

## Practical - 1

### Aim: Introduction to Excel

- Perform conditional formatting on a dataset using various criteria.
- Create a pivot table to analyze and summarize data.
- Use the VLOOKUP function to retrieve information from a different worksheet or table.
- Perform what-if analysis using Goal Seek to determine input values for desired output.

### Requirements : Platform : MS Excel

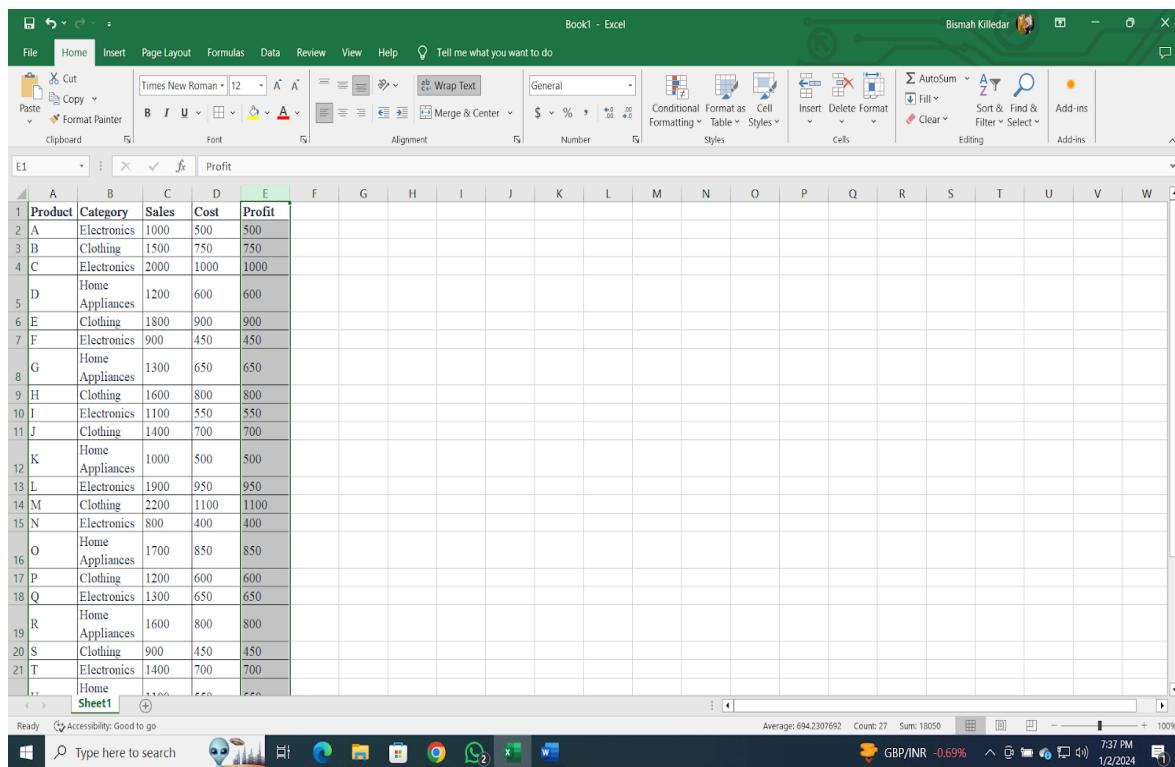
Dataset: Pivot Table.xlsx, Use conditional formatting.xlsx, VLOOKUP.xlsx

- Perform conditional formatting on a dataset using various criteria.

We perform conditional formatting on the "Profit" column to highlight cells with a profit greater than 800 using following steps:

#### Steps:

1. Select the "Profit" column (Column E).



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Product	Category	Sales	Cost	Profit																		
2	A	Electronics	1000	500	500																		
3	B	Clothing	1500	750	750																		
4	C	Electronics	2000	1000	1000																		
5	D	Home Appliances	1200	600	600																		
6	E	Clothing	1800	900	900																		
7	F	Electronics	900	450	450																		
8	G	Home Appliances	1300	650	650																		
9	H	Clothing	1600	800	800																		
10	I	Electronics	1100	550	550																		
11	J	Clothing	1400	700	700																		
12	K	Home Appliances	1000	500	500																		
13	L	Electronics	1900	950	950																		
14	M	Clothing	2200	1100	1100																		
15	N	Electronics	800	400	400																		
16	O	Home Appliances	1700	850	850																		
17	P	Clothing	1200	600	600																		
18	Q	Electronics	1300	650	650																		
19	R	Home Appliances	1600	800	800																		
20	S	Clothing	900	450	450																		
21	T	Electronics	1400	700	700																		
		Home	1100	550	550																		

2. Go to the "Home" tab on the ribbon.
3. Click on "Conditional Formatting" in the toolbar.

A screenshot of Microsoft Excel showing a data table in the 'Sheet1' tab. The table has columns for Product, Category, Sales, Cost, and Profit. The 'Conditional Formatting' ribbon tab is selected, and the 'Highlight Cells Rules' dropdown is open, showing options like 'Top/Bottom Rules', 'Data Bars', 'Color Scales', 'Icon Sets', 'New Rule...', 'Clear Rules', and 'Manage Rules...'.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Product	Category	Sales	Cost	Profit							
2	A	Electronics	1000	500	500							
3	B	Clothing	1500	750	750							
4	C	Electronics	2000	1000	1000							
5	D	Home Appliances	1200	600	600							
6	E	Clothing	1800	900	900							
7	F	Electronics	900	450	450							
8	G	Home Appliances	1300	650	650							
9	H	Clothing	1600	800	800							
10	I	Electronics	1100	550	550							
11	J	Clothing	1400	700	700							
12	K	Home Appliances	1000	500	500							
13	L	Electronics	1900	950	950							
14	M	Clothing	2200	1100	1100							
15	N	Electronics	800	400	400							
16	O	Home Appliances	1700	850	850							
17	P	Clothing	1200	600	600							
18	Q	Electronics	1300	650	650							
19	R	Home Appliances	1600	800	800							
20	S	Clothing	900	450	450							
21	T	Electronics	1400	700	700							
		Home Appliances	1100	550	550							

