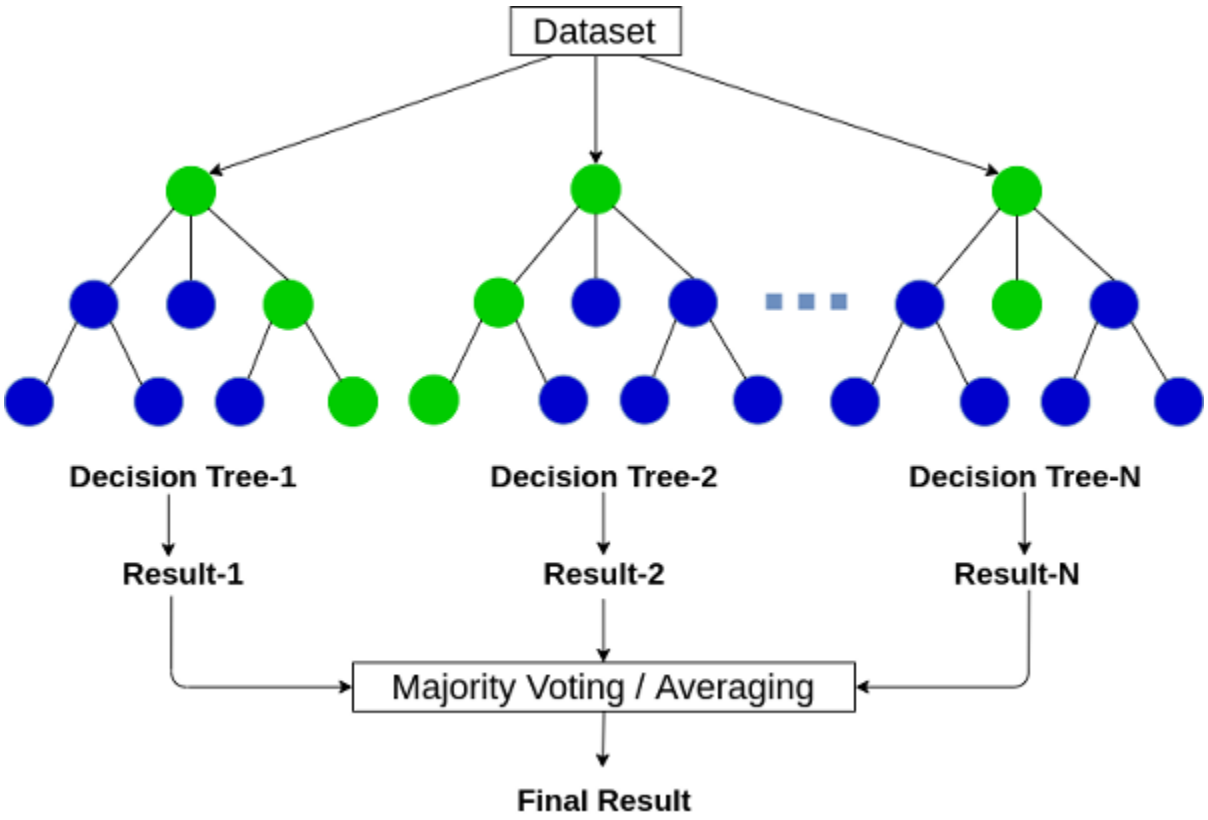
The decision tree algorithm is quite easy to understand and interpret. But data scientist uses random forest mostly, a single tree is not sufficient for producing effective results. This is where the Random Forest algorithm comes into the picture.

Random Forest is a tree-based machine learning algorithm that leverages the power of multiple decision trees for making decisions. As the name suggests, it is a “forest” of trees!

But why do we call it a “random” forest? That’s because it is a forest of **randomly created decision trees**. Each node in the decision tree works on a random subset of features to calculate the output. The random forest then combines the output of individual decision trees to generate the final output. Bootstrapping is the process of randomly selecting items from the training dataset. This is a haphazard technique. It assembles randomized decisions based on several decisions and makes the final decision based on the majority voting.

In simple words:

> The Random Forest Algorithm combines the output of multiple (randomly created) Decision Trees to generate the final output.



This process of combining the output of multiple individual models (also known as weak learners) is called **Ensemble Learning**. If you want to read more about how the random forest and other ensemble learning algorithms work, check out the following articles:

- Building a Random Forest from Scratch & Understanding Real-World Data Products
- A Beginner's Guide to Random Forest Hyperparameter Tuning
- A Comprehensive Guide to Ensemble Learning (with Python codes)
- How to build Ensemble Models in Machine Learning? (with code in R)

Now the question is, how can we decide which algorithm to choose between a decision tree and a random forest? Let's see them both in action before we make any conclusions!

Random Forest vs. Decision Tree in Python

In this section, we will be using Python to solve a binary classification problem using both a decision tree as well as a random forest. We will then compare their results and see which one suited our problem the best.

We'll be working on the Loan Prediction dataset from Analytics Vidhya's DataHack platform. This is a binary classification problem where we have to determine if a person should be given a loan or not based on a certain set of features.

Note: You can go to the **DataHack** platform and compete with other people in various online machine-learning competitions and stand a chance to win exciting prizes.

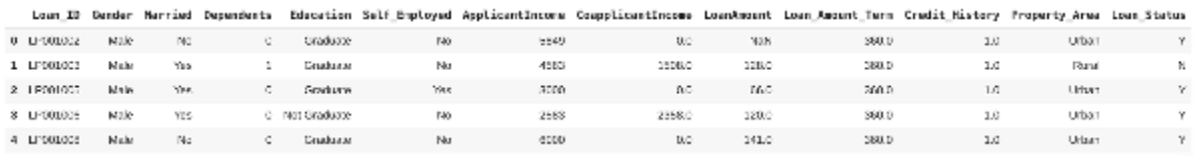
Ready to code?

Step 1: Loading the Libraries and Dataset

Let's start by importing the required Python libraries and our dataset:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import f1_score
from sklearn.model_selection import train_test_split
# Importing dataset
df=pd.read_csv('dataset.csv')
df.head()
```

view rawrfc_vs_dt-1.py hosted with by GitHub



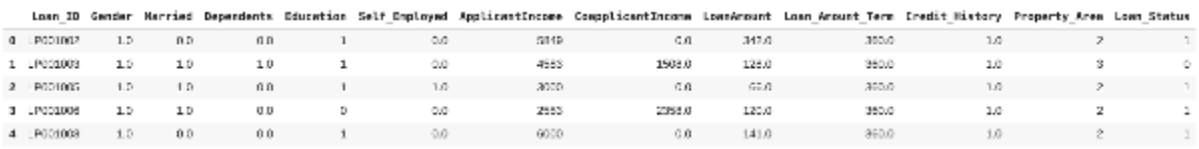
The dataset consists of 614 rows and 13 features, including credit history, marital status, loan amount, and gender. Here, the target variable is *Loan_Status*, which indicates whether a person should be given a loan or not.

Step 2: Data Preprocessing

Now comes the most crucial part of any data science project – **data preprocessing** and **feature engineering**. In this section, I will deal with the categorical variables in the data and imputing the missing values.

I will impute the missing values in the categorical variables with the mode and the continuous variables with the mean (for the respective columns). Also, we will label encoding the categorical values in the data. You can read this article to learn more about **Label Encoding**.

Python Code:



	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	1.0	0.0	0.0	1	0.0	5849	0.0	347.0	360.0	1.0	2	1
1	LP001003	1.0	1.0	1.0	1	0.0	4583	1508.0	128.0	360.0	1.0	0	0
2	LP001005	1.0	1.0	0.0	1	1.0	3000	0.0	66.0	360.0	1.0	2	1
3	LP001006	1.0	1.0	0.0	0	0.0	2583	2358.0	120.0	360.0	1.0	2	1
4	LP001008	1.0	0.0	0.0	1	0.0	6000	0.0	141.0	360.0	1.0	2	1

Step 3: Creating Train and Test Sets

Now, let's split the dataset in an **80:20** ratio for training and test set, respectively:

```
X=df.drop(columns=['Loan_ID','Loan_Status']).values

Y=df['Loan_Status'].values

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 42)
```

There are many different ways in which machine learning models make decisions. Decision Trees and Random Forests are two of the most common decision-making processes used in ML. Hence, there is always confusion, comparison, and debate about Random Forest vs Decision Tree. They both have their advantages, disadvantages, and specific use case, based on which we can choose the right one specific to our requirement and project. This article will give you all the information required to make this choice.

Learning Objectives

- In this tutorial, there is a brief Introduction to Decision Trees and an overview of random forests.
- Clash of Random Forest and Decision Tree (in Code!) and why did random forest outperform decision tree and when to choose which algorithm.

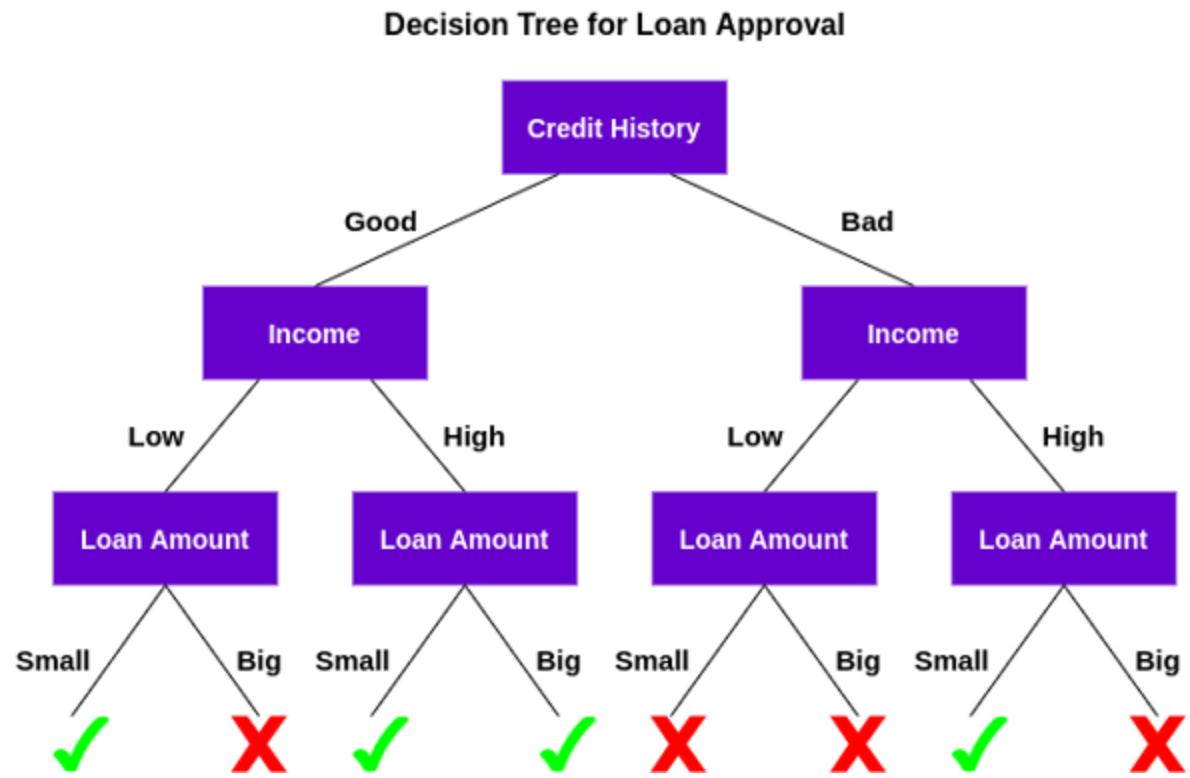
Random Forest vs. Decision Tree Explained by Analogy

Let's start with a thought experiment that will illustrate the difference between a decision tree and a random forest model.

Suppose a bank has to approve a small loan amount for a customer, and the bank needs to make a decision quickly. The bank checks the person's credit history and financial condition and finds that they haven't re-paid the older loan yet. Hence, the bank rejects the application. But here's the catch – the loan amount was very small for the bank's immense coffers, and they could have easily approved it in a very low-risk move. Therefore, the bank lost the chance of making some money.

Now, another loan application comes in a few days down the line, but this time the bank comes up with a different strategy – multiple decision-making processes. Sometimes it checks for credit history first, and sometimes it checks for the customer's financial condition and loan amount first. Then, the bank combines the results from these multiple decision-making processes and decides to give the loan to the customer.

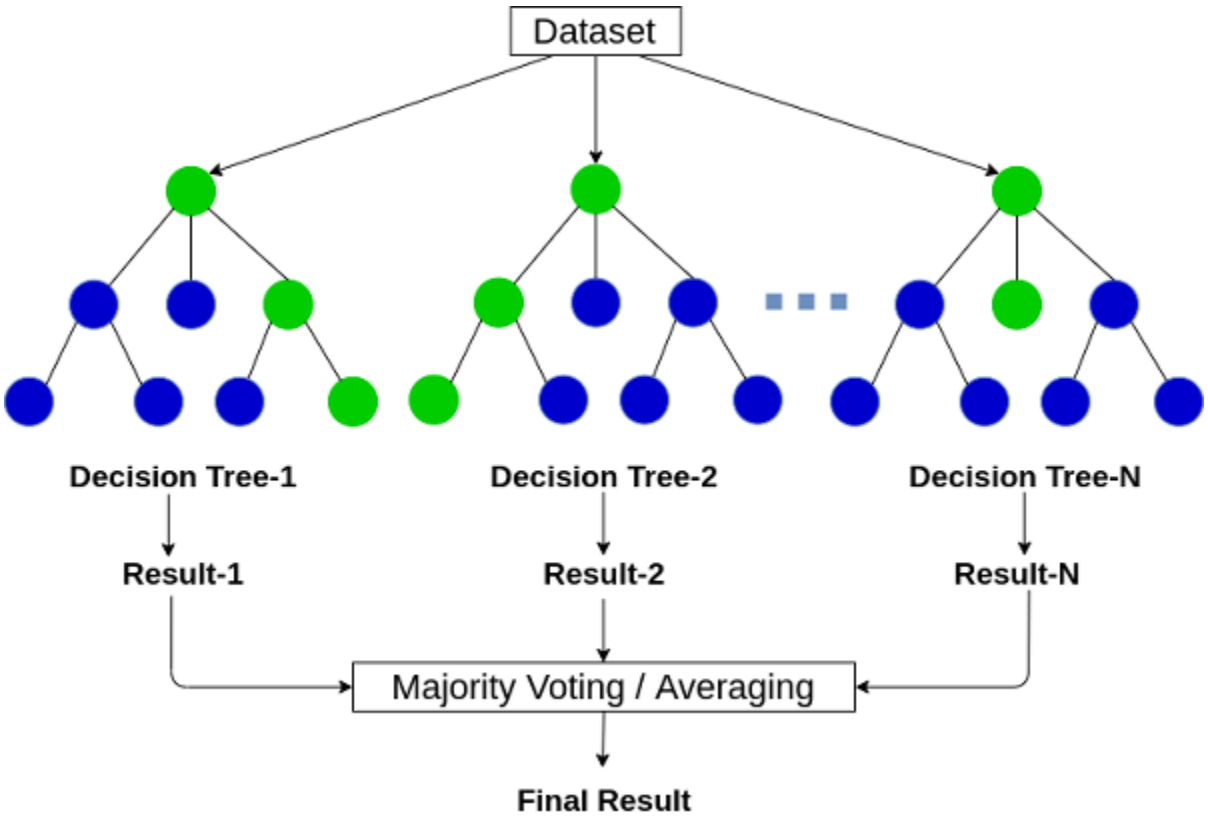
A decision tree is a supervised machine-learning algorithm that can be used for both classification and regression problems. Algorithm builds its model in the structure of a tree along with decision nodes and leaf nodes. A decision tree is simply a series of sequential decisions made to reach a specific result. Here's an illustration of a decision tree in action (using our above example):



Let's understand how this tree works.

First, it checks if the customer has a good credit history. Based on that, it classifies the customer into two groups, i.e., customers with good credit history and customers with bad credit history. Then, it checks the income of the customer and again classifies him/her into two groups. Finally, it checks the loan amount requested by the customer. Based on the outcomes from checking these three features, the decision tree decides if the customer's loan should be approved or not.

The features/attributes and conditions can change based on the data and complexity of the problem, but the overall idea remains the same. So, a decision tree makes a series of decisions based on a set of features/attributes present in the data, which in this case were credit history, income, and loan amount.



This process of combining the output of multiple individual models (also known as weak learners) is called **Ensemble Learning**. If you want to read more about how the random forest and other ensemble learning algorithms work, check out the following articles:

- Building a Random Forest from Scratch & Understanding Real-World Data Products
- A Beginner's Guide to Random Forest Hyperparameter Tuning
- A Comprehensive Guide to Ensemble Learning (with Python codes)
- How to build Ensemble Models in Machine Learning? (with code in R)

Now the question is, how can we decide which algorithm to choose between a decision tree and a random forest? Let's see them both in action before we make any conclusions!

Random Forest vs. Decision Tree in Python

In this section, we will be using Python to solve a binary classification problem using both a decision tree as well as a random forest. We will then compare their results and see which one suited our problem the best.

We'll be working on the Loan Prediction dataset from Analytics Vidhya's DataHack platform. This is a binary classification problem where we have to determine if a person should be given a loan or not based on a certain set of features.

Note: You can go to the **DataHack** platform and compete with other people in various online machine-learning competitions and stand a chance to win exciting prizes.

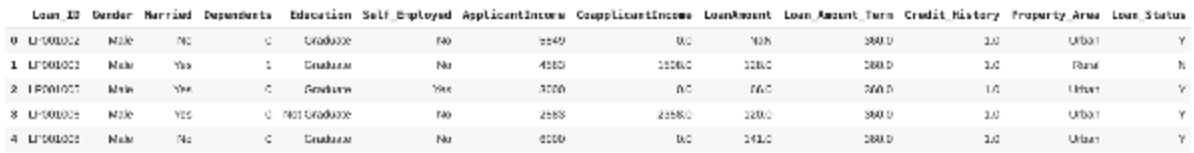
Ready to code?

Step 1: Loading the Libraries and Dataset

Let's start by importing the required Python libraries and our dataset:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import f1_score

from sklearn.model_selection import train_test_split

# Importing dataset

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

df.head()
```

view rawrfc_vs_dt-1.py hosted with by GitHub



	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

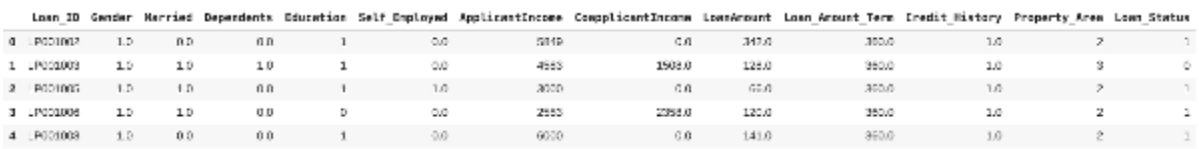
The dataset consists of 614 rows and 13 features, including credit history, marital status, loan amount, and gender. Here, the target variable is *Loan_Status*, which indicates whether a person should be given a loan or not.

Step 2: Data Preprocessing

Now comes the most crucial part of any data science project – **data preprocessing** and **feature engineering**. In this section, I will deal with the categorical variables in the data and imputing the missing values.

I will impute the missing values in the categorical variables with the mode and the continuous variables with the mean (for the respective columns). Also, we will label encoding the categorical values in the data. You can read this article to learn more about **Label Encoding**.

Python Code:

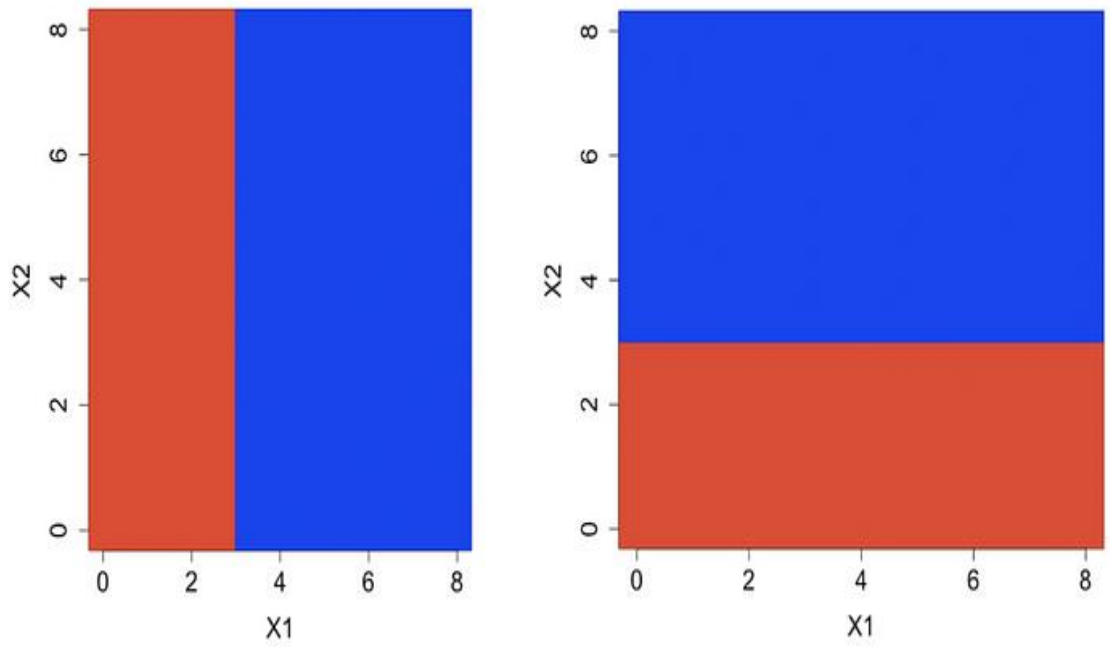


	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	1.0	0.0	0.0	1	0.0	5849	0.0	347.0	360.0	1.0	2	1
1	LP001003	1.0	1.0	1.0	1	0.0	4583	1508.0	128.0	360.0	1.0	0	0
2	LP001005	1.0	1.0	0.0	1	1.0	3000	0.0	66.0	360.0	1.0	2	1
3	LP001006	1.0	1.0	0.0	0	0.0	2583	2358.0	120.0	360.0	1.0	2	1
4	LP001008	1.0	0.0	0.0	1	0.0	6000	0.0	141.0	360.0	1.0	2	1

Step 3: Creating Train and Test Sets

Now, let's split the dataset in an **80:20** ratio for training and test set, respectively:

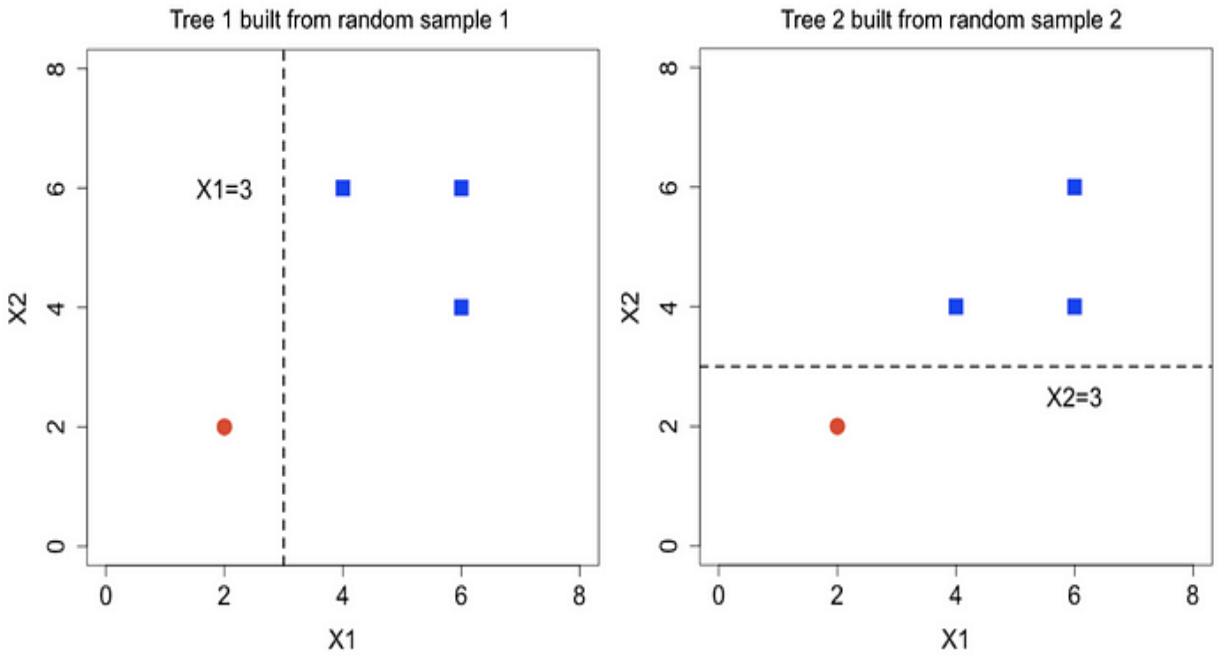
```
X=df.drop(columns=['Loan_ID','Loan_Status']).values

Y=df['Loan_Status'].values

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 42)
```



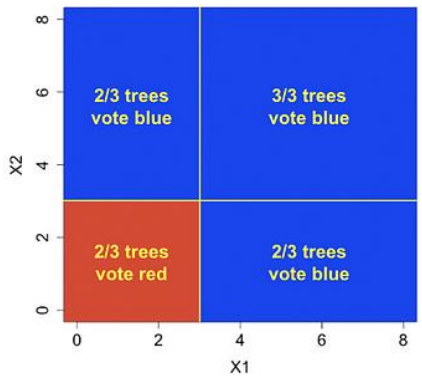
Now consider random forests. For each random sample used for training a tree, the probability that the red point missing from the sample is

$$\left(1-\frac{1}{5}\right)^5 \approx 33\%$$

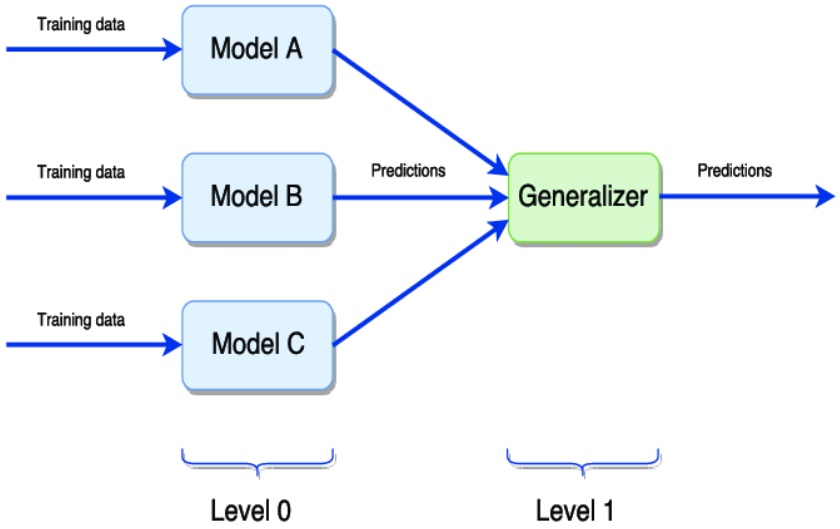
So roughly 1 out of 3 trees is built with all blue data and always predict class blue. The other 2/3 of the trees have the red point in the training data. Since at each node a random subset of features is considered, we expect roughly 1/3 of the trees use x1, and the rest 1/3 uses x2. The splits from the two types of trees are illustrated below.



By aggregating the three types of trees, the decision boundary shown below is now symmetric for x1 and x2. As long as there are enough trees, the boundary should be stable and does not depend on irrelevant information such as the ordering of variables.



The randomness and voting mechanisms in random forests elegantly solve the overfitting problem.



Above picture shows how the random forest model working

AIM:

Problems statements

Usually in our daily life we are using some food delivery apps like zomato , swiggy to order our delicious food at a mean time so may be some of the food are not so much as good as previous one so most of the time we have to give some feed back to those organization which helps to track there quality of food now we are survey for same thing but at that moment one question arose in our mind that which application (online food order) is used by most of the time in near future.

Solution

Now we have to solve this problem using machine learning algorithm so make our machine efficient to predict the result in optimized nature for that we need to collect the data from survey from various location we collected those data by the help of google form so we collect our response in csv file.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Age	Gender	Marital sta	Occupatio	Education	Member i	Place	Which you	Feedback of last order			
2	26	Male	Unmarrie	Service	Graduatio	6	Alipurdua	Zomato	Positive			
3	21	Female	Unmarrie	Student	Graduatio	3	Alipurdua	Zomato	Positive			
4	20	Male	Unmarrie	Student	Graduatio	7	Coochbeh	Zomato	Positive			
5	20	Female	Unmarrie	Student	Graduatio	4	Alipurdua	Zomato	Positive			
6	21	Male	Unmarrie	Student	Graduatio	4	Coochbeh	Zomato	Positive			
7	20	Female	Unmarrie	Student	Graduatio	4	Alipurdua	Zomato	Positive			
8	22	Male	Unmarrie	Student	Graduatio	3	Alipurdua	Zomato	Positive			
9	26	Female	Unmarrie	Service	Graduatio	4	Alipurdua	Zomato	Positive			
10	28	Male	Unmarrie	Service	Graduatio	5	Dinhata	Zomato	Positive			
11	20	Female	Unmarrie	Student	Graduatio	4	Alipurdua	Zomato	Positive			
12	20	Male	Unmarrie	Student	Higher Se	5	Mathabha	Swiggy	Negative			
13	20	Female	Unmarrie	Student	Higher Se	4	Coochbeh	Swiggy	Negative			
14	25	Male	Unmarrie	Service	Graduatio	3	Coochbeh	Swiggy	Positive			
15	30	Female	Married	Business	Masters	4	Jalpaiguri	Swiggy	Positive			
16	21	Male	Unmarrie	Student	Graduatio	3	Alipurdua	Zomato	Positive			
17	21	Female	Unmarrie	Student	Graduatio	3	Alipurdua	Zomato	Positive			
18	21	Male	Unmarrie	Student	Graduatio	4	Coochbeh	Swiggy	Positive			
19	21	Female	Unmarrie	Student	Higher Se	3	Coochbeh	Zomato	Negative			
20	20	Male	Unmarrie	Student	Higher Se	4	Coochbeh	Zomato	Positive			
21	20	Female	Unmarrie	Student	Higher Se	4	Coochbeh	Zomato	Positive			
22	27	Male	Married	Service	Secondary	3	Coochbeh	Zomato	Negative			
23	21	Female	Unmarrie	Student	GNM Nurs	3	Coochbeh	Zomato	Positive			
24	28	Male	Unmarrie	Business	PHD	1	Coochbeh	Zomato	Positive			
25	20	Female	Unmarrie	Student	Graduatio	10	Coochbeh	Zomato	Positive			

Final_response

Response has been taken by the help of google form, we were circulating the form in the various towns and cities. The form contains Age, Gender, Marital Status, Occupation, Education, Member in family, Place, Which app you prefer, Feedback of last order. We collect all the response in CSV format.

Zomato/Swiggy which app you like to prefer

Survey form for education purpose ; DATM, Alipurduar

Age : *

Short-answer text

Gender *

- Male
- Female

Material status *

- Married
- Unmarried

Occupation *

- Service
- Business
- Student

Education Qualification *

- Primary
- Secondary
- Higher Secondary
- Graduation

- Primary
- Secondary
- Higher Secondary
- Graduation
- Masters
- PHD
- Other...

Member in family *

Short-answer text

Place *

- Alipurduar
- Coochbehar
- Other...

Which you like to prefer for online food app *

- Zomato
- Swiggy

Feedback of last order *

- Positive
- Negative

Tools and Hardware

Hardware:

Operating system Windows 10

RAM Minimum 4GB

Software:

Install Python in the system(Updated version Minimum Require 3.10)

Supporting Libraries:

At first install jupyter notebook in your project directory For execute jupyter note book Command:write the command into power shell or terminal inside your project directory

- Jupyter notebook then press enter
- To execute a specific cell in jupyter press shift+enter key

Command to install all the Libraries in the Jupyter

```
!pip3 install plotly
!pip3 install  numpy
!pip3 install pandas
!pip3 install matplotlib
!pip3 install seaborn
```

Jupyter Notebook

Jupyter Notebook is an **open-source, web-based interactive environment,** which allows you to create and share documents that contain **live code, mathematical equations, graphics, maps, plots, visualizations,** and **narrative text**. It integrates with many programming languages like **Python, PHP, R, C#,** etc.

Advantages of Jupyter Notebook

There are the following advantages of Jupyter Notebook –

1. **All in one place:** As you know, Jupyter Notebook is an open-source web-based interactive environment that combines code, text, images, videos, mathematical equations, plots, maps, graphical user interface and widgets to a single document.
2. **Easy to convert:** Jupyter Notebook allows users to convert the notebooks into other formats such as HTML and PDF. It also uses online tools and nbviewer which allows you to render a publicly available notebook in the browser directly.
3. **Easy to share:** Jupyter Notebooks are saved in the structured text files (JSON format), which makes them easily shareable.
4. **Language independent:** Jupyter Notebook is platform-independent because it is represented as JSON (JavaScript Object Notation) format, which is a language-independent, text-based file format. Another reason is that the notebook can be processed by any programing language, and can be converted to any file formats such as Markdown, HTML, PDF, and others.
5. **Interactive code:** Jupyter notebook uses **ipywidgets** packages, which provide many common user interfaces for exploring code and data interactivity.

Disadvantages of Jupyter Notebook

There are the following disadvantages of Jupyter Notebook:

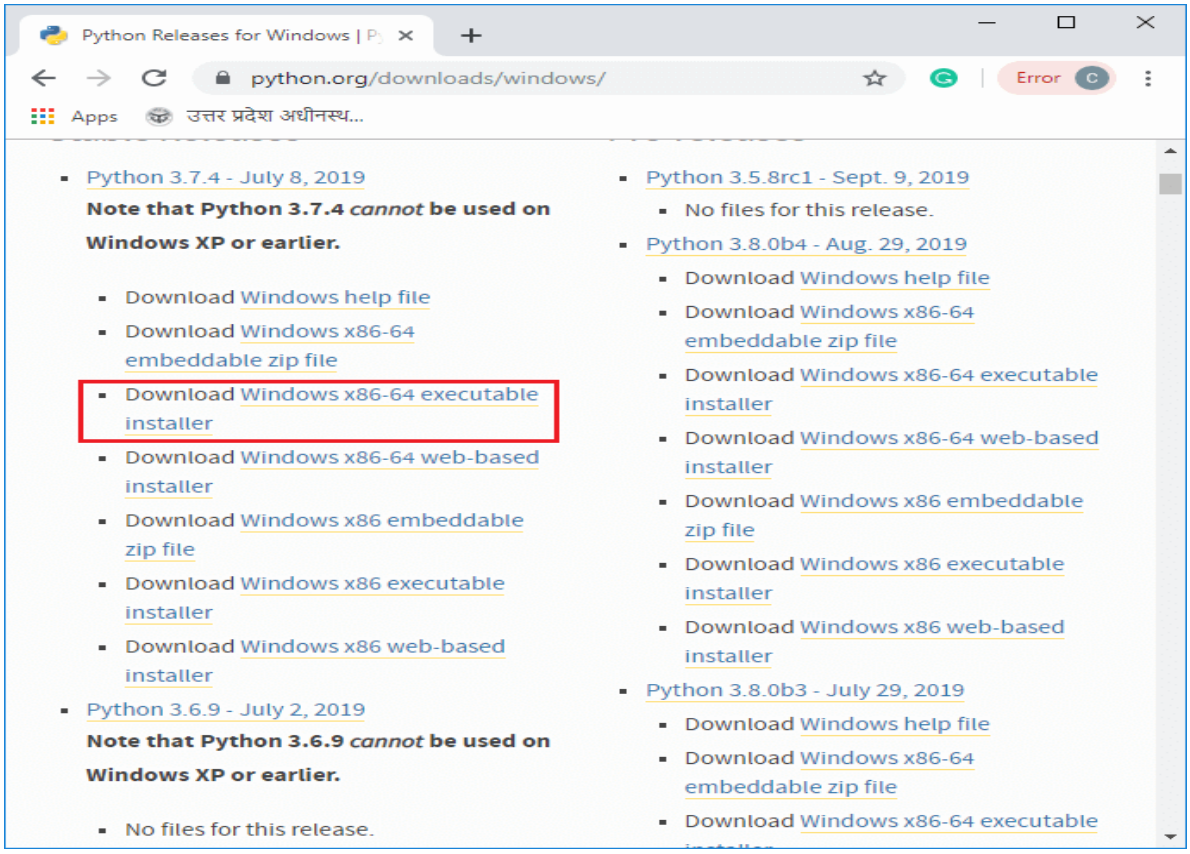
- It is very hard to test long asynchronous tasks.
- Less Security
- It runs cell out of order

- In Jupyter notebook, there is no IDE integration, no linting, and no code-style correction.

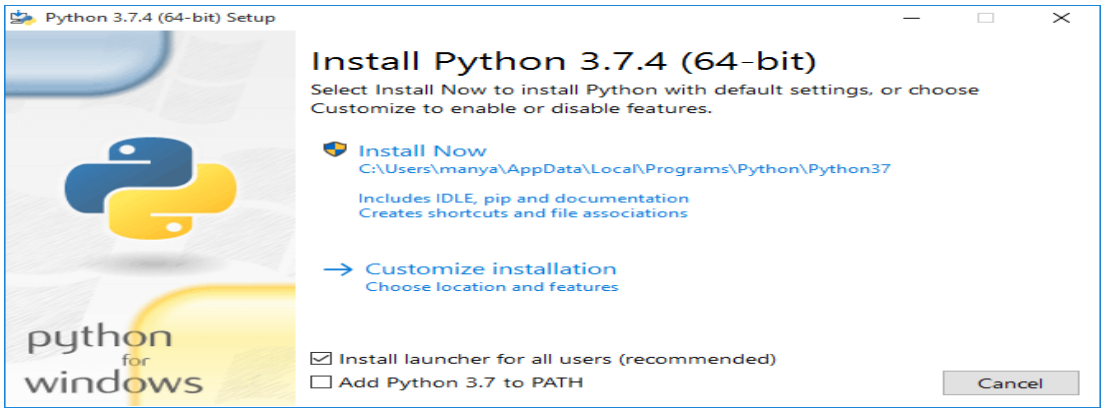
Installation of Jupyter Notebook using pip package

To install the Jupyter Notebook, first, you need to install the Python. You can follow the below steps to download the Python.

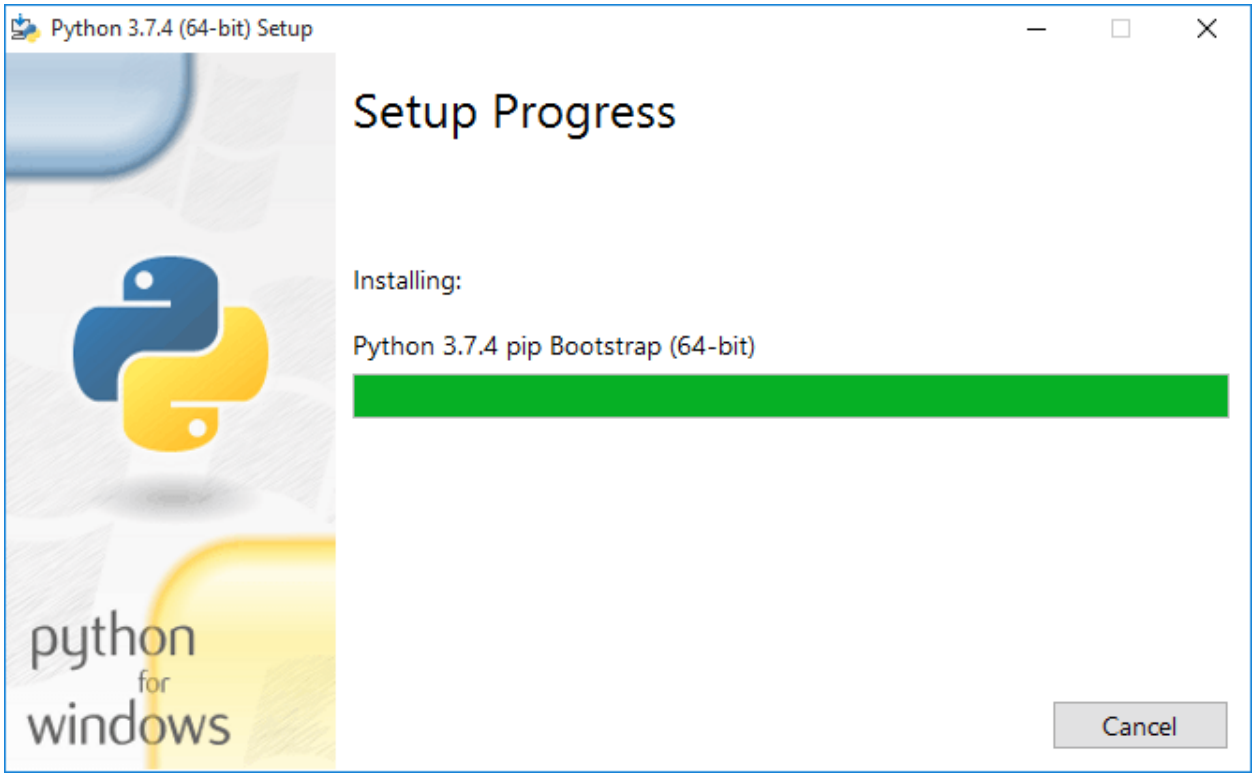
Step1: Click on the link https://www.python.org/downloads/windows/ to download the latest version of the Python.



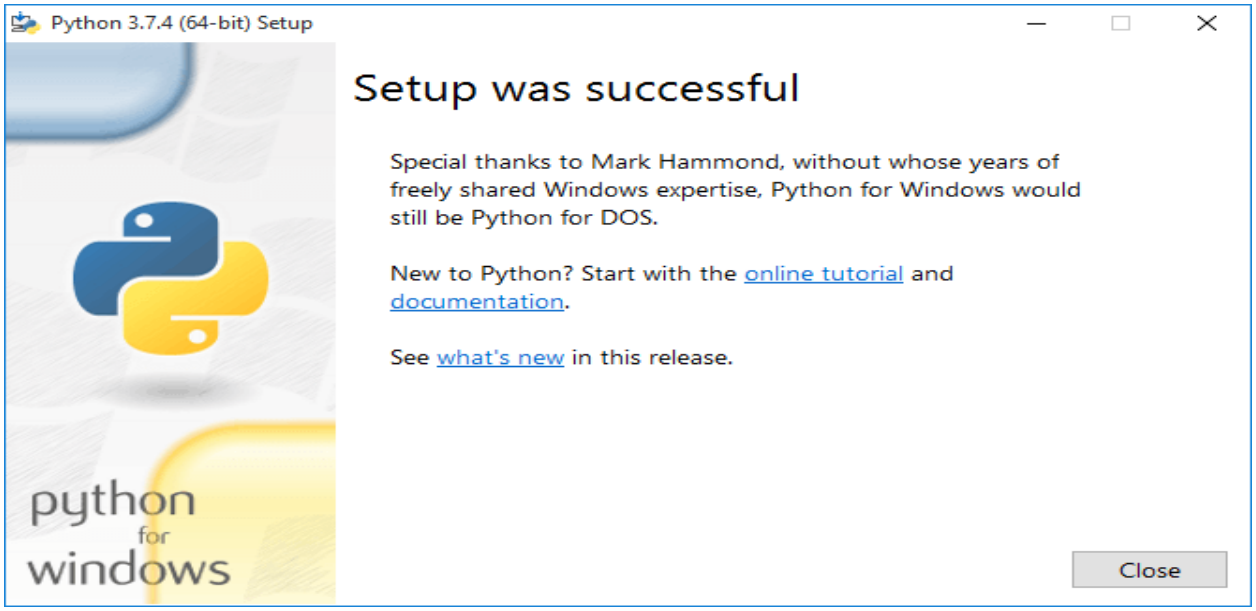
Step2: Now, double click on the downloaded file, the following window is opened. Select **Install Now** to Install Python.



Step3: You can see that installation is in process.



Step4: Once the installation is completed, the following window is open, you simply click on the close.



Once the Python installation is completed, follow the below steps to install the Jupyter Notebook with pip package.

Step1: Open the command prompt.

Step2: Copy/ set the path, where the Python script is presented.

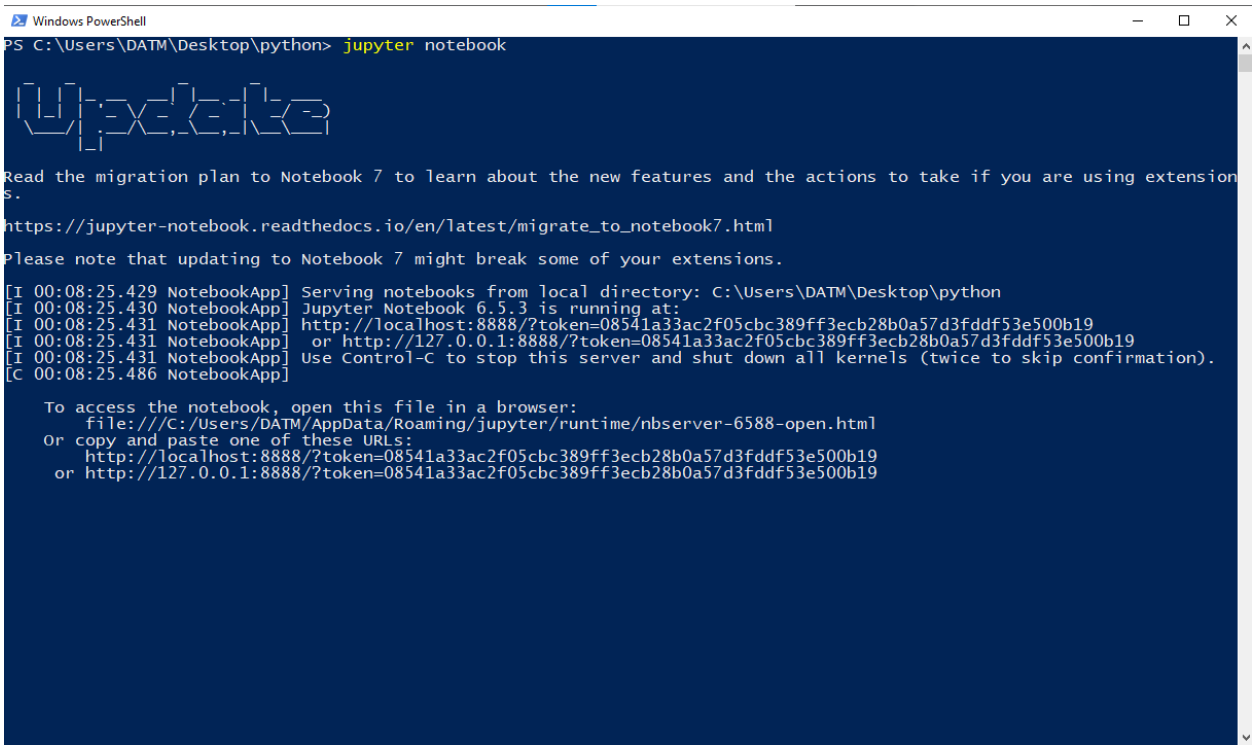
Step3: To upgrade the older version of pip, give the following command after the specified path:

pip install --upgrade pip

Step4: To install the Jupyter Notebook, type the below command:

pip install jupyter

Step5: Once the installation process is completed, you can run your notebook on the server using the following command in command prompt.



After Successfully run the code we get this jupyter interface in the browser

```
import numpy as np

import pandas as pd

import numpy as np

import plotly.express as px

import plotly.graph_objects as go

import matplotlib.pyplot as plt

import seaborn as sns

sns.set_theme(style="whitegrid")

data=pd.read_csv("Final_response.csv")

print(data)
```

```
    Age   Gender Material status Occupation Education Qualification  \
0     26    Male       Unmarried    Service              Graduation
1     21  Female       Unmarried    Student              Graduation
2     20    Male       Unmarried    Student              Graduation
3     20  Female       Unmarried    Student              Graduation
4     21    Male       Unmarried    Student              Graduation
..   ...     ...             ...        ...                     ...
95    26    Male       Unmarried    Student              Graduation
96    32    Male       Unmarried    Student                 Masters
97    20    Male       Unmarried    Student        Higher Secondary
98    20    Male       Unmarried    Student              Graduation
99    25    Male         Married   Business                 Masters

    Member in family       Place Which you like to prefer for online food
app  \
0                  6  Alipurduar                                      Zom
ato
1                  3  Alipurduar                                      Zom
ato
2                  7  Coochbehar                                      Zom
ato
3                  4  Alipurduar                                      Zom
ato
4                  4  Coochbehar                                      Zom
ato
..               ...         ...
...
```

```
95                  3  Alipurduar                                    Swi
ggy
96                  4  Coochbehar                                    Zom
ato
97                  3  Coochbehar                                    Zom
ato
98                  1  Coochbehar                                    Zom
ato
99                  4  Coochbehar                                    Zom
ato

   Feedback of last order
0                Positive
1                Positive
2                Positive
3                Positive
4                Positive
..                    ...
95               Positive
96               Positive
97               Positive
98               Positive
99               Negative

[100 rows x 9 columns]
```

###displying first few record of the data fream as data using the function head(n)

data.head(5

	Age	Gender	Material status	Occupation	Education Qualification	Member in family	Place	Which you like to prefer for online food app	Feedback of last order
0	26	Male	Unmarried	Service	Graduation	6	Alipurduar	Zomato	Positive
1	21	Female	Unmarried	Student	Graduation	3	Alipurduar	Zomato	Positive
2	20	Male	Unmarried	Student	Graduation	7	Coochbehar	Zomato	Positive
3	20	Female	Unmarried	Student	Graduation	4	Alipurduar	Zomato	Positive
4	21	Male	Unmarried	Student	Graduation	4	Coochbehar	Zomato	Positive

finding summury of the data fream as data to disply the all the column name

list(data.columns)

```
['Age ',
 'Gender',
 'Material status',
 'Occupation',
 'Education Qualification',
 'Member in family',
 'Place',
 'Which you like to prefer for online food app',
 'Feedback of last order']
```

the dimention or the size of the data frame can be retrived through the shape which returns tuple from data frame

data.shape

```
(100, 9)
This data set contains 100 rows and 9 column
```

Information of all the columns data types in the data frame

data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 9 columns):
 #   Column                                      Non-Null Count  Dtype
---  ------                                      --------------  -----
 0   Age                                         100 non-null    int64
 1   Gender                                      100 non-null    object
 2   Marital status                              100 non-null    object
 3   Occupation                                  100 non-null    object
 4   Education Qualification                     100 non-null    object
 5   Member in family                            100 non-null    int64
 6   Place                                       100 non-null    object
 7   Which you like to prefer for online food app  100 non-null  object
 8   Feedback of last order                      100 non-null    object
dtypes: int64(2), object(7)
memory usage: 7.2+ KB
```

Slicing and indexing of dataframe

```
import pandas as pd
data=pd.read_csv("Final_response.csv")
data[0:10]
```

	Age	Gender	Marital status	Occupation	Education Qualification	Member in family	Place	Which you like to prefer for online food app	Feedback of last order
0	26	Male	Unmarried	Service	Graduation	6	Alipurduar	Zomato	Positive
1	21	Female	Unmarried	Student	Graduation	3	Alipurduar	Zomato	Positive
2	20	Male	Unmarried	Student	Graduation	7	Coochbehar	Zomato	Positive
3	20	Female	Unmarried	Student	Graduation	4	Alipurduar	Zomato	Positive
4	21	Male	Unmarried	Student	Graduation	4	Coochbehar	Zomato	Positive
5	20	Female	Unmarried	Student	Graduation	4	Alipurduar	Zomato	Positive
6	22	Male	Unmarried	Student	Graduation	3	Alipurduar	Zomato	Positive
7	26	Female	Unmarried	Service	Graduation	4	Alipurduar	Zomato	Positive
8	28	Male	Unmarried	Service	Graduation	5	Dinhata	Zomato	Positive
9	20	Female	Unmarried	Student	Graduation	4	Alipurduar	Zomato	Positive

```
data[-5:]
```

	Age	Gender	Marital status	Occupation	Education Qualification	Member in family	Place	Which you like to prefer for online food app	Feedback of last order
95	26	Male	Unmarried	Student	Graduation	3	Alipurduar	Swiggy	Positive
96	32	Male	Unmarried	Student	Masters	4	Coochbehar	Zomato	Positive
97	20	Male	Unmarried	Student	Higher Secondary	3	Coochbehar	Zomato	Positive
98	20	Male	Unmarried	Student	Graduation	1	Coochbehar	Zomato	Positive
99	25	Male	Married	Business	Masters	4	Coochbehar	Zomato	Negative

```
data['Gender'][0:5]
```

```
0      Male
1    Female
2      Male
3    Female
4      Male
Name: Gender, dtype: object
```

```
data[['Gender', 'Place']][0:5]
```

	Gender	Place
0	Male	Alipurduar
1	Female	Alipurduar
2	Male	Coochbehar
3	Female	Alipurduar
4	Male	Coochbehar

data.iloc[4:9,1:4]

	Gender	Marital status	Occupation
4	Male	Unmarried	Student
5	Female	Unmarried	Student
6	Male	Unmarried	Student
7	Female	Unmarried	Service
8	Male	Unmarried	Service

data.Occupation.value_counts()

```
Student     79
Service     12
Business     9
Name: Occupation, dtype: int64
```

data.Occupation.value_counts(normalize=True)*100

```
Student     79.0
Service     12.0
Business     9.0
Name: Occupation, dtype: float64
```

pd.crosstab(data['Occupation'],data['Feedback of last order'])

Feedback of last order	Negative	Positive
Occupation		
Business	4	5
Service	1	11
Student	9	70

NULL CHECKING in the data frame

data.isna().sum()

```
Age                                              0
Gender                                           0
Marital status                                   0
Occupation                                       0
Education Qualification                          0
Member in family                                 0
Place                                            0
Which you like to prefer for online food app     0
Feedback of last order                           0
dtype: int64
```

grouping and aggregating

import pandas as pd

data=pd.read_csv("Final_response.csv")

data.groupby('Which you like to prefer for online food app')['Member in family'].mean()

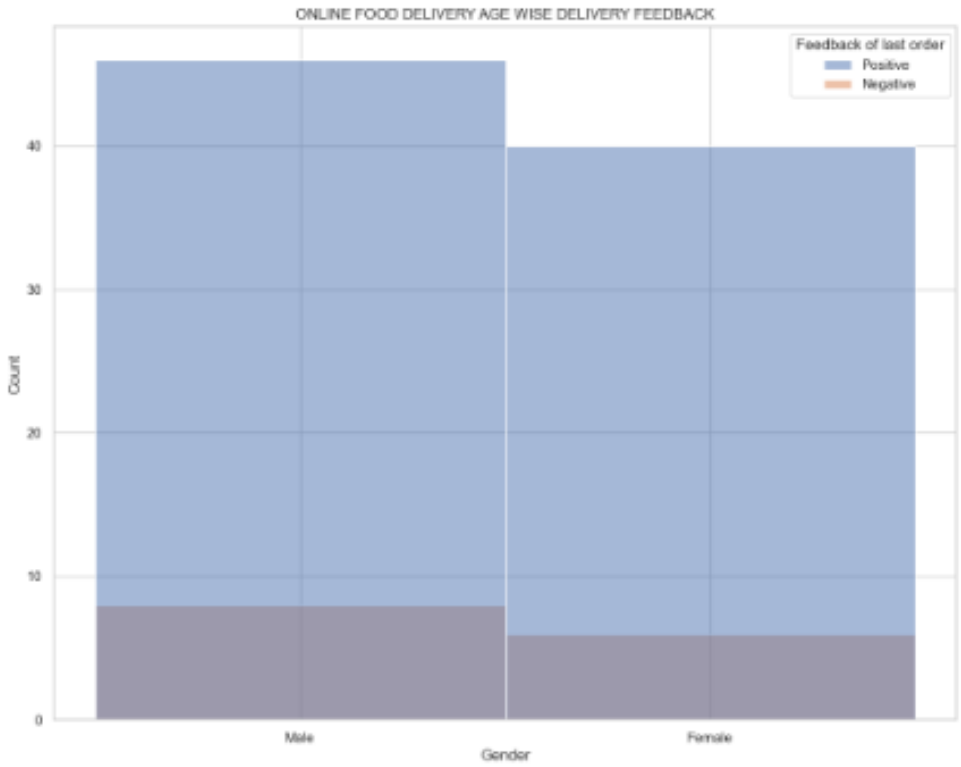
```
Which you like to prefer for online food app
Swiggy    3.88
Zomato    4.20
Name: Member in family, dtype: float64
```

we determining the feedback from the male or female of the whole data set

dataframe =pd.read_csv("Final_response.csv")

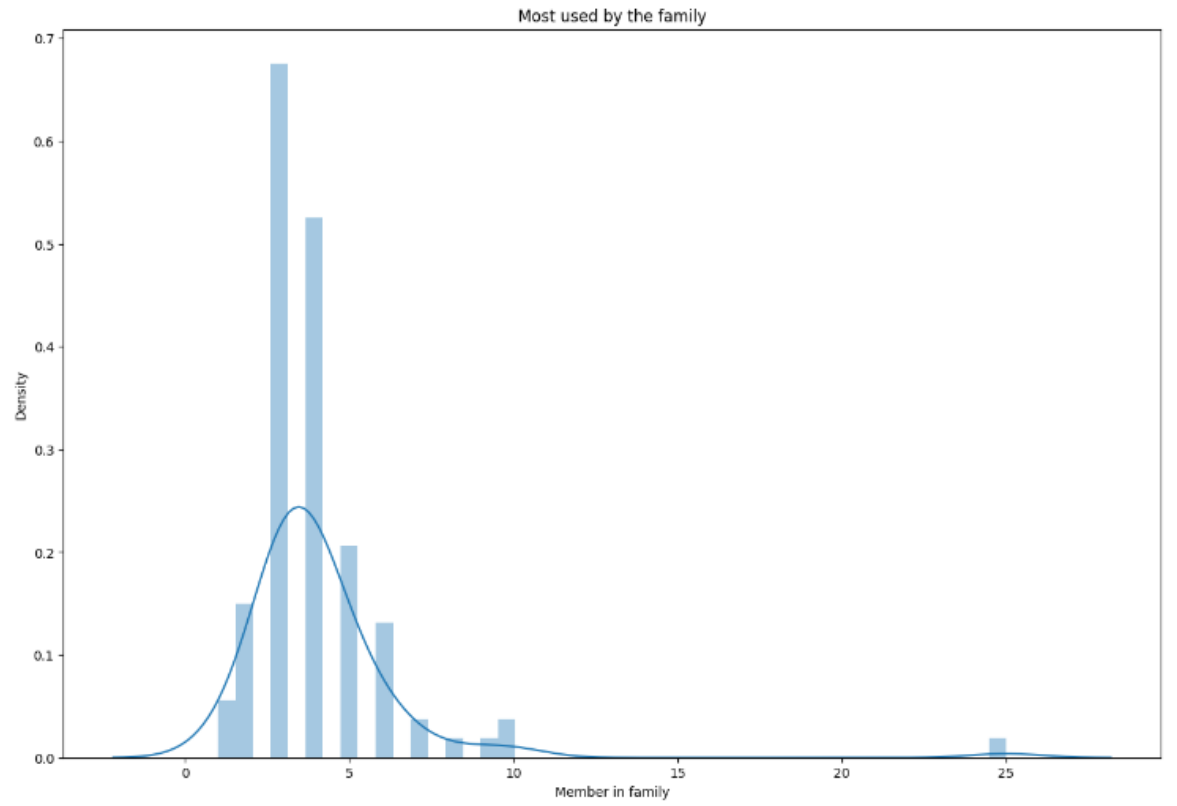
plt.figure(figsize=(15,10))

```
plt.title("ONLINE FOOD DELIVERY AGE WISE DELIVERY FEEDBACK")

sns.histplot(data=dataframe, x="Gender",hue="Feedback of last order")

plt.show()
```



density plot

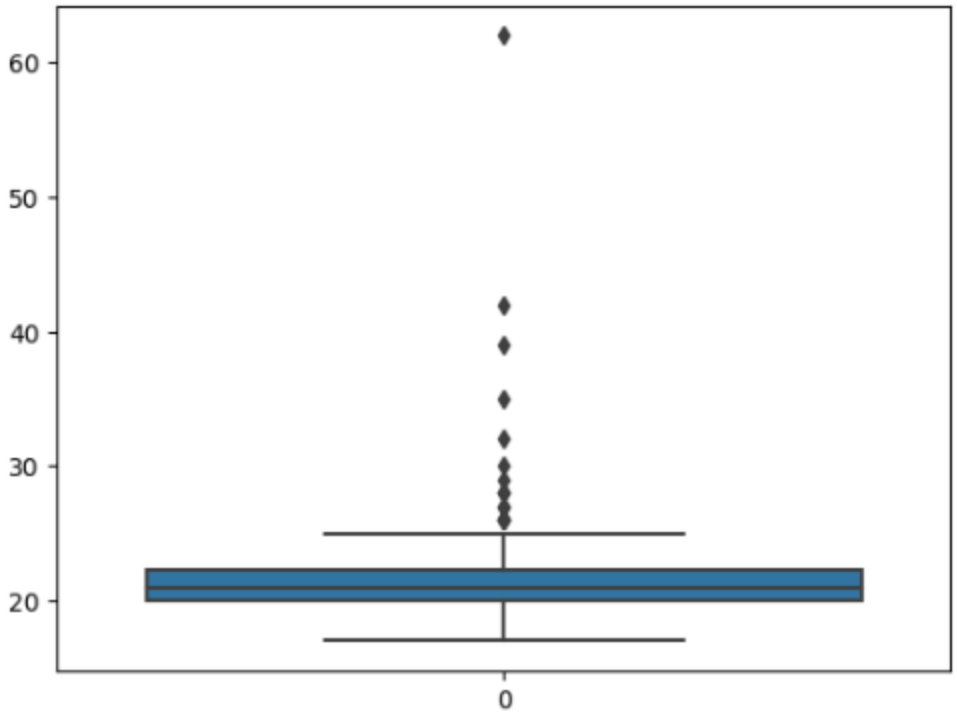
```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

data =pd.read_csv("Final_response.csv")

plt.figure(figsize=(15,10))

plt.title("Most used by the family")

sns.distplot(data['Member in family'])

plt.show()
```



boxplot

1. lower quartile(1st quartile), median and upper quartile (3rd quartile)
2. lowest and highest values
3. inter-quartile range (iqr) 25% for 1st quartile and 75% for 3rd quartile

```
box=sns.boxplot(data['Age '])
```

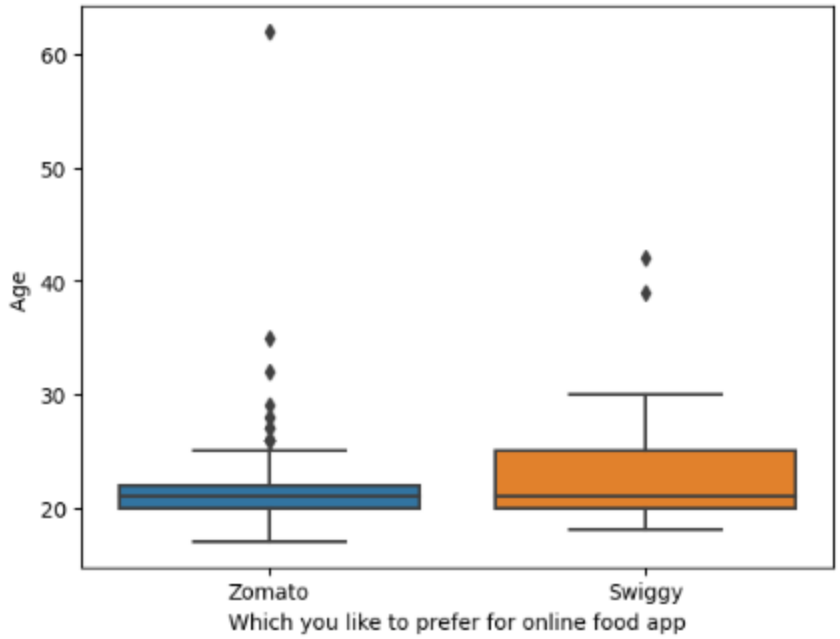


Tested Report

Here the median is 21 and thew highest outlier is 40-45. The highest value is 25, lowest value is 19

comparing the data between age and most app used

sns.boxplot(x='Which you like to prefer for online food app',y='Age ',data=data)



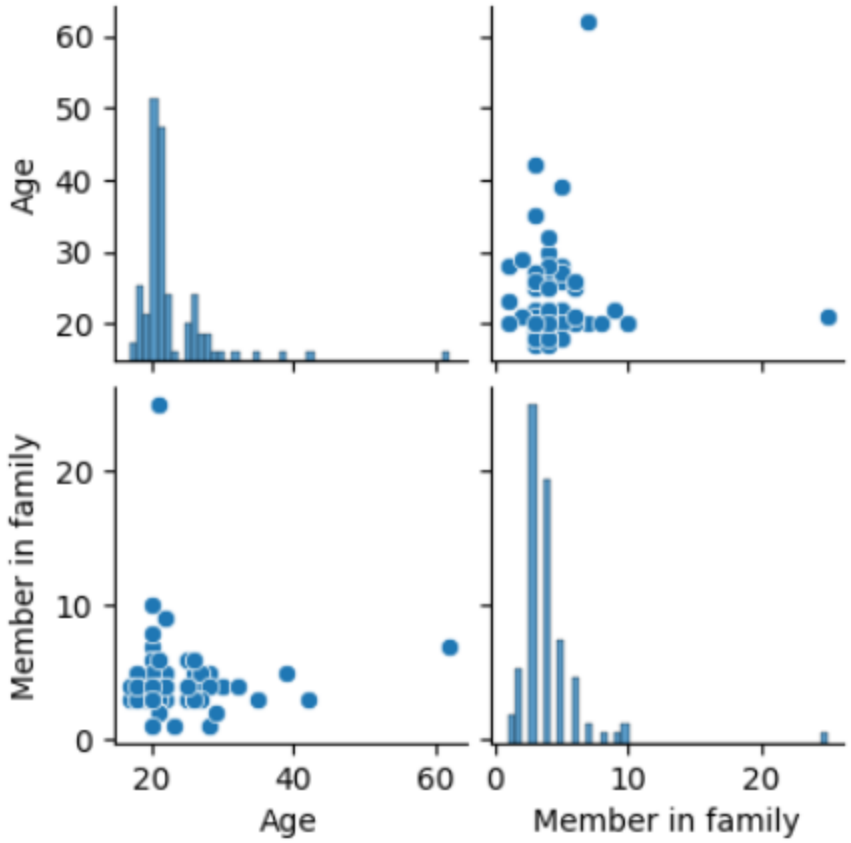
Zomato is most friendly app other than swiggy

Pair plot

influence_feature =['Age','Member in family']

sns.pairplot(data[influence_feature],height=2)

```
<seaborn.axisgrid.PairGrid at 0x2523e239870>
```



correlation and heat map

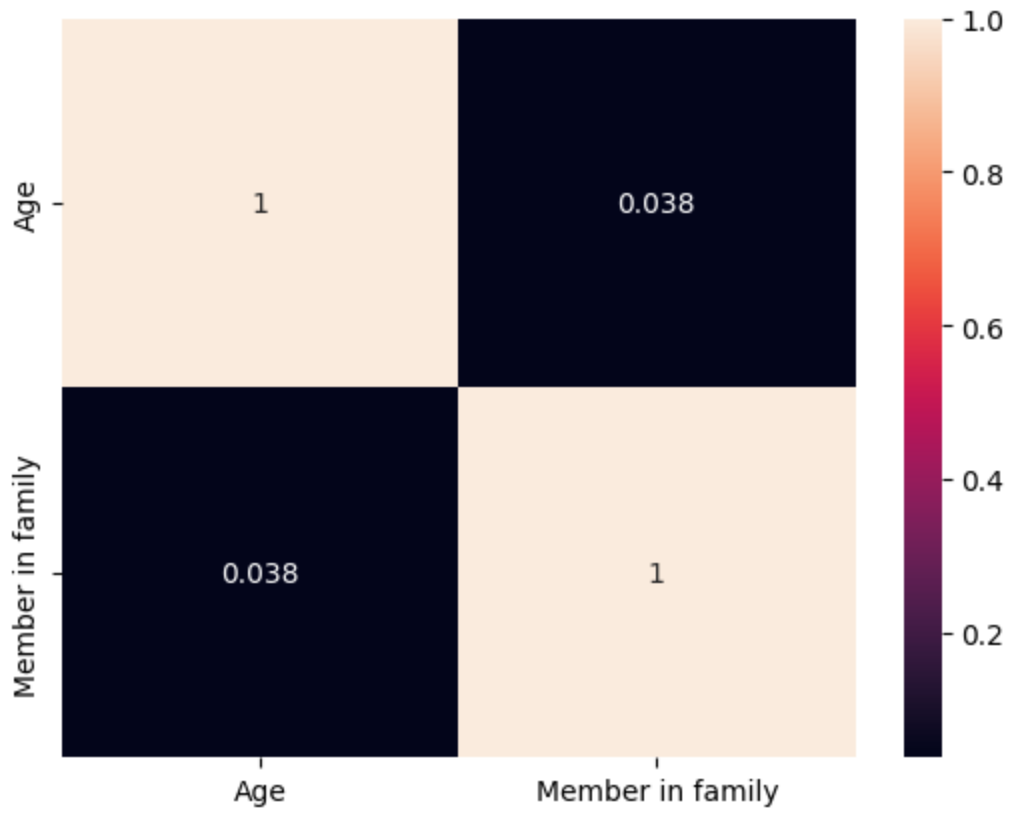
correlation is used for measuring the strength and direction of the linear relationship between continuous random varibles x and y.it is a statistical measure that indicates the extent to which variables change together The correlation value lies between -1.0 and 1.0 which indicates positive and negative correlation (perfect)

```
data[influence_feature].corr()
```

	Age	Member in family
Age	1.000000	0.037748
Member in family	0.037748	1.000000

sns.heatmap (data[influence_feature].corr(),annot=True)

<Axes: >



TRAIN THE DATASET

```
data["Gender"]=data["Gender"].map({"Male":1,"Female":2})

data["Marital status"]=data["Marital status"].map({"Unmarried":1,"Married":2})

data["Occupation"]=data["Occupation"].map({"Service":1,"Student":2,"Business":3})

data["Education Qualification"]=data["Education Qualification"].map({"Higher
Secondary":1,"Graduation":2,"Masters":3,"PHD":4})

data["Place"]=data["Place"].map({"Alipurduar":5,"Coochbehar":6,"Other":7})

data["Feedback of last order"]=data["Feedback of last order"].map({"Positive":1,"Negative":0})

print(data.head())
```

```
  Age  Gender  Marital status  Occupation  Education Qualification  \
0   26       1               1           1                        2.0
1   21       2               1           2                        2.0
2   20       1               1           2                        2.0
3   20       2               1           2                        2.0
4   21       1               1           2                        2.0

   Member in family  Place Which you like to prefer for online food app  \
0                 6    5.0                                        Zomato
1                 3    5.0                                        Zomato
2                 7    6.0                                        Zomato
3                 4    5.0                                        Zomato
4                 4    6.0                                        Zomato

   Feedback of last order
0                       1
1                       1
2                       1
3                       1
4                       1
```

Now let's train a machine learning model to predict which apps¶

does most of the time used will start by splitting the data into training and test sets:

#splitting data

import matplotlib.pyplot as plt

import numpy as np

from sklearn.model_selection import train_test_split

x = np.array(data[["Age","Gender","Marital status","Occupation","Education Qualification","Member in family","Place","Feedback of last order"]])

y = np.array(data[["Which you like to prefer for online food app"]])

training a machine learning model

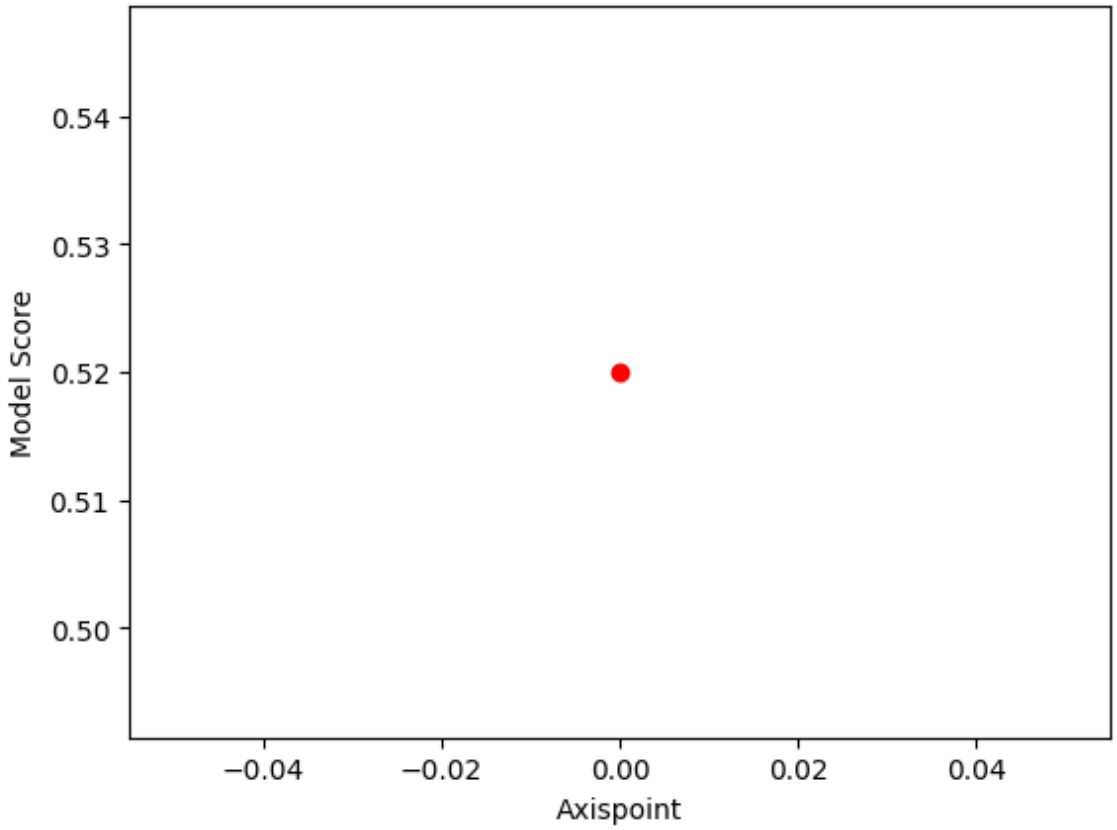
nan_value =np.nan_to_num(x)

#print(nan_value)

from sklearn.ensemble import RandomForestClassifier

```
xtrain, xtest, ytrain, ytest = train_test_split(nan_value, y,test_size=0.25,random_state=42)

model = RandomForestClassifier()

model.fit(xtrain, ytrain.ravel())

modelsocre = model.score(xtest, ytest)

print(modelsocre)

plt.plot([modelsocre],'ro')

plt.ylabel('Model Score')

plt.xlabel('Axispoint')

plt.show()
```

```
0.52
```



Predicting out come through a program

```
print("Enter the customer details below:-")

a = int(input("Enter the age of the customer:"))

b = int(input("Enter the gender (1=Male, 2=Female):"))

c = int(input("Enter the marital status(1=unmarried,2=married):"))

d = int(input("Enter the Occupation(1=Service,2=Student,3=Business):"))

e = int(input("Enter the Education Qualification(Higher Secondary:1,Graduation:2,Masters:3,PHD:4):"))

f = int(input("Enter the member in family:"))

g = int(input("Enter the place (Alipurduar:5,Coochbehar:6 ,Other:7):"))

h = int(input("Review of last order feedback (1= Positive,0=Negative):"))

features = np.array([[a, b, c, d, e, f, g,h]])

print("Finding Which app is used most familiar:",model.predict(features))
```

PREDICTION RESULT (Testing)

```
Enter the customer details below:-
Enter the age of the customer:26
Enter the gender (1=Male, 2=Female):1
Enter the marital status(1=unmarried,2=married):1
Enter the Occupation(1=Service,2=Student,3=Business):2
Enter the Education Qualification(Higher Secondary:1,Graduation:2,Masters:
3,PHD:4):2
Enter the member in family:3
Enter the place (Alipurduar:5,Coochbehar:6 ,Other:7):5
Review of last order feedback (1= Positive,0=Negative):1
Finding Which app is used most familiar: ['Swiggy']


Enter the customer details below:-
Enter the age of the customer:20
Enter the gender (1=Male, 2=Female):2
Enter the marital status(1=unmarried,2=married):1
Enter the Occupation(1=Service,2=Student,3=Business):2
Enter the Education Qualification(Higher Secondary:1,Graduation:2,Masters:
3,PHD:4):2
Enter the member in family:3
Enter the place (Alipurduar:5,Coochbehar:6 ,Other:7):5
Review of last order feedback (1= Positive,0=Negative):1
Finding Which app is used most familiar: ['Zomato']
```

CONCLUSION / TESTING REPORT:

Conclusion This algorithm helps to analysis the SWOT of any organization as per the feedback of the user or we can say the last order feedback the customer satisfaction .It will help the organization to increase the quality of their products in every parameters. So most of the user is used swiggy in near future

SWOT Analysis of an organization:

What Is a SWOT Analysis?

SWOT stands for Strengths, Weaknesses, Opportunities, and Threats, and so a SWOT analysis is a technique for assessing these four aspects of your business.

SWOT Analysis is a tool that can help you to analyze what your company does best now, and to devise a successful strategy for the future. SWOT can also uncover areas of the business that are holding you back, or that your competitors could exploit if you don't protect yourself.

A SWOT analysis examines both internal and external factors – that is, what's going on inside and outside your organization. So some of these factors will be within your control and some will not. In either case, the wisest action you can take in response will become clearer once you've discovered, recorded and analyzed as many factors as you can.

In this article, video and infographic, we explore how to carry out a SWOT analysis, and how to put your findings into action. We also include a worked example and a template to help you get started on a SWOT analysis in your own workplace.

Why Is SWOT Analysis Important?

SWOT Analysis can help you to challenge risky assumptions and to uncover dangerous blindspots about your organization's performance. If you use it carefully and collaboratively, it can deliver new insights on where your business currently is, and help you to develop exactly the right strategy for any situation.

For example, you may be well aware of some of your organization's strengths, but until you record them alongside weaknesses and threats you might not realize how unreliable those strengths actually are.

Equally, you likely have reasonable concerns about some of your business weaknesses but, by going through the analysis systematically, you could find an opportunity, previously overlooked, that could more than compensate.

Strengths What do you do well? What unique resources can you draw on? What do others see as your strengths?	**Weaknesses** What could you improve? Where do you have fewer resources than others? What are others likely to see as weaknesses?
Opportunities What opportunities are open to you? What trends could you take advantage of? How can you turn your strengths into opportunities?	**Threats** What threats could harm you? What is your competition doing? What threats do your weaknesses expose to you?

GANTT CHART:

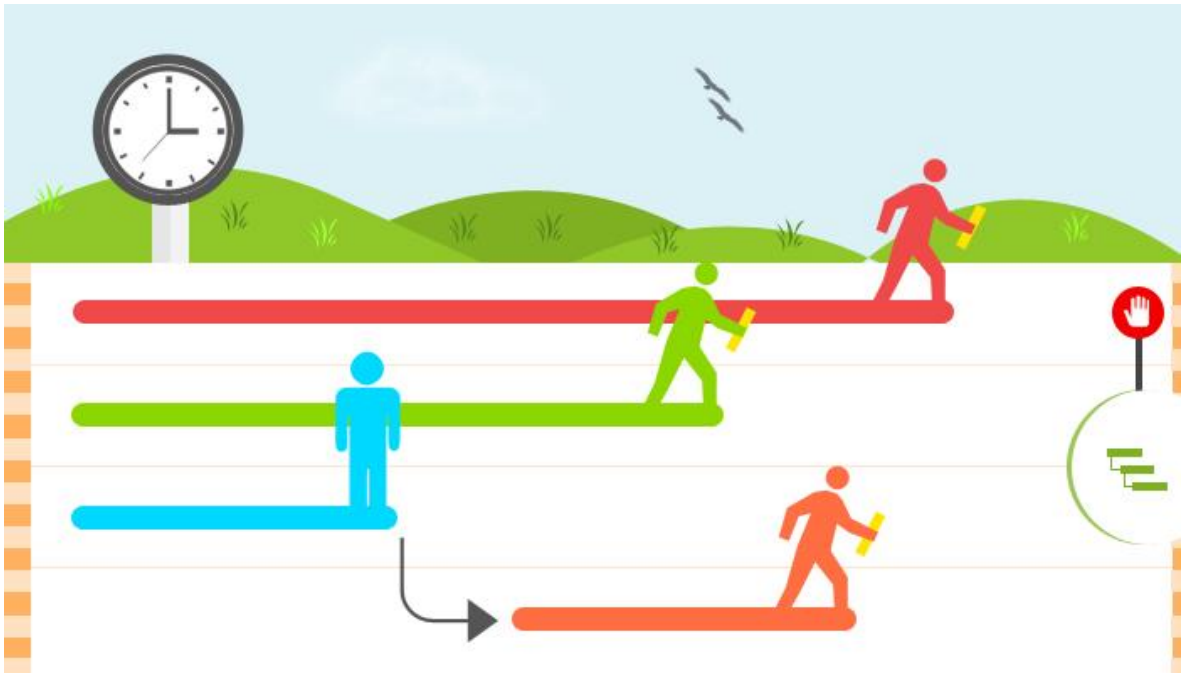
"A Gantt chart is a type of bar chart that illustrates a project schedule and shows the dependency relationships between activities and current schedule status."

In simpler words, Gantt charts are a visual view of tasks displayed against time. They represent critical information such as who is assigned to what, duration of tasks, and overlapping activities in a project.

All in all, Gantt charts are the perfect allies for planning, scheduling, and managing a project.

Gantt charts have an interesting history that needs to be shared. The origin of the tool dates back to the late 1800s when a Polish engineer Karol Adamiecki invented a diagram called harmonograph. The major reason behind inventing this diagram was to improve the way production schedules were displayed during that point of time. The only problem was that all findings and other critical information were published in Polish and Russian only.

USEFULNESS OF GANTT CHART



If you are wondering what is the **uses of the Gantt chart** and why it is useful for leaders, you need to know that these charts simplify the planning process. Since they are simple to create, use, and keep track of, they prove to be of great help for managers.

A Gantt Chart, in its simplest form, is a timeline that illustrates how the project will progress during the project management process. And the timeline view offered by the Gantt Chart app is proven quite useful for planning and scheduling projects. It helps project managers and project teams to assess how long a project should take, determine the resources needed, understand the dependencies between tasks, and plan the order in which each task will be completed if the whole project is to deliver on time.

As the project moves forward, a Gantt chart adjusts simultaneously, displaying an up-to-date project schedule to keep everyone (teams, clients, and stakeholders) informed of the progress. In addition to this, Gantt charts replace meetings and enhance other status updates. They make it easy for teams to understand task progress and talk about the problems they may encounter.

In all, using a Gantt chart or a Gantt chart tool is useful during project management because it allows you to picture your project against the time frame it needs to be done in.

Features of Gantt charts

- ***DUE DATES*:** The dates are one of the most essential aspects of a Gantt chart since they show project managers not only when the project will start and end, but also when each job will take place. These are shown at the top of the graph.
- ***TASKS:*** Major projects always seem to have a lot of sub-tasks. A Gantt chart assists project managers in keeping track of all sub-tasks in a project so that nothing is overlooked or delayed. The tasks are displayed on the left side of the page.
- ***MILESTONES:*** Milestones are tasks that are critical to the completion and success of a project. Unlike the minor things that must also be completed, completing a milestone provides a sense of accomplishment and progress. At the conclusion of each block on a Gantt chart, milestones are represented by different shapes or icons.
- ***BARS:*** Bars are used to represent the time frame in which each task should be performed after the subtasks have been stated. This ensures that each sub-task is finished on time, ensuring that the overall project is completed on time.
- ***TASKBARS:*** While many sub-tasks can be accomplished pretty quickly, there will be occasions when you want to know how your project is progressing at a glance. The taskbars are shaded to represent the portion of each task that has previously been performed, indicating progress.
- ***DEPENDENCIES:*** In a project, there are some tasks and subtasks that are dependent on one another for success. For instance, a task must be completed before another task can begin or terminate. On a Gantt chart, task dependencies represent this type of relationship. Small arrows between the taskbars are generally used to show these relationships.

- ***TASK ID:*** You probably have numerous tasks going on at the same time in today 's hyper work world. The task ID is included on the Gantt chart to help everyone involved readily identify the task you're discussing.

Benefits

1. Know what's going on in your projects

The biggest advantage of using free online Gantt charts in project management is that you get to see everything related to project at a single place. It acts as a great visualization and prioritization tool as it provides the total overview of the project and tells you about the critical information such as members involved in each step, the order of tasks, duration, start-to-end dates, task dependencies and progress made in them. Hence, online Gantt charts software are helpful in equipping project managers with the information they need to oversee while managing projects.

2. Improved communication and team cohesion

Communication is an integral part of a project that can make or break it. In fact, 86% of employees feel lack of communication as the biggest reason for workplace failures. On the other hand, Gantt charts are known to provide crystal clear communication. Project managers can use these charts to know who is working on what and give inputs on certain tasks and pass relevant information to them. This helps him to communicate better with team members and also improves their relationship as a team. Most importantly, it eliminates the need for having a separate tool for tracking and communication purposes.

3. Avoid resource overload

Too many problems arise when resources are stretched over too many tasks and processes. Gantt charts allow you to use your resources effectively as you get to see a project's timeline where you can easily

see how and where resources are being utilized. Within Gantt charts, you can also delegate tasks and align resources without burdening them with too much work. This way you can effectively manage resources and when resources are properly managed, projects are more likely to be completed within budget and deadlines.

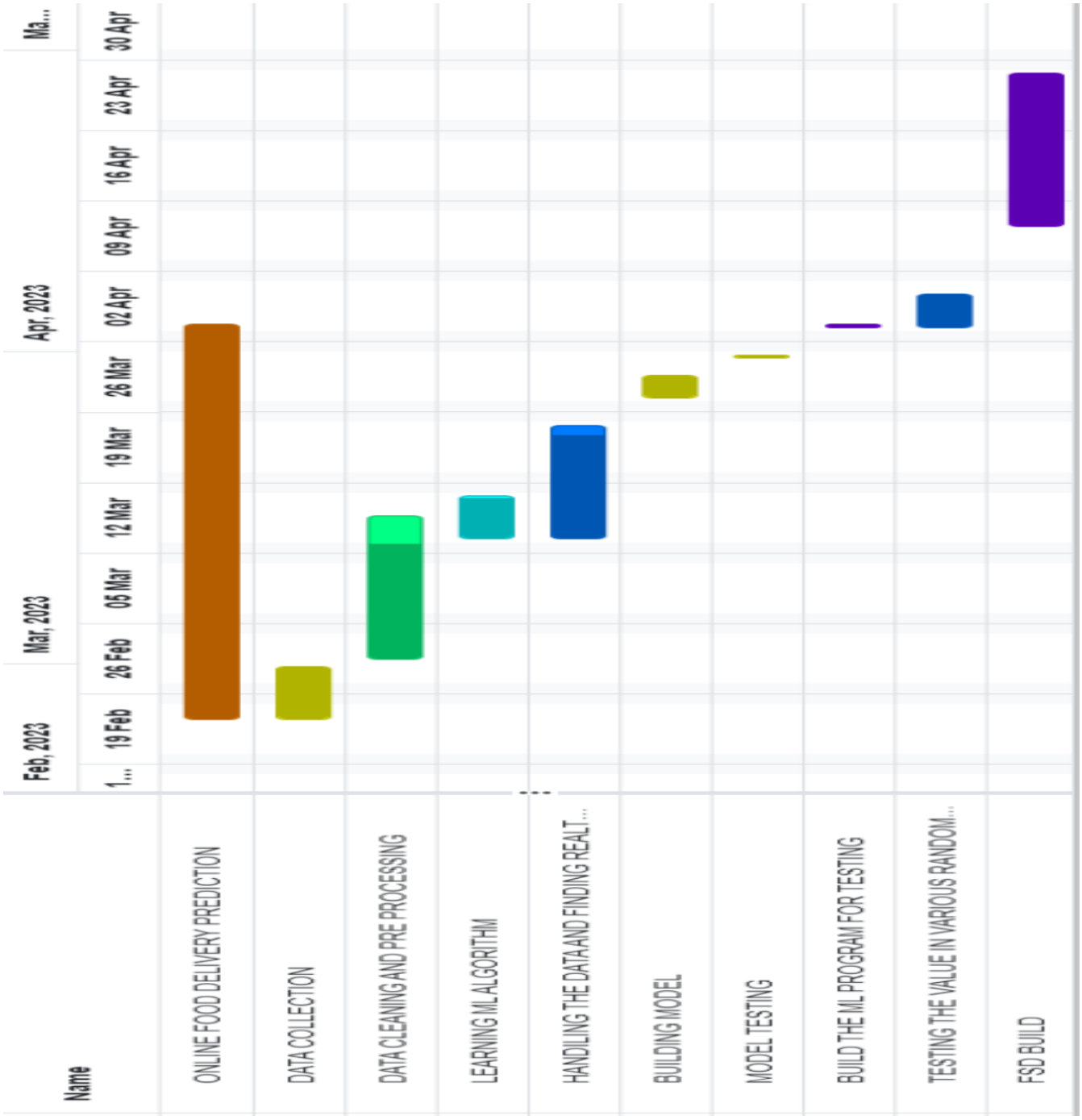
4. Measure the progress of projects

Once you schedule your project in a Gantt chart software you can check the progress of the projects in it. The feature to measure a project's progress enables you to make changes in your strategy if it is lagging behind its schedule. As project members can update the completed percent of each task, everyone stays on the same page. This information is quite useful to determine if the project is headed in the right direction or not.

5. See overlapping activities and task dependencies

At times, you cannot start a certain task unless another task which is dependent on the former is completed. Such task dependencies can make things difficult to be managed especially in case of complex projects that have too many tasks. As you get to see an overview of projects in a Gantt chart software, you can easily see which tasks are dependent on each other and schedule them in a way that they don't impact the overall progress in any way and team members share equal responsibility.

In our project **(Online Food Delivery prediction)** the risk is minimum most of the we have taken minimum span of time (12 weeks) to deploy the project and testing also. The track report or the Gantt Chart given below .According to the task analysis.


Name
Feb, 2023
Mar, 2023
Apr, 2023
Ma...
1...
19 Feb
26 Feb
05 Mar
12 Mar
19 Mar
26 Mar
02 Apr
09 Apr
16 Apr
23 Apr
30 Apr
ONLINE FOOD DELIVERY PREDICTION
DATA COLLECTION
DATA CLEANING AND PRE PROCESSING
LEARNING ML ALGORITHM
HANDLING THE DATA AND FINDING REALT...
BUILDING MODEL
MODEL TESTING
BUILD THE ML PROGRAM FOR TESTING
TESTING THE VALUE IN VARIOUS RANDOM...
FSD BUILD

Bibliography:

1. Et.al, Osman, M. H. "Ambi Detect: An Ambiguous Software Requirements Specification Detection Tool." *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* 12, no. 3 (April 10,2021):2023–28. http://dx.doi.org/10.17762/turcomat.v12i3.1066.

2. Mahalakshmi, K., Udayakumar Allimuthu, L. Jayakumar, and Ankur Dumka. "A Timeline Optimization Approach of Green Requirement Engineering Framework for Efficient Categorized Natural Language Documents in Non-Functional Requirements." *International Journal of Business Analytics* 8, no. 1 (January 2021): 21–37. http://dx.doi.org/10.4018/ijban.2021010102.

3. Hayman Oo, Khin, Azlin Nordin, Amelia Ritahani Ismail, and Suriani Sulaiman. "An Analysis of Ambiguity Detection Techniques for Software Requirements Specification (SRS)." *International Journal of Engineering & Technology* 7, no. 2.29 (May 22, 2018): 501. http://dx.doi.org/10.14419/ijet.v7i2.29.13808.

About Us

"Hello we are Developers team"

Soumyadip Saha Chaudhuri:

I am Soumyadip Saha Chaudhuri My hobby is traveling, plying cricket,witching movies and listening to music seeking to learn various technology about IT.

Saikat Dhar :

I am saikat Dhar, My hobbies are plying cricket, witching movies and listening to music

Anamika Sarkar :

I am Anamika Sarkar, My hobby is traveling, seeking to learn various technology about IT.

Srija Bhattacharjee :

I am Srija Bhattacharjee, My hobby is traveling, cooking, seeking to learn various technology about IT.

Taniya Nandi :

I am Taniya Nandi. In my hobbies I like painting Mandala Arts, Glass Painting. Learned new skills about computer programming language.c

Joydeb Barai :

I am Joydeb Barai , my hobby is seeking to learn various technology about IT. I would like to play cricket and watching Youtube videos and also playing Clash of Clan with my friends.

Uday Dey :

I am Uday Dey my hobbies are I like painting Mandala Arts, Glass Painting and graphics designing. Seek Learned new skills about computer programming language.c Playing Guitar duringfree time.

Arijit Roy :

I am Arijit Roy, my hobbies are watching documentary, chasing new skills. I would lick to play outdoor game and watching cricket.

Subhankar Paul :

I am Subhankar Paul, my hobby is seeking to learn various technology about IT. I would like to watching Youtube videos and also playing Clash of Clan and Free fire with my friends.

Aritri Mitra :

I am Aritri Mitra. In my hobbies I love cooking, gardening in my free times. I would like to play Clash Of Clan with my friends.

MACHINE LEARNING WITH PYTHON

plotly

pandas

DATM

DATM