

# **Bachelor Of Technology**

## **Computer Science and Engineering (Data Science)**

311 Program in collaboration with Virginia Tech. USA

### **Mini-Project Presentation**



**SVKM's NMIMS**

**Mukesh Patel School of Technology Management and  
Engineering, JVPD, Vile Parle, West**

**Mumbai- 400056**

**A PRESENTATION ON**

**“SALES FORECASTING FOR  
WOWMART”**

By  
**NIHARIKA AMRITKAR  
SHREYAS DESAI  
KAUNCHI JAIN**

**Faculty Mentors:**  
Prof. Ami Munshi, Prof. Mahesh Mali and Prof. Khinal Parmar

# Contents

- Introduction
- Literature Survey
- Problem Statement
- System Description
- Implementation
- Result and Analysis
- Conclusion
- Future Scope
- References

# Introduction

For our project, we are taking help of WoW-Mart, a fictional company and using it's sales related dataset to carry out the operations. Our project aims at predicting sales for the store's outlets in various locations and regions and helping the management staff to take decisions like whether a particular region, or outlet in general is proving profitable for them or not and whether they should continue operating there.

Our Experiment will also help WoW-Mart to help them in planning their supplies, incase a specific region or store is expecting heavy demand, then more supplies can be directed towards it, and if some store is expecting fall in demand, then it can either be shut or less resources can be directed towards it.

# Literature Survey

We surveyed 3 research papers on Moving Average:

- 1) Ms. Savita Satav, Dr. Netra Apte: “The Moving Average Crossover Strategy: A Study”, JMME, Volume 10, No. 03, July 2020, pp- 141- 146
- 2) Puchong Praekhaow: “Determination of Trading Points using Moving Average Methods”, June 2010.
- 3) Seng Hansun: “A Novel Research of New Moving Average Method in Time Series Anaysis”, August 2014

# Problem Statement

To forecast sales with the help of Moving Averages on the basis of Stores, Locations and Regions for comparative analysis.

To identify crossovers and MAD for prediction of trend and volatility in sales for the next quarter.

# System Description

## Hardware requirements:

- Functional Computer with sufficient RAM

## Software requirements:

- Jupyter Notebook
- SAS
- Excel
- Microsoft Presentation, Microsoft Word

# Algorithm & Flowchart

- Obtaining dataset through statistic means and organizing it
- Cleaning irrelevant data and managing null values
- Specifying the purpose of the analysis and opting for suitable method to analyze the dataset
- Working through the programming algorithm to develop the fundamentals
- Implementing the analysis technique using python
- Importing dataset in the python notebook and working with the fundamentals developed
- Making program flexible to different types of statistical data
- Running the code and checking if the code works for all types of Inputs
- Error management
- Concluding the finding of the program
- Analyzing the findings of the program
- Commenting on the analysis with regard to sales and predicting the trend with the help of MAD, Crossovers, Variance and Standard Deviation



# Our Dataset

HomeInsertDrawPage LayoutFormulasDataReviewViewDeveloper

Paste

Calibri (Body)11A A

=

=

=

General

Conditional FormattingFormat as TableCell Styles

Insert

Delete

Format

Σ

Sort & Filter

Find & Select

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	ID	Store_id	Store_Type	Location	Region	Date	Holiday	Discount	#Order	Sales					
2	T100000	1.0	S1	L3	R1	01/01/18	1	Yes	9	7011.84					
3	T100000	253.0	S4	L2	R1	01/01/18	1	Yes	60	51789.12					
4	T100000	252.0	S3	L2	R1	01/01/18	1	Yes	42	36868.20					
5	T100000	251.0	S2	L3	R1	01/01/18	1	Yes	23	19715.16					
6	T100000	250.0	S2	L3	R4	01/01/18	1	Yes	62	45614.52					
7	T100000	249.0	S1	L3	R2	01/01/18	1	Yes	39	34211.22					
8	T100000	248.0	S1	L1	R2	01/01/18	1	Yes	40	35352.66					
9	T100000	247.0	S1	L1	R3	01/01/18	1	Yes	64	52650.00					
10	T100000	246.0	S3	L1	R3	01/01/18	1	Yes	62	42633.78					
11	T100000	254.0	S4	L1	R1	01/01/18	1	Yes	87	62572.80					
12	T100000	245.0	S4	L1	R2	01/01/18	1	Yes	36	27468.21					
13	T100000	11.0	S4	L2	R1	01/01/18	1	Yes	69	57590.40					
14	T100000	243.0	S4	L2	R2	01/01/18	1	Yes	69	45563.25					
15	T100000	242.0	S4	L1	R1	01/01/18	1	Yes	51	47322.90					
16	T100000	241.0	S1	L1	R4	01/01/18	1	Yes	56	42889.89					
17	T100000	240.0	S3	L2	R3	01/01/18	1	Yes	59	44319.33					
18	T100000	239.0	S3	L1	R3	01/01/18	1	Yes	35	29781.84					
19	T100000	238.0	S1	L1	R2	01/01/18	1	Yes	29	26906.61					
20	T100000	237.0	S2	L3	R3	01/01/18	1	Yes	55	44858.67					
21	T100000	244.0	S2	L5	R4	01/01/18	1	Yes	46	39193.20					
22	T100000	236.0	S1	L1	R3	01/01/18	1	Yes	36	32486.76					
23	T100000	255.0	S3	L2	R4	01/01/18	1	Yes	63	47317.14					
24	T100000	256.0	S1	L2	R4	01/01/18	1	Yes	78	69937.98					
25	T100000	273.0	S2	L3	R4	01/01/18	1	Yes	42	31509.72					
26	T100000	272.0	S2	L4	R2	01/01/18	1	Yes	48	33665.10					

TRAIN

+

ReadyAccessibility: Investigate

143%

# Data Cleaning

- Discounts, Order Id, Holidays and Order number was removed using python.
- A new variable was created and the useful information was stored in it.
- The variable was later converted to .csv file, which is the data we worked on.

# Project Outlook

- Let's look at the sales of store type S1 and predict how will it perform in the next quarter.

```
1 import os
2 import webbrowser
3 choice = input("Choose the category for sales forecasting: \n1. Location \n2. Store \n3. Region \n")
4 location=['L1','L2','L3','L4','L5']
5 region=['R1','R2','R3']
6 store=['S1','S2','S3','S4']
7 locdict = {"L1": [18242.473622960486, 13520.53723990161, 182804927.25556624],
8             "L2": [15433.59878593115, 20642.76291054448, 426123660.58095074],
9             "L3": [8158.019792302034, 10720.330010626263, 114925475.53673409],
10            "L4": [7016.536928487199, 9176.00817467835, 84199126.0217639],
11            "L5": [6916.977443440596, 9989.776948963, 80816085.20309618]}
12 storedict = {"S4": [15485.596804601007, 20750.802114082184, 430562587.77176505],
13              "S3": [11235.522173634065, 14907.164575749493, 222223555.68848056],
14              "S1": [9293.13329077292, 12303.08177750587, 151365821.22399607],
15              "S2": [6993.575434728554, 9168.68074146217, 84064706.53805931]}
16 regdict = {"R1": [16174.836187763278, 21285.929101593338, 453090777.7180582],
17            "R2": [12389.862692369281, 16468.46754720104, 271210423.35321385],
18            "R3": [12617.572095062265, 16615.543092039592, 276076272.2434246],
19            "R4": [11999.81163508174, 15930.185662734784, 253770815.24920088]}
20
21 def show_graph(ch):
22     tempch+= "Rolling.png"
23     temp2ch+= "Expo.png"
24     os.startfile(temp)
25     os.startfile(temp2)
26     webbrowser.open('Gyaan.html')
27 if choice == "Location":
28     print("Choose from",location)
29     subchoice = input()
30     print("MAD:", locdict[subchoice][0], "\nVariance: ", locdict[subchoice][1],
31           "\nStandard Deviation",locdict[subchoice][2])
32     show_graph(subchoice)
33 elif choice == "Store":
34     print("Choose from",store)
35     subchoice = input()
36     print("MAD:", storedict[subchoice][0], "\nVariance: ", storedict[subchoice][1],
37           "\nStandard Deviation",storedict[subchoice][2])
38     show_graph(subchoice)
39 elif choice == "Region":
40     print("Choose from",region)
```

Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]  
Type "copyright", "credits" or "license()" for more information.  
IPython 7.29.0 -- An enhanced Interactive Python.  
In [1]: runfile('C:/Users/Shreyas/OneDrive/Desktop/Project\_Backend/IPP\_Final.py', wdir='C:/Users/Shreyas/OneDrive/Desktop/Project\_Backend')  
Choose the category for sales forecasting:  
1. Location  
2. Store  
3. Region  
Store  
Choose from ['S1', 'S2', 'S3', 'S4']



The image shows the Spyder Python IDE interface. The main editor displays a Python script named `IPP_Final.py` that implements a sales forecasting tool. The script uses a web browser to present a menu to the user. The menu options are: 1. Location, 2. Store, 3. Region. The user has selected 'Store', and the script has displayed the following data for Store S1:

- Store: S1
- MAD: 9293.13329077292
- Variance: 12303.08177750587
- Standard Deviation: 151365821.22399697

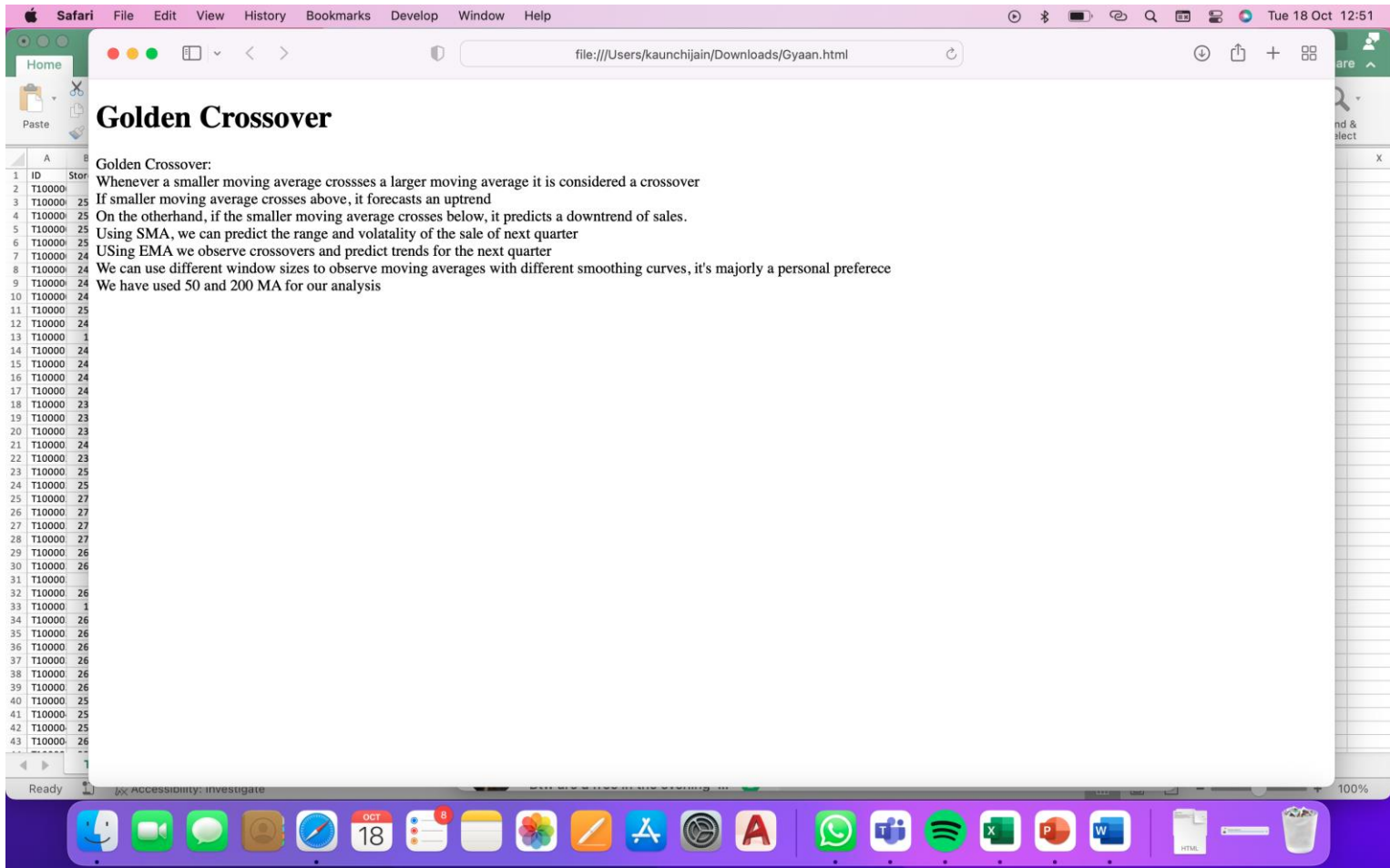
The console window on the right shows the execution of the script, including the prompt to choose a category and the resulting output for Store S1. The status bar at the bottom indicates the current file is `IPP_Final.py`, line 18, column 79.

We chose Store from menu and then chose S1 for the results

# Results:

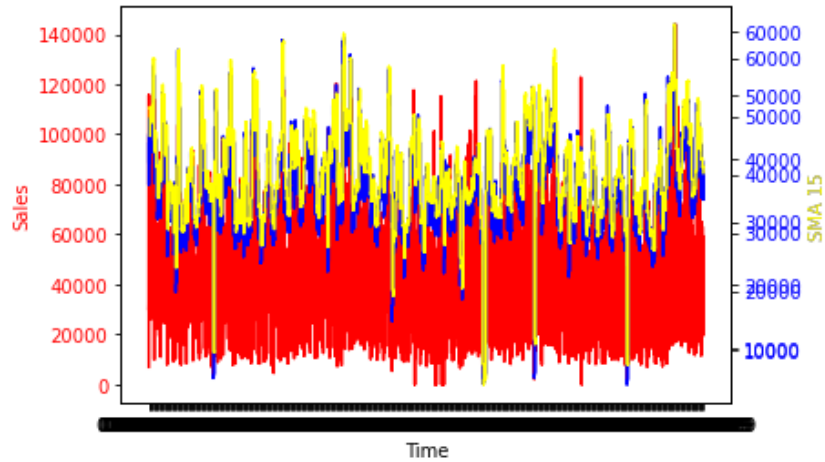
- MAD is derived
- Standard Deviation is calculated
- Variance is calculated
- 2 graphs are generated:
  1. Simple Moving Average with 1 window size
  2. Exponential Moving Average with 2 window sizes
- Manual to analyze the results is popped up

# Manual using HTML

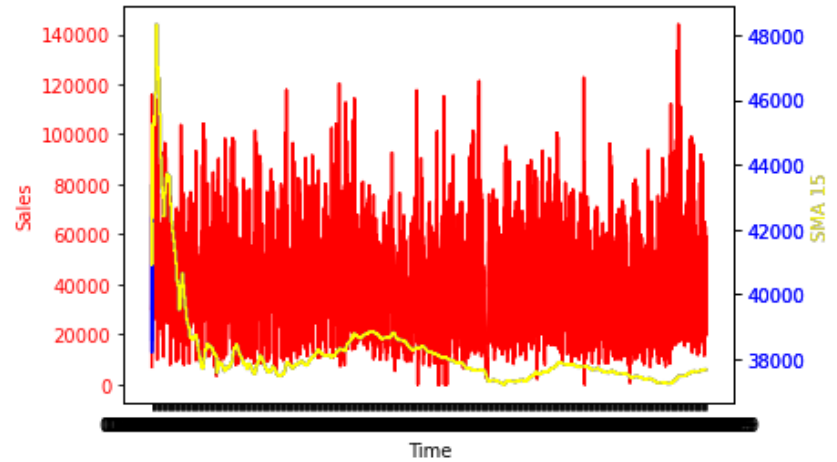


# Graphs

- SMA



- EMA



# Concepts Used: IBSAM Statistics

- Moving Average
  1. Simple Moving Average
  2. Exponential Moving Average
- Standard Deviation
- Variance
- Mean Absolute Deviation
- Crossovers in Moving Averages of 2 window sizes
- Forecasting: Uptrend & Downtrends



# Implementation: Driver code

```
choicedf = region_list.rolling(50)
choicedf1 = region_list.rolling(200)
```

```
moving_average=choicedf.mean()
moving_average_list = moving_average.tolist()
moving_average_list = np.array(moving_average_list)
```

```
moving_average1=choicedf1.mean()
moving_average_list1 = moving_average1.tolist()
moving_average_list1 = np.array(moving_average_list1)
```

Calculating values for SMA

- `fig, ax1 = plt.subplots()`
- 
- `ax1.set_xlabel('Time')`
- `ax1.set_ylabel('Sales', color = 'red')`
- `ax1.plot(specific_region['Date'], specific_region['Sales'], color = 'red')`
- `ax1.tick_params(axis = 'y', labelcolor = 'red')`
- 
- `ax2 = ax1.twinx()`
- 
- `ax2.set_ylabel('SMA', color = 'blue')`
- `ax2.plot(specific_region['Date'], moving_average_list, color = 'blue')`
- `ax2.tick_params(axis = 'y', labelcolor = 'blue')`
- 
- `ax3 = ax1.twinx()`
- 
- `ax3.set_ylabel('SMA', color = 'yellow')`
- `ax3.plot(specific_region['Date'], moving_average_list1, color = 'yellow')`
- `ax3.tick_params(axis = 'y', labelcolor = 'blue')`
- 
- `plt.show()`

Plotting 2 values on y-axis and one value on x axis i.e. dates

# Main code

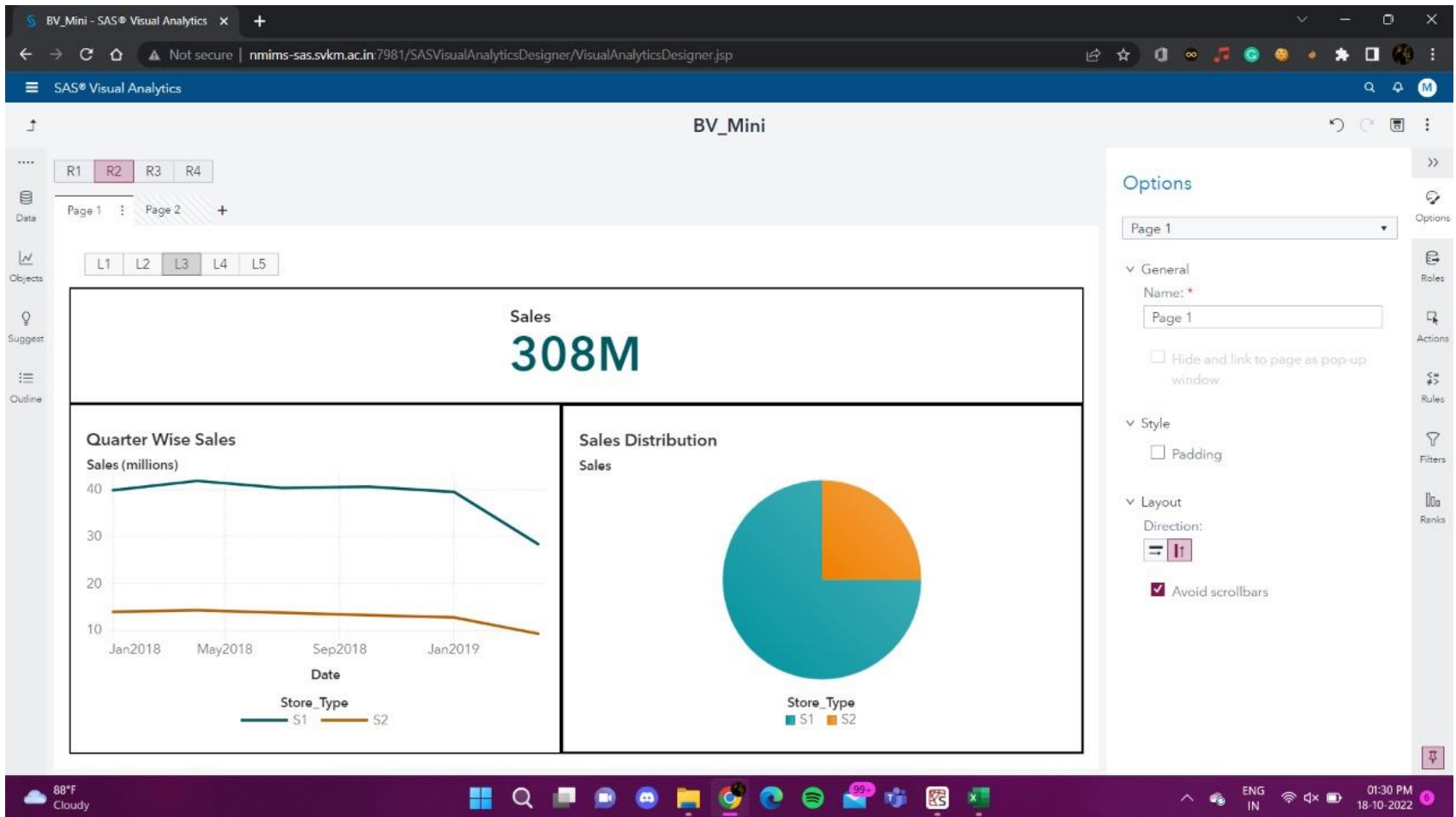
```
def show_graph(ch):
    temp=ch+"_Rolling.png"
    temp2=ch+"_Expo.png"
    os.startfile(temp)
    os.startfile(temp2)
    webbrowser.open('Gyaan.html')
if choice == "Location":
    print("Choose from",location)
    subchoice = input()
    print("MAD:", locdict[subchoice][0], "\nVariance: ", locdict[subchoice][1],
          "\nStandard Deviation",locdict[subchoice][2])
    show_graph(subchoice)
elif choice == "Store":
    print("Choose from",store)
    subchoice = input()
    print("MAD:", storedict[subchoice][0], "\nVariance: ", storedict[subchoice][1],
          "\nStandard Deviation",storedict[subchoice][2])
    show_graph(subchoice)
elif choice == "Region":
    print("Choose from",region)
    subchoice = input()
    print("MAD:", regdict[subchoice][0], "\nVariance: ", regdict[subchoice][1],
          "\nStandard Deviation",regdict[subchoice][2])
    show_graph(subchoice)
else:
    print("Invalid Choice")
```

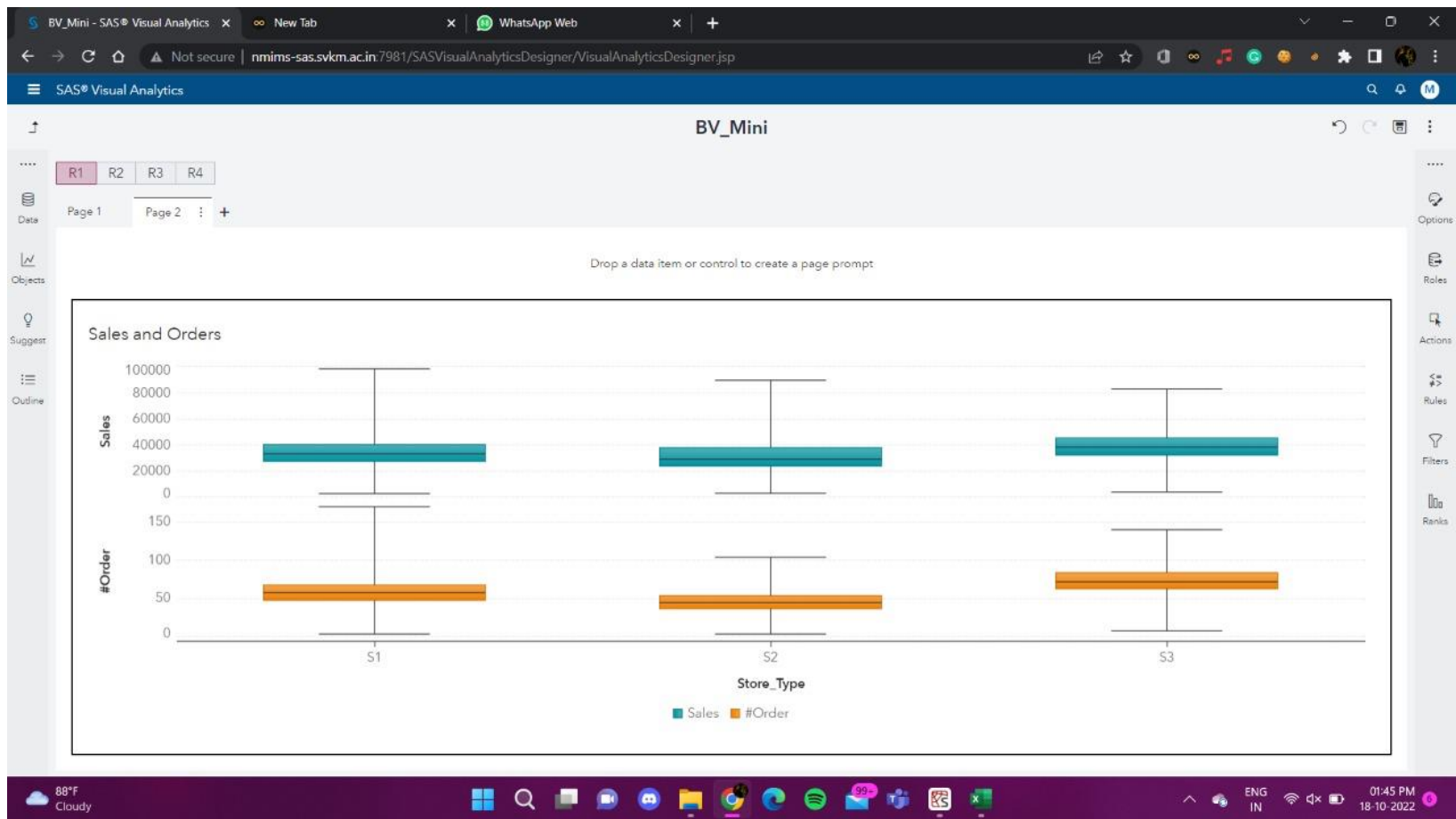
Taking input for required analysis and using code flexibility to predict sales for the same

# Concepts used: Python

- Numpy
- Functions
- Pandas
- Matplotlib
- Lambda functions
- Data Structures

# Visualization of Data





Directed Page Link

# Concepts Used: Business Visualization

- Action Prompts
- Pie Chart
- Line Chart
- Box Plot
- Information Graphics
- Hidden Pages and page links

# Result and Analysis

- Whenever a smaller moving average crosses a larger moving average, it is considered a crossover.
- If smaller moving average crosses above, it forecasts an uptrend.
- On the other hand, if the smaller moving average crosses below, it predicts a downtrend of sales.
- Using SMA, we can predict the range and volatility of the sale of next quarter.
- Using EMA we observe crossovers and predict trends for the next quarter.

We can use different window sizes to observe moving averages with different smoothing curves, it's majorly a personal preference.

- We have used 50 and 200 MA for our analysis.
- If MAD is larger, the sales are more volatile.
- We can use variances and standard deviation to predict sales for the next term.



# References

<file:///Users/kaunchijain/Downloads/Gyaan.html>

<https://www.geeksforgeeks.org/python-pandas-dataframe-rolling/>

<https://www.geeksforgeeks.org/append-extend-python/>

[https://www.researchgate.net/publication/233988919\\_Determination of Trading Points using the Moving Average Methods](https://www.researchgate.net/publication/233988919_Determination_of_Trading_Points_using_the_Moving_Average_Methods)

[https://www.researchgate.net/publication/301547550\\_A Novel Research of New Moving Average Method in Time Series Analysis](https://www.researchgate.net/publication/301547550_A_Novel_Research_of_New_Moving_Average_Method_in_Time_Series_Analysis)

<https://core.ac.uk/download/pdf/153776849.pdf>

# THANK YOU