#### 4. Choose "Highlight Cells Rules" and then "Greater Than."

A screenshot of Microsoft Excel showing the same data table as the previous image. The 'Conditional Formatting' ribbon tab is selected, and the 'Highlight Cells Rules' dropdown is open, with 'Greater Than...' highlighted. Other options visible include 'Less Than...', 'Between...', 'Equal To...', 'Text that Contains...', 'A Date Occurring...', 'Duplicate Values...', and 'More Rules...'.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Product	Category	Sales	Cost	Profit							
2	A	Electronics	1000	500	500							
3	B	Clothing	1500	750	750							
4	C	Electronics	2000	1000	1000							
5	D	Home Appliances	1200	600	600							
6	E	Clothing	1800	900	900							
7	F	Electronics	900	450	450							
8	G	Home Appliances	1300	650	650							
9	H	Clothing	1600	800	800							
10	I	Electronics	1100	550	550							
11	J	Clothing	1400	700	700							
12	K	Home Appliances	1000	500	500							
13	L	Electronics	1900	950	950							
14	M	Clothing	2200	1100	1100							
15	N	Electronics	800	400	400							
16	O	Home Appliances	1700	850	850							
17	P	Clothing	1200	600	600							
18	Q	Electronics	1300	650	650							
19	R	Home Appliances	1600	800	800							
20	S	Clothing	900	450	450							
21	T	Electronics	1400	700	700							
		Home Appliances	1100	550	550							

## 5. Enter the threshold value as 800.

The screenshot shows a Microsoft Excel spreadsheet titled "Book1 - Excel". The data is organized into columns A through W, with rows numbered from 1 to 21. Column E is labeled "Profit". The "Greater Than" dialog box is open, displaying the condition "800" and the format "Light Red Fill with Dark Red Text". The status bar at the bottom indicates "Average: 694.2307692 Count: 27 Sum: 18050".

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Product	Category	Sales	Cost	Profit																		
2	A	Electronics	1000	500	500																		
3	B	Clothing	1500	750	750																		
4	C	Electronics	2000	1000	1000																		
5	D	Home Appliances	1200	600	600																		
6	E	Clothing	1800	900	900																		
7	F	Electronics	900	450	450																		
8	G	Home Appliances	1300	650	650																		
9	H	Clothing	1600	800	800																		
10	I	Electronics	1100	550	550																		
11	J	Clothing	1400	700	700																		
12	K	Home Appliances	1000	500	500																		
13	L	Electronics	1900	950	950																		
14	M	Clothing	2200	1100	1100																		
15	N	Electronics	800	400	400																		
16	O	Home Appliances	1700	850	850																		
17	P	Clothing	1200	600	600																		
18	Q	Electronics	1300	650	650																		
19	R	Home Appliances	1600	800	800																		
20	S	Clothing	900	450	450																		
21	T	Electronics	1400	700	700																		
	U	Home Appliances	1300	650	650																		

## 6. Customize the formatting options (e.g., choose a fill color).

7. Click "OK" to apply the rule.

The screenshot shows the same Microsoft Excel spreadsheet as the previous one, but now the cells containing values greater than 800 are filled with a light red color and have dark red text. The "Greater Than" dialog box is no longer visible.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Product	Category	Sales	Cost	Profit																		
2	A	Electronics	1000	500	500																		
3	B	Clothing	1500	750	750																		
4	C	Electronics	2000	1000	1000																		
5	D	Home Appliances	1200	600	600																		
6	E	Clothing	1800	900	900																		
7	F	Electronics	900	450	450																		
8	G	Home Appliances	1300	650	650																		
9	H	Clothing	1600	800	800																		
10	I	Electronics	1100	550	550																		
11	J	Clothing	1400	700	700																		
12	K	Home Appliances	1000	500	500																		
13	L	Electronics	1900	950	950																		
14	M	Clothing	2200	1100	1100																		
15	N	Electronics	800	400	400																		
16	O	Home Appliances	1700	850	850																		
17	P	Clothing	1200	600	600																		
18	Q	Electronics	1300	650	650																		
19	R	Home Appliances	1600	800	800																		
20	S	Clothing	900	450	450																		
21	T	Electronics	1400	700	700																		
	U	Home Appliances	1300	650	650																		

- **Create a pivot table to analyze and summarize data.**

Following are the steps to create a pivot table to analyze total sales by category.

### Steps:

1. Select the entire dataset including headers.

The screenshot shows a Microsoft Excel window titled "Book1 - Excel". The ribbon is visible at the top with tabs like File, Home, Insert, Page Layout, Formulas, Data, Review, View, and Help. The "Home" tab is selected. A table is displayed in the worksheet, starