PROFESSIONAL INDUSTRIAL TRAINING/

INTERNSHIP REPORT

OF

“PYTHON WITH MACHINE LEARNING”

AT

FAME WORLD EDUCATION HUB
LUCKNOW

AN INDUSTRY INTERNSHIP REPORT SUBMITTED
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF DEGREE OF

BACHELOR OF ENGINEERING In COMPUTER SCIENCE & ENGINEERING SUBMITTED BY

TAWHEED MUSAIB FAFOO
Roll Number: 2020A1T023

CANDIDATES DECLARATION

I, **Tawheed musaib fafoo, 2020A1T23,** hereby declare that the work which is being presented in the Industry Internship Report entitled, “**AMAZON SELLER: ORDER STATUS PREDICTION**” in partial fulfilment of requirement for the award of degree of B.E. (Computer Science & Engineering) and submitted in the Computer Science & Engineering department, Model Institute of Engineering and Technology (Autonomous), Jammu is an authentic record of my own work carried by me at “Fame World Education hub Lucknow” under the supervision and mentorship of

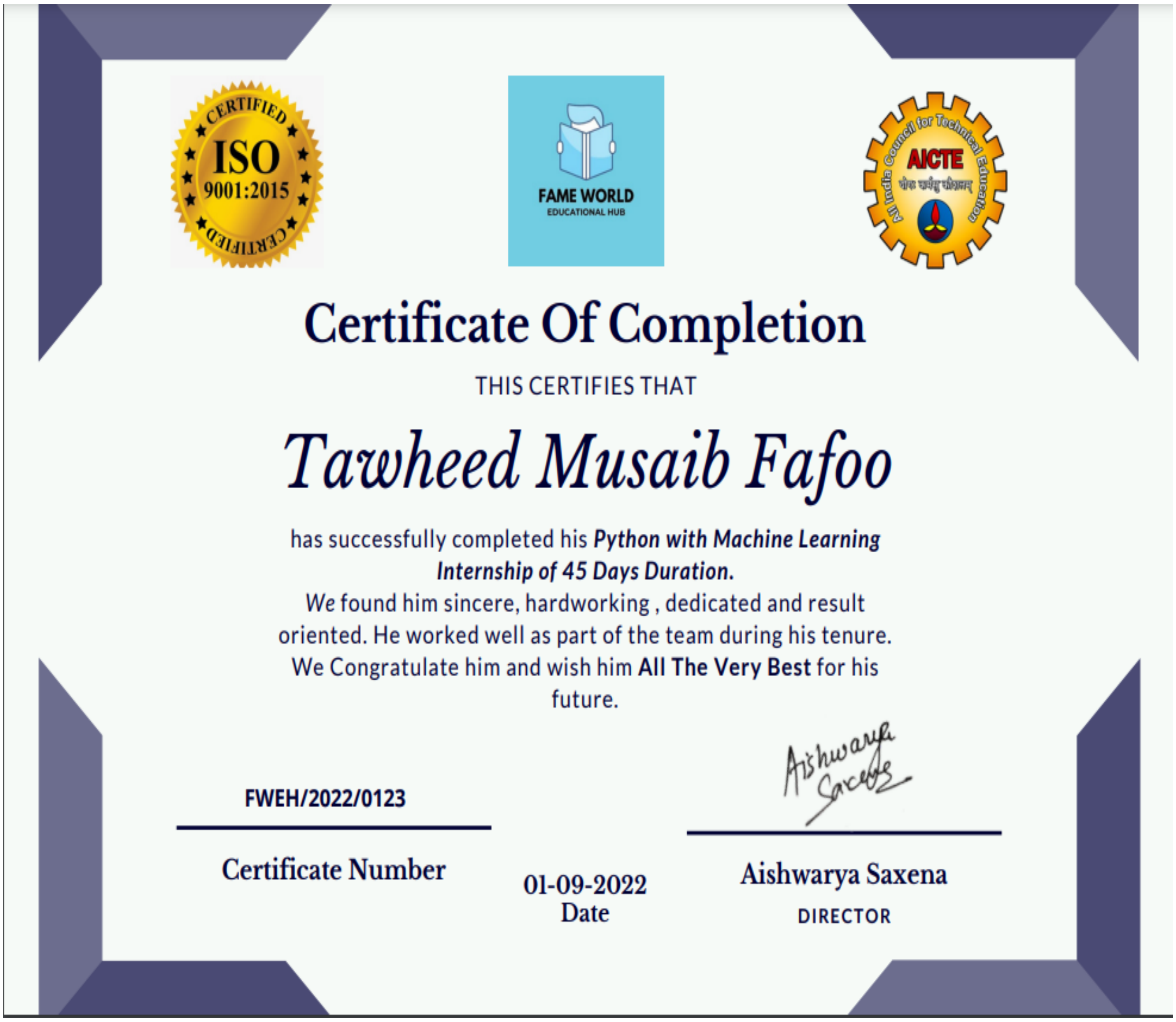
Ms. Aishwarya Saxena. The matter presented in this report has not been submitted in this or any other University / Institute for the award of B.E. Degree.

Musaib.

Signature of the Student *Dated*: 11/10/2022

(Tahwheed Musaib Fafoo)
2020A1T023

INTERNSHIP CERTIFICATE



CERTIFIED ISO 9001:2015 CERTIFIED

FAME WORLD
EDUCATIONAL HUB

All India Council for Technical Education AICTE

Certificate Of Completion

THIS CERTIFIES THAT

Tawheed Musaib Fafoo

has successfully completed his ***Python with Machine Learning Internship of 45 Days Duration.***
We found him sincere, hardworking , dedicated and result oriented. He worked well as part of the team during his tenure. We Congratulate him and wish him **All The Very Best** for his future.

FWEH/2022/0123
Certificate Number

01-09-2022
Date

Aishwarya Saxena
Aishwarya Saxena
DIRECTOR

Department of Computer Science & Engineering

Model Institute of Engineering and Technology (Autonomous)

Kot Bhalwal, Jammu, India

(NAAC "A" Grade Accredited)

Ref. No.: 2020A1T023 **Date: 11/10/2022**

CERTIFICATE

Certified that this Industry Internship Report entitled **"MACHINE LEARNING REPORT"** is the bonafide work of "**Tawheed Musaib, Roll No.- 2020A1T023, of 5th Semester, CSE, Model Institute of Engineering and Technology (Autonomous), Jammu",** who carried out the Industry Internship at "**Fame World Education hub LUCKNOW**" work under my mentorship during 11 July, 2022 – 16 Aug, 2022.

Dr. Mekhla Sharma
Assistant Professor
CSE, MIET

This is to certify that the above statement is correct to the best of my knowledge.

Prof. (Dr.) Ashok Kumar
Dean Academic Affairs &
Head of the Department
CSE, MIET

ACKNOWLEDGEMENTS

The internship opportunity I had with “Fame World Education hub” was a great chance for learning and professional development. Therefore, I consider myself as a very lucky individual as I was provided with an opportunity to be a part of it. I am also grateful for having a chance to meet so many wonderful people and professionals who led me though this internship period.

Bearing that in mind, I am using this opportunity to express my deepest gratitude and special thanks to trainer, Ms Aishwarya Saxena, who in spite of being extraordinarily busy with his duties, took time out to hear, guide and keep me on the correct path and allowing me to carry out my project at their esteemed organization and extending during the training.

I am also highly indebted to Prof. (Dr.) Ankur Gupta, Director MIET and Prof. (Dr.) Ashok Kumar, Dean Academics Affair MIET / H.O.D - CSE Department for the facilities provided to accomplish this internship.

I would also like to extend my supreme gratitude to Asst. Prof. Shafalika Vijayal and Asst. Prof. Saurabh Sharma for putting their hard efforts to make this internship program achieve its desired goals.
I would also like to thank my parents, friends etc. who helped me in my Industry Internship.

I express my sincere gratitude to Model Institute of Engineering and Technology (Autonomous), Jammu for giving me the opportunity to work on Industry Internship during my third year of B.E.

At the end thanks to the Almighty for helping me starting from the beginning to the end of my interspersion.

Tawheed Musaib Fafoo

Roll Number – 2020A1T023

SELF EVALUATION

I am a 3rd year B.E. undergraduate student pursuing Computer Science and Engineering at Model Institute of Engineering and Technology, Jammu. I recently completed an internship with "FAME WORLD EDUCATION HUB",Lucknow as a Machine Learning Developer Intern.

At Fame World, I learned quite a bit about Python programming language and its applications in day-to-day lives. Also, I learned about the Machine Learning concepts and algorithms in a very easy and efficient manner.

I was also provided with multiple assessments during my internship, which I always completed on time with full dedication and zeal. I still experienced a learning curve due to this being my first exposure to this kind of work. By the end of my internship, however, I felt comfortable in completing my assigned tasks and even received reviews from team leaders expressing their opinions about my work.

I developed great communication skills with people and this helped me to be a good team member when difficult situations aroused in meeting a deadline or solving a problem. Teamwork is valuable to me because I welcome co-worker insights into these types of challenges.

I totally understand the importance of regular practice and learning conceptual theories while being a CS student. And due to this internship opportunity, I got the chance to learn the topics not only theoretically but practically too. I got a firmer grasp on the coding part and learned a lot of new concepts.

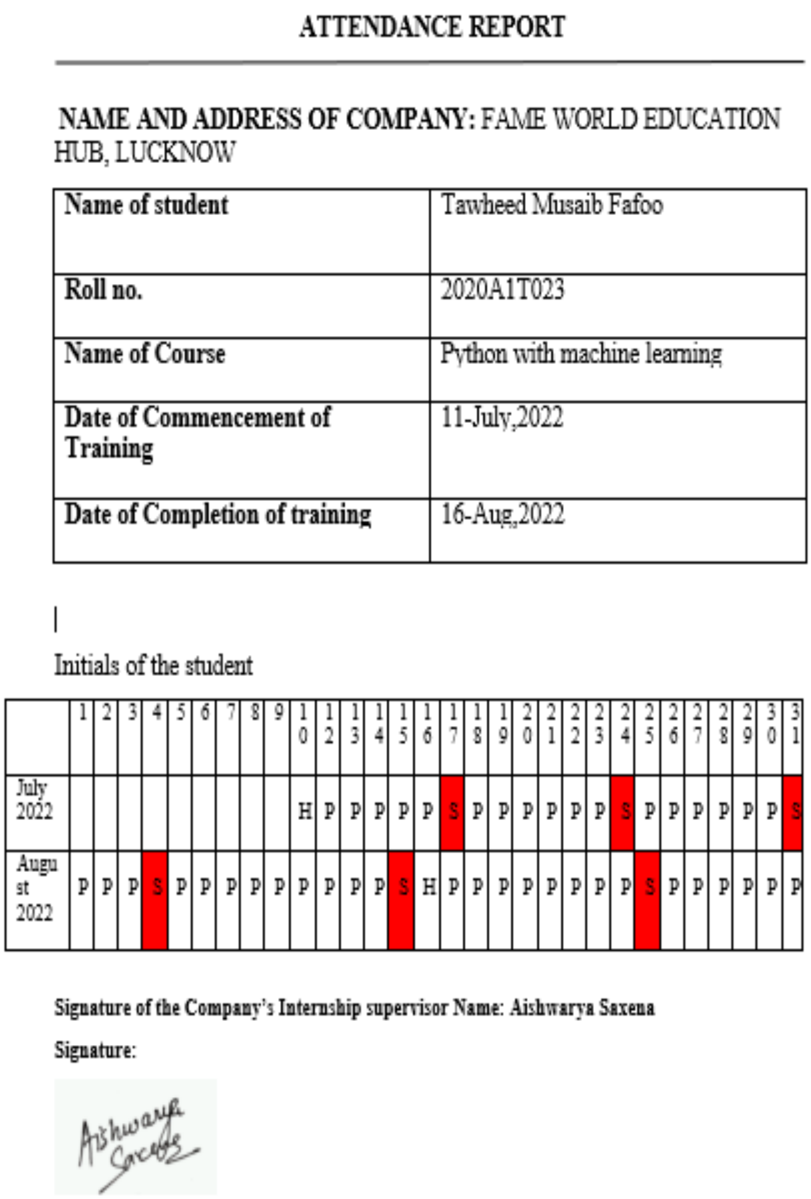
While working as a Machine Learning Intern at Fame World, I gained a newer kind of experience which is surely going to help me for my future endeavours.

Tawheed Musaib Fafoo

Roll no. – 2020A1T023

CSE - 5th Sem

ATTENDANCE REPORT



ATTENDANCE REPORT

NAME AND ADDRESS OF COMPANY: FAME WORLD EDUCATION HUB, LUCKNOW

Name of student	Tawheed Musaib Fafoo
Roll no.	2020A1T023
Name of Course	Python with machine learning
Date of Commencement of Training	11-July,2022
Date of Completion of training	16-Aug,2022

|

Initials of the student

	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
July 2022										H	P	P	P	P	P	S	P	P	P	P	P	P	S	P	P	P	P	P	P	S
August 2022	P	P	P	S	P	P	P	P	P	P	P	P	P	S	H	P	P	P	P	P	P	P	P	S	P	P	P	P	P	P

Signature of the Company's Internship supervisor Name: Aishwarya Saxena

Signature:

Aishwarya Saxena

ABSTRACT

This report contains a detailed overview of my internship learning and project implementation during my internship training at EME Technologies.

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without being explicitly programmed.

Every time a web search engine like Google is used to search the internet, one of the reasons that work so well is because a learning algorithm that has learned how to rank web pages. These algorithms are used for various purposes like data mining, image processing, predictive analytics, etc. to name a few.

During my internship, I learned a lot about Machine Learning and its applications in day-to-day lives. I have also learned how to work in a corporate space which is going to be very beneficial for my career. I have applied all my knowledge to the project which I created at the end of the internship.

In my internship, I create a project who's aims are to develop a Machine learning model which would predict the order status. (i.e., DILVERED TO BUYER OR RETURNED TO SELLER)

The model various the dataset downloaded from Kaggle and analysis various factors on which the order status depends and on the bases of that make the prediction.

The model also performs the data cleaning and visualization of the data through the various graphs

Contents

\

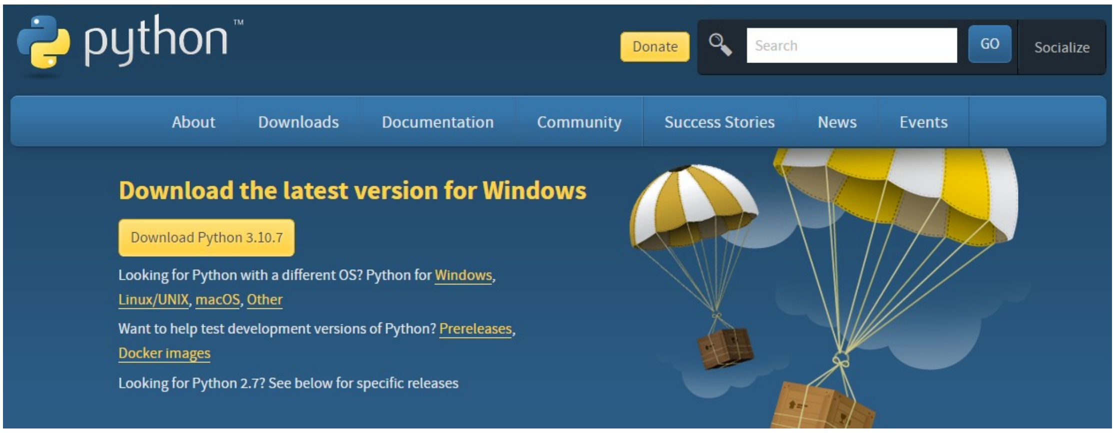
Chapter 1 PYTHON

Installation and Working with Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

For working on Python, if you don’t already have a copy of Python installed on your computer, you will need to open up your Internet browser and go to the Python download page (http://www.python.org/download/).



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

Want to help test development versions of Python? Prereleases, Docker images

Looking for Python 2.7? See below for specific releases

Figure 1.1 – Downloading Python[10]

Now that you are on the download page, select which of the software builds you would like to download.

Once the installer is downloaded, run the Python installer. Check the Install launcher for all users check box. Further, you may check the Add Python 3.7 to path check box to include the interpreter in the execution path.

Select Customize installation. Add Python to environment variables. After selecting the Advanced options, click Install to start installation. Once the installation is over, you will see a Python Setup Successful windows.

You have now successfully installed Python.

Understanding Python blocks, Variables, Datatypes, Operations, Loops

Datatypes: Python has many native data types. Here are the important ones:
Booleans are either True or False.
Numbers can be integers (1 and 2), floats (1.1 and 1.2), fractions (1/2 and 2/3), or even complex numbers.
Strings are sequences of Unicode characters, e.g., an HTML document.
Bytes and byte arrays, e.g., a JPEG image file.
Lists are ordered sequences of values.
Tuples are ordered, immutable sequences of values.
Sets are unordered bags of values.

Variables: Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.
Example:

```
counter = 100 # An integer assignment
miles = 1000.0 # A floating point name
= "John" # A string
```

Conditional Statements: Decision making is anticipation of conditions occurring while execution of the program and specifying actions taken according to the conditions.
Decision structures evaluate multiple expressions which produce TRUE or FALSE as outcome. You need to determine which action to take and which statements to execute if outcome is TRUE or FALSE otherwise.
Python programming language provides following types of decision-making statements.

Loop Type Description

if statements -

An if statement consists of a Boolean expression followed by one or more statements.

if...else statements -

An if statement can be followed by an optional else statement, which executes when the Boolean expression is FALSE.

Nested if statements -

You can use one if or else if statement inside another if or else if statement(s).

Operations: There are 3 types of operations-

- Arithmetic Operations: Arithmetic operators are used to performing mathematical operations like addition, subtraction, multiplication, and division.

  Example -

  ```
  add = a + b      # Addition of numbers
  sub = a - b      # Subtraction of numbers
  mul = a * b      # Multiplication of number
  div1 = a / b     # Division(float) of number
  div2 = a // b    # Division(floor) of number
  mod = a % b      # Modulo of both number
  p = a ** b       # Power
  ```

- Logical Operations: Logical operators perform Logical AND, Logical OR, and Logical NOT operations. It is used to combine conditional statements.

  Example –

  ```
  a = True
  b = False
  print(a and b)     # Print a and b is False
  print(a or b)      # Print a or b is True
  ```

```
print(not a)        # Print not a is False
```

· Comparison Operations: Comparison of Relational operators compares the values. It either returns True or False according to the condition.

Example –

```
a=13
b = 33
  # a > b is False
  print(a > b)
  # a < b is True
   print(a < b)

   # a == b is False
   print(a == b)
   # a != b is True
   print(a != b)
   # a >= b is False
   print(a >= b)
  # a <= b is True
   print(a <= b)
```

· Assignment Operators: Assignment operators are used to assign values to the variables.

Example –

```
a = 10
 b = a print(b)       # Assign value
 b += a print(b)     # Add and assignvalue
 b -= a print(b)     # Subtract and assign value
 b *= a print(b)     # Multiply and assign
```

```
b <<= a print(b)   # Bitwise shift operator
```

- Identity Operators: ‘is’ and ‘is not’ are the identity operators both are used to check if two values are located on the same part of the memory. Two variables that are equal do not imply that they are identical.

  Example –

  ```
  a = 10
  b = 20
  c = a
  print(a is not b)
  print(a is c)
  ```

- Membership Operators: ‘in’ and ‘not in’ are the membership operators; used to test whether a value or variable is in a sequence.

  Example –

  ```
  # Python program to illustrate not 'in'
  operator x = 24 y = 20 list = [10, 20, 30, 40,
  50]

  if (x not in list):
      print("x is NOT present in given list")
  else:
      print("x is present in given list")

  if (y in list):
      print("y is present in given list")
  else:
      print("y is NOT present in given list")
  ```

· Bitwise Operators: Bitwise operators act on bits and perform the bit-by-bit operations. These are used to operate on binary numbers.

Example –

```
a = 10
b = 4

# Print bitwise AND operation
print(a & b)

# Print bitwise OR operation
print(a | b)

# Print bitwise NOT operation
print(~a)

# print bitwise XOR operation
print(a ^ b)

# print bitwise right shift operation
print(a >> 2)

# print bitwise left shift operation
print(a << 2)
```

Transfer Statements –

Transfer statements alter the way a logic gets executed. These statements are often used in loops for and while. These are:

1. Break:

   We can use break statement inside loops to break loop execution based on some condition.

E.g.:

```
        for i in range(10):
            if i==7:
                print("Processing is enough. Please break the loop")
                break
    print(i)
```

2. Continue:

   We use continue statement to skip current iteration and continue next iteration.

E.g.: To print odd numbers in the range 0 to 9

```
  for  i in range(10):
            if i%2==0:
              continue
       print(i)
```

3. Pass:

   Pass is a keyword in Python. In our programming syntactically if block is required which won't do anything then we can define that empty block with pass keyword.

   This statement does nothing. It is used to define an empty block of code or a class. When written in a loop statement, it's usually the last statement.

E.g.:

```
numbers  =  [10,11,12,13,14]
for num in numbers:
    if num%2 == 0:
print(num)
```

```
else
    pass
```

Python Functions, Modules, Packages, List and Dictionary Manipulation

Functions: Function blocks begin with the keyword def followed by the function name and parentheses (( )). Any input parameters or arguments should be placed within these parentheses. The code block within every function starts with a colon (:) and is indented.

Its Syntax –

```
def functionname( parameters ):
 "function_docstring
 "   function_suite
 return [expression]
```

Modules: A module is a file containing Python definitions and statements. Modules can import other modules. It is customary but not required to place all import statements at the beginning of a module. [3]

Packages: Packages are a way of structuring Python’s module namespace by using “dotted module names”. For example, the module name A.B designates a submodule named B in a package named A. Just like the use of modules saves the authors of different modules from having to worry about each other’s global variable names, the use of dotted module names saves the authors of multi-module packages like NumPy or Pillow from having to worry about each other’s module names.

Lists: A list is exactly what it sounds like. Lists can contain any kind of objects, as long as they're between square brackets. In lists:

- the order stays the same
- you can get an item by referring to its position in the list, a number called the index

```
Example: cities = ['Tokyo', 'Los Angeles', 'New York', 'San Francisco']
print cities # see the specific list object this variable refers to
Output- ['Tokyo', 'Los Angeles', 'New York', 'San Francisco']
cities[1]

            Output- 'Los Angeles'
```

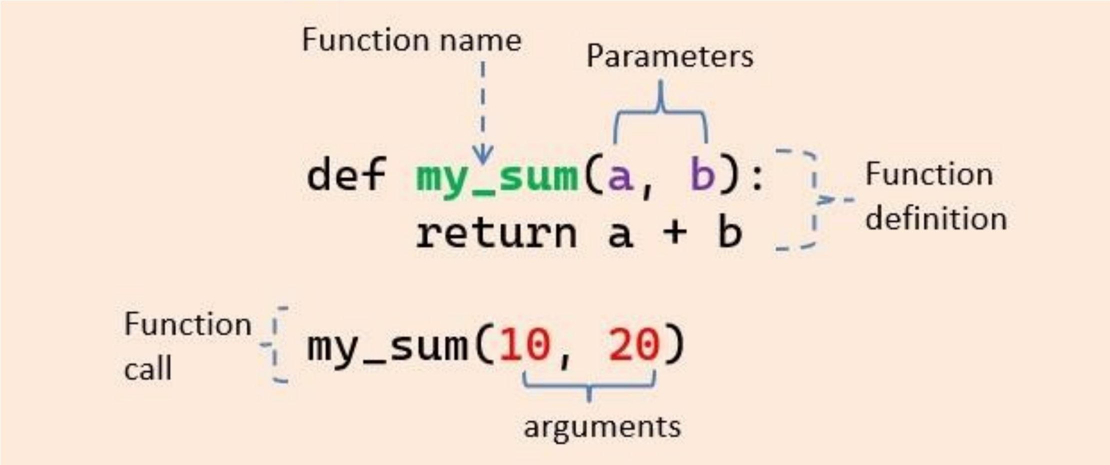
Dictionary: Dictionaries are also what they sound like - a list of definitions that correspond to unique terms. [4]

- Dictionaries are unordered
- Dictionary values are accessed by keys

```
Example: city_population = {
   'Tokyo': 13350000, # a key-value pair
   'Los Angeles': 18550000,
    'New York City': 8400000,
    'San Francisco': 1837442,
}
city_population['New York City']
Output- 8400000
```

Parameters and Arguments

Parameters are the variable listed inside the parentheses in the function definition. While as Arguments are the values sent to the function when it is called. It is the data on which the function performs some action and returns the result.



```
def my_sum(a, b):
    return a + b

my_sum(10, 20)
```

Figure 1.2. - Example of Parameters and Arguments [16]

Types of function arguments:

There are various ways to use arguments in a function. In Python, we have the following 4 types of arguments:

1. Default argument

2. Keyword arguments (named arguments)
3. Positional arguments
4. Arbitrary arguments (variable-length arguments *args and **kwargs)

1. Default argument: In a function, arguments can have default values. We assign default values to the argument using the '=' (assignment) operator at the time of function definition. You can define a function with any number of default arguments.

The default value of an argument will be used inside a function if we do not pass a value to that argument at the time of the function call. Due to this, the default arguments become optional during the function call.

It overrides the default value if we provide a value to the default arguments during function calls.

2. Keyword arguments: Keyword arguments are those arguments where values get assigned to the arguments by their keyword (name) when the function is called. It is preceded by the variable name and an (=) assignment operator. The Keyword Argument is also called a named argument.

Python allows functions to be called using keyword arguments. But all the keyword arguments should match the parameters in the function definition. When we call functions in this way, the order (position) of the arguments can be changed.

3. Positional arguments: Positional arguments are those arguments where values get assigned to the arguments by their position when the function is called. For example, the 1st positional argument must be 1st when the function is called. The 2nd positional argument needs to be 2nd when the function is called, etc. By default, Python functions are called using the positional arguments.

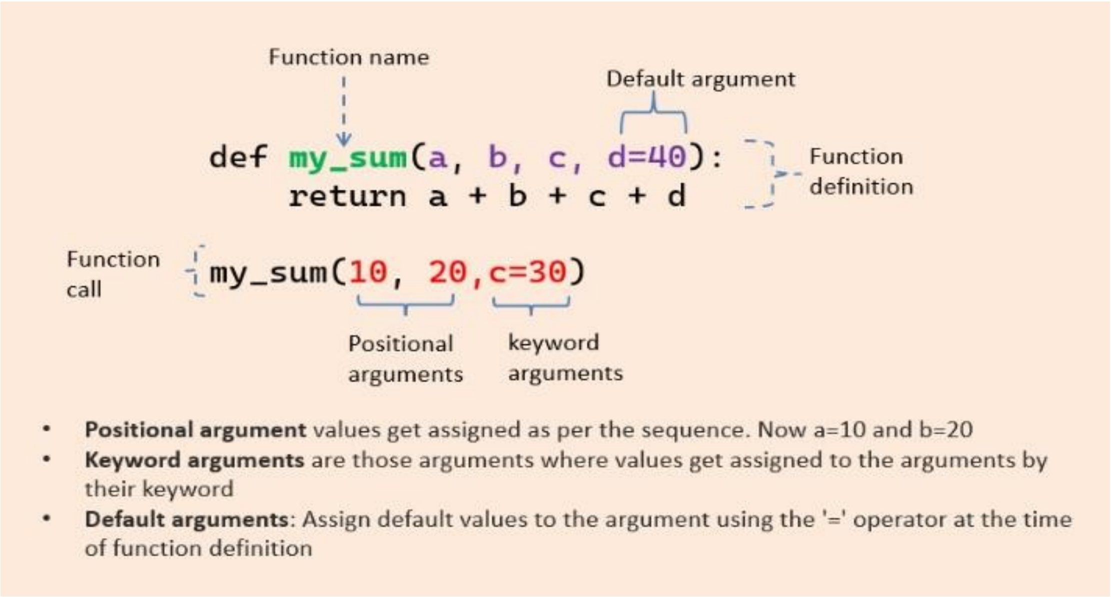
4. Arbitrary positional arguments(*args): We can declare a variable-length argument with the * (asterisk) symbol. Place an asterisk (*) before a parameter in the function definition to define an arbitrary positional argument.

we can pass multiple arguments to the function. Internally all these values are represented in the form of a tuple.

5. Arbitrary keyword arguments (**kwargs): The **kwargs allow you to pass multiple keyword arguments to a function. Use the **kwargs if you want to handle named arguments in a function.

Use the unpacking operator (**) to define variable-length keyword arguments.

Keyword arguments passed to a kwargs are accessed using key-value pair.



```
def my_sum(a, b, c, d=40):
    return a + b + c + d

my_sum(10, 20,c=30)
```

- **Positional argument** values get assigned as per the sequence. Now a=10 and b=20
- **Keyword arguments** are those arguments where values get assigned to the arguments by their keyword
- **Default arguments**: Assign default values to the argument using the '=' operator at the time of function definition

Figure 1.3. - Types of Arguments [16]

Chapter 2 TKINTER

Working with Tkinter Widgets

Tkinter is Python’s standard GUI (Graphical User Interface) package. Tkinter provides us with a variety of common GUI elements which we can use to build out interface – such as buttons, menus and various kind of entry fields and display areas. We call these elements Widgets.

In general, Widget is an element of Graphical User Interface (GUI) that displays/ illustrates information or gives a way for the user to interact with the OS. In Tkinter, Widgets are objects; instances of classes that represent buttons, frames, and so on.[8]

Example:

```
from tkinter import *
# create root window
root = Tk()
# frame inside root window
frame = Frame(root)
# geometry method
frame.pack()
# button inside frame which is
# inside root
button = Button(frame, text ='Geek')
button.pack()
# Tkinter event loop
root.mainloop()
```

Controlling layout with Geometry Managers

Tkinter possess three layout managers: Pack, Grid, Place

The three layout managers pack, grid, and place should never be mixed in the same master window! Geometry managers serve various functions. They:

1) arrange widgets on the screen
2) register widgets with the underlying windowing system
3) manage the display of widgets on the screen

Pack: Pack is the easiest to use of the three geometry managers of Tk and Tkinter. Instead of having to declare precisely where a widget should appear on the display screen, we can declare the positions of widgets with the pack command relative to each other. The pack command takes care of the details. Though the pack command is easier to use, this layout managers is limited in its possibilities compared to the grid and place mangers.

Place: The Place geometry manager allows you explicitly set the position and size of a window, either in absolute terms, or relative to another window. The place manager can be accessed through the place method. It can be applied to all standard widgets.

Grid: Grid was introduced in 1996 as an alternative to pack. Though grid is easier to learn and to use and produces nicer layouts, lots of developers keep using pack. The Grid geometry manager places the widgets in a 2-dimensional table, which consists of a number of rows and columns. The position of a widget is defined by a row and a column number.

Chapter 3 MACHINE LEARNING

Pandas Library

Pandas is an open-source Python package that is most widely used for data science/ data analysis and machine learning tasks. It is built on top of another package named NumPy, which provides support for multi-dimensional arrays.

It is used for data cleaning and analysis. It has features which are used for exploring, cleaning, transforming and visualizing from data.

# To install pandas in terminal or command line use one of the commands pip
install pandas

NumPy Library

NumPy stands for 'Numerical Python'. It is an open-source Python library used to perform various mathematical and scientific tasks. It contains multi-dimensional arrays and matrices, along with many high-level mathematical functions that operate on these arrays and matrices. You can install NumPy with:
pip install numpy.

After installing NumPy, you can now use this library by importing it. To import NumPy use: import numpy as np.

Missingno

Missingno is an excellent and simple to use Python library that provides a series of visualisations to understand the presence and distribution of missing data within a pandas dataframe. This can be in the form of either a barplot, matrix plot, heatmap, or a dendrogram.

# To install pandas in terminal or command line use one of the commands pip install missingno.

Matplotlib

Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc.

# To install pandas in terminal or command line use one of the commands pip install matplotlib.

Randonforest classifier

Random forests or random decision forests is an ensemble learning method for classification and regression and other tasks that operates by constructing a multitude of decision tree at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision tree but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

Chapter 6 CONCLUSIONS

Through this project I have build the machine learning project to predict the status of the order and sale of the different items by analysing the different graphs and on the bases of that can make business plan accordingly and avoid loss and earn more and more profit.

REFERENCES

- For different kind of query and doubt:

  https://www.quora.com/search?q=what%20is%20the%20dilevery%20model%20for%20online%20apps

- For study and research purpose:

  https://www.amazon.in/gp/help/customer/display.html?nodeId=GJK6YT2NCPCAV7DC

- For the dataset and project detail:

  https://www.kaggle.com/

- For the code and graph:

  https://www.kaggle.com/datasets/pranalibose/amazon-seller-order-status-prediction

PROJECT BACKGROUND

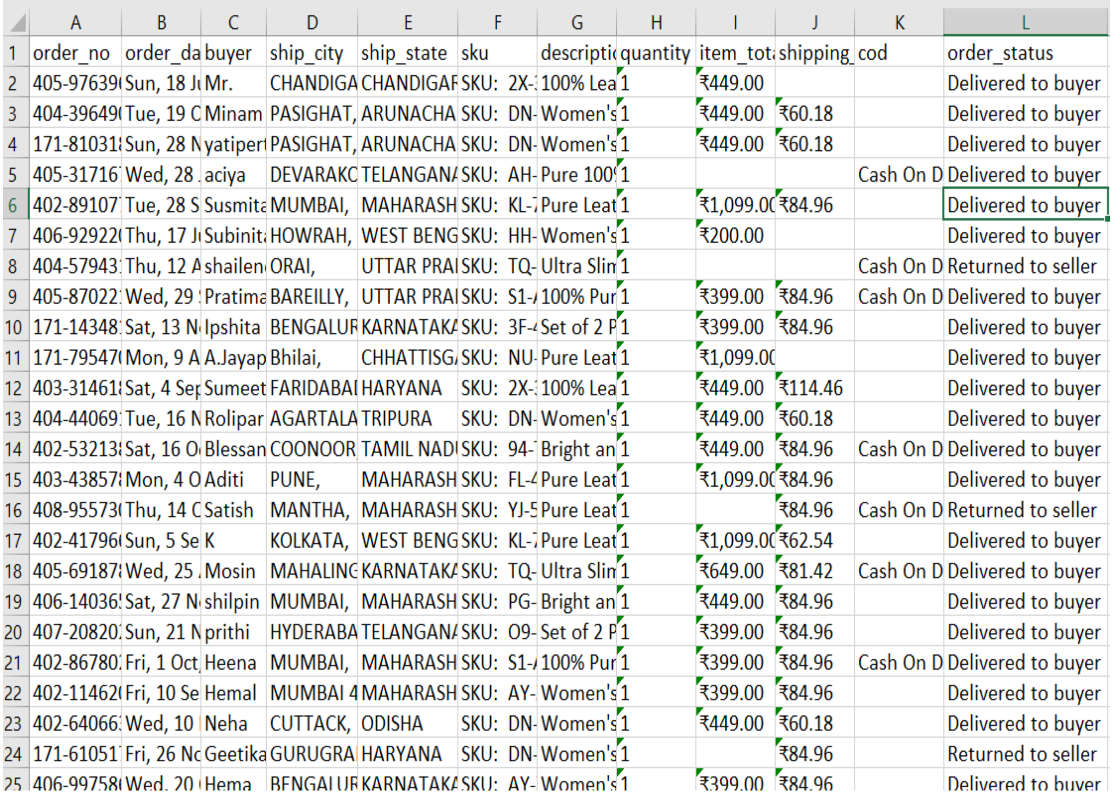
The aim of the project is to predict the status of order. We prepare a dataset by the keyword amazon seller (The dataset is obtained from the Kaggle https://www.kaggle.com/)

Which gives the list of the buyer and the seller with other features like order no., shipping no., features of the product etc.

The link of the site from where we can download the dataset:

https://www.kaggle.com/datasets/pranalibose/amazon-seller-order-status-prediction

For reference purpose the Image of the dataset:



	A	B	C	D	E	F	G	H	I	J	K	L
1	order_no	order_da	buyer	ship_city	ship_state	sku	descripti	quantity	item_tot	shipping_	cod	order_status
2	405-97639	Sun, 18 J	Mr.	CHANDIGA	CHANDIGAF	SKU: 2X-	100% Lea	1	₹449.00			Delivered to buyer
3	404-39649	Tue, 19 C	Minam	PASIGHAT,	ARUNACHA	SKU: DN-	Women's	1	₹449.00	₹60.18		Delivered to buyer
4	171-81031	Sun, 28 N	yatiperi	PASIGHAT,	ARUNACHA	SKU: DN-	Women's	1	₹449.00	₹60.18		Delivered to buyer
5	405-31716	Wed, 28 .	aciya	DEVARAKC	TELANGANA	SKU: AH-	Pure 100	1			Cash On D	Delivered to buyer
6	402-89107	Tue, 28 S	Susmita	MUMBAI,	MAHARASH	SKU: KL-7	Pure Leat	1	₹1,099.0	₹84.96		Delivered to buyer
7	406-92922	Thu, 17 J	Subinit	HOWRAH,	WEST BENG	SKU: HH-	Women's	1	₹200.00			Delivered to buyer
8	404-57943	Thu, 12 A	shailen	ORAI,	UTTAR PRA	SKU: TQ-	Ultra Slin	1			Cash On D	Returned to seller
9	405-87022	Wed, 29	Pratima	BAREILLY,	UTTAR PRA	SKU: S1-/	100% Pur	1	₹399.00	₹84.96	Cash On D	Delivered to buyer
10	171-14348	Sat, 13 N	Ipshita	BENGALUF	KARNATAKA	SKU: 3F-4	Set of 2 P	1	₹399.00	₹84.96		Delivered to buyer
11	171-79547	Mon, 9 A	A.Jayap	Bhilai,	CHHATTISGA	SKU: NU-	Pure Leat	1	₹1,099.0			Delivered to buyer
12	403-31461	Sat, 4 Sep	Sumeet	FARIDABA	HARYANA	SKU: 2X-	100% Lea	1	₹449.00	₹114.46		Delivered to buyer
13	404-44069	Tue, 16 N	Rolipar	AGARTALA	TRIPURA	SKU: DN-	Women's	1	₹449.00	₹60.18		Delivered to buyer
14	402-53213	Sat, 16 O	Blessan	COONOOR	TAMIL NAD	SKU: 94-	Bright an	1	₹449.00	₹84.96	Cash On D	Delivered to buyer
15	403-43857	Mon, 4 O	Aditi	PUNE,	MAHARASH	SKU: FL-4	Pure Leat	1	₹1,099.0	₹84.96		Delivered to buyer
16	408-95573	Thu, 14 C	Satish	MANTHA,	MAHARASH	SKU: YJ-5	Pure Leat	1		₹84.96	Cash On D	Returned to seller
17	402-41796	Sun, 5 Se	K	KOLKATA,	WEST BENG	SKU: KL-7	Pure Leat	1	₹1,099.0	₹62.54		Delivered to buyer
18	405-69187	Wed, 25 .	Mosin	MAHALING	KARNATAKA	SKU: TQ-	Ultra Slin	1	₹649.00	₹81.42	Cash On D	Delivered to buyer
19	406-14036	Sat, 27 N	shilpin	MUMBAI,	MAHARASH	SKU: PG-	Bright an	1	₹449.00	₹84.96		Delivered to buyer
20	407-20820	Sun, 21 N	prithi	HYDERABA	TELANGANA	SKU: O9-	Set of 2 P	1	₹399.00	₹84.96		Delivered to buyer
21	402-86780	Fri, 1 Oct,	Heena	MUMBAI,	MAHARASH	SKU: S1-/	100% Pur	1	₹399.00	₹84.96	Cash On D	Delivered to buyer
22	402-11462	Fri, 10 Se	Hemal	MUMBAI 4	MAHARASH	SKU: AY-	Women's	1	₹399.00	₹84.96		Delivered to buyer
23	402-64066	Wed, 10	Neha	CUTTACK,	ODISHA	SKU: DN-	Women's	1	₹449.00	₹60.18		Delivered to buyer
24	171-61051	Fri, 26 Nc	Geetika	GURUGRA	HARYANA	SKU: DN-	Women's	1		₹84.96		Returned to seller
25	406-99758	Wed, 20	Hema	BENGALUF	KARNATAKA	SKU: AY-	Women's	1	₹399.00	₹84.96		Delivered to buyer

IMPORT PACKAGE

- import numpy as np   #linear algebra
- import pandas as pd   #data processing
- import seaborn as sns
- import matplot.pyplot as plt

IMPORT DATA

- data = pd.read_excel(‘order_status.xlsx’)

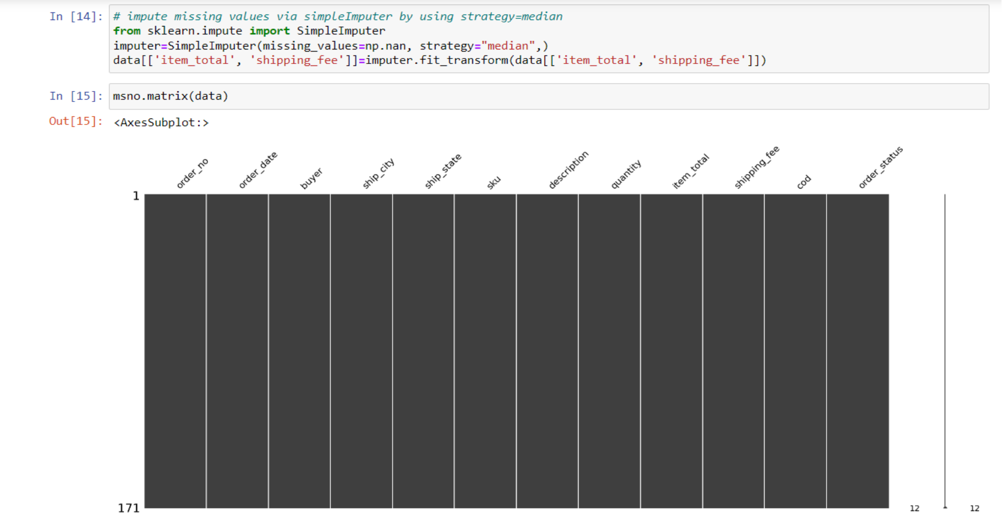
UNDERSTANDING THE DATA

- There are 171 datapoints and 12 features
- order_no and date are unique
- One buyer made at most 3 orders
- City,state, sku and description need further analysis
- The ordered quantity is at most 4 but median is 1

- Item_total and shipping_fee should be converted to float
- Cash on delivery has only 1 value, so it seems like boolean
- Order status is target variable, and has 2 classes.
- item_total, shipping_fee and cod has missing values, we should deal with them
- Order_data should be date dtype
- Item_total and Shipping_fee should be float or int
- Cod should be boolean
- Order_status is also should be Boolean

Data Cleaning

Checking for null values, missing values and check for the column which is better to take out of the model, what are the redundant variables.

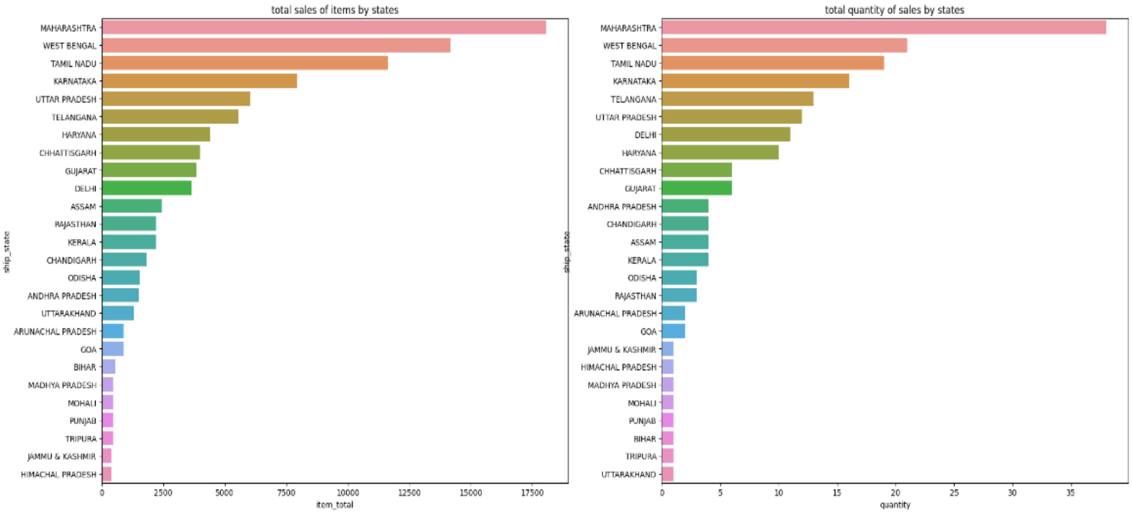


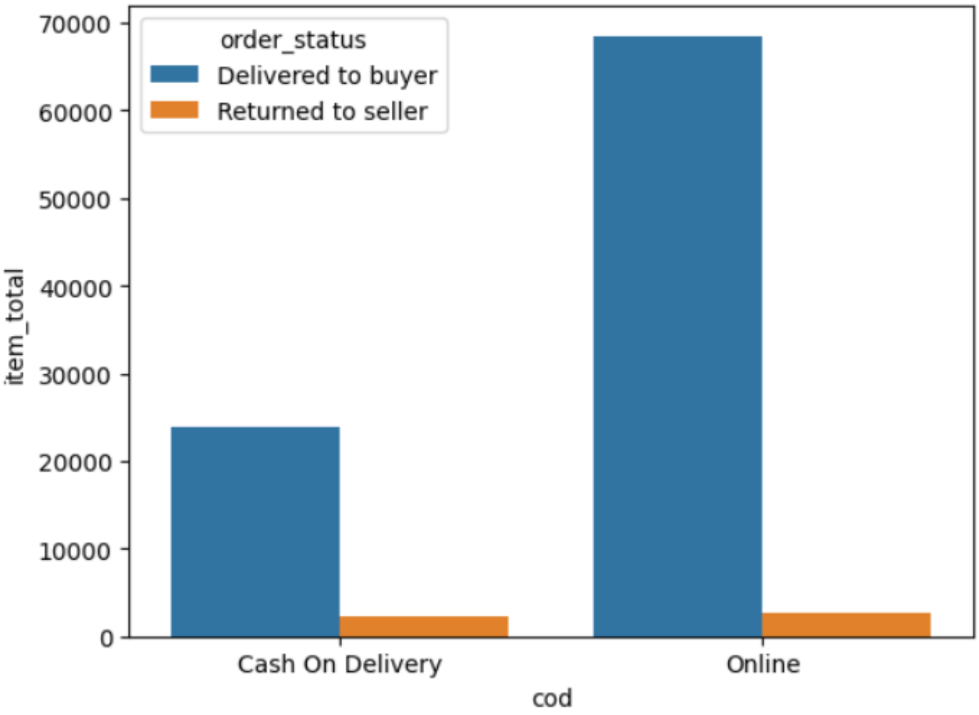
```
In [14]: # impute missing values via simpleImputer by using strategy=median
         from sklearn.impute import SimpleImputer
         imputer=SimpleImputer(missing_values=np.nan, strategy="median",)
         data[['item_total', 'shipping_fee']]=imputer.fit_transform(data[['item_total', 'shipping_fee']])

In [15]: msno.matrix(data)

Out[15]: <AxesSubplot:>
```

Visualization

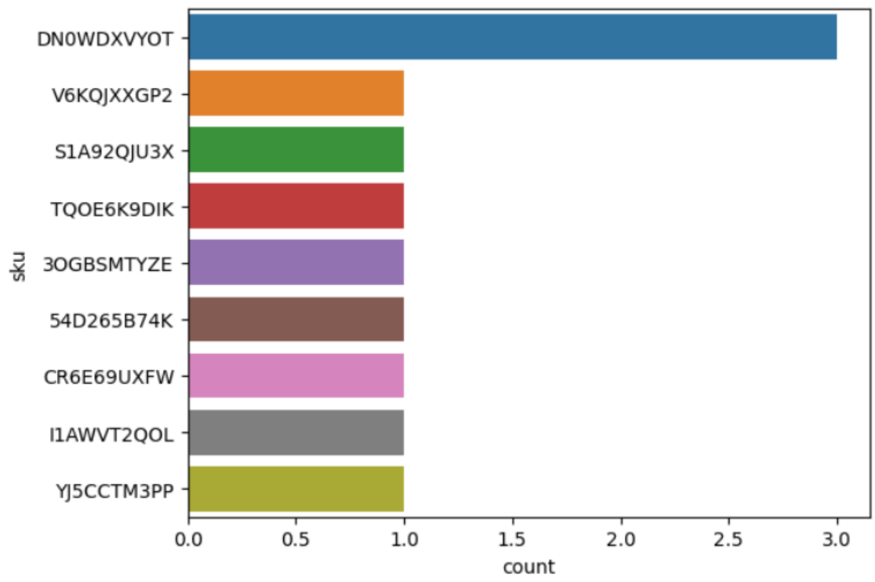
Visualization of data through graphs and plots

```
<AxesSubplot:title={'center':'total quantity of sales by states'}, xlabel='quantity', ylabel='ship_state'>
```



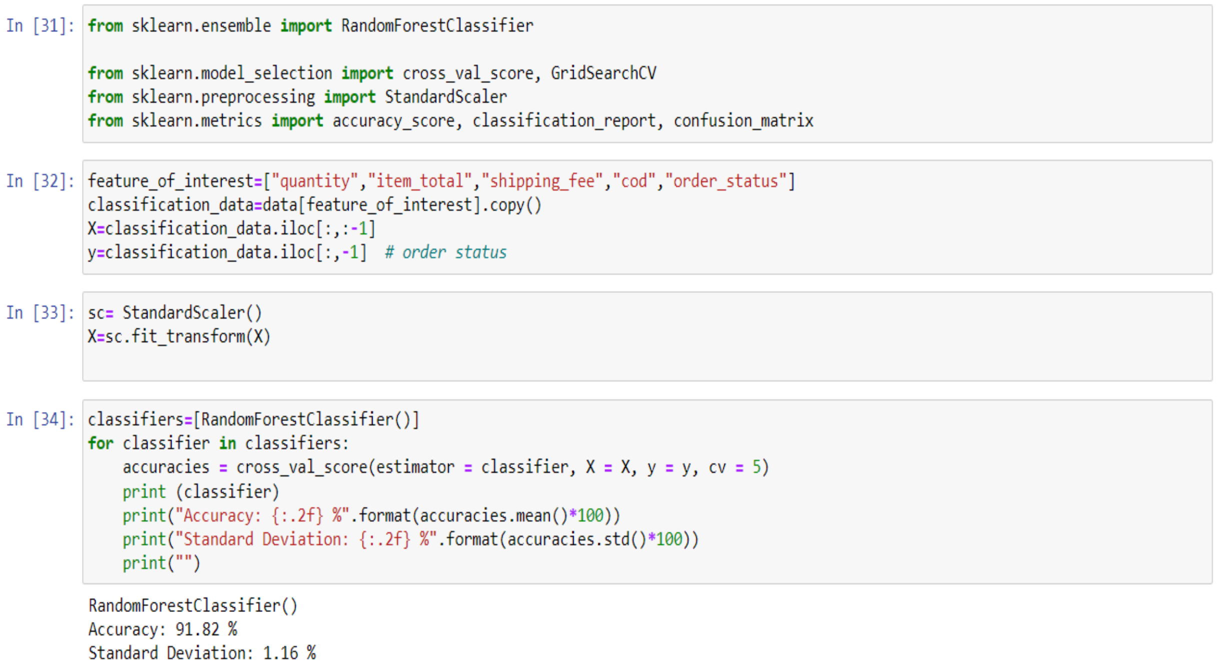


```
<AxesSubplot:xlabel='count', ylabel='sku'>
```



Modelling

Selecting predictive model



```
In [31]: from sklearn.ensemble import RandomForestClassifier

         from sklearn.model_selection import cross_val_score, GridSearchCV
         from sklearn.preprocessing import StandardScaler
         from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
In [32]: feature_of_interest=["quantity","item_total","shipping_fee","cod","order_status"]
         classification_data=data[feature_of_interest].copy()
         X=classification_data.iloc[:,:-1]
         y=classification_data.iloc[:,-1]  # order status
```

```
In [33]: sc= StandardScaler()
         X=sc.fit_transform(X)
```

```
In [34]: classifiers=[RandomForestClassifier()]
         for classifier in classifiers:
             accuracies = cross_val_score(estimator = classifier, X = X, y = y, cv = 5)
             print (classifier)
             print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
             print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
             print("")

         RandomForestClassifier()
         Accuracy: 91.82 %
         Standard Deviation: 1.16 %
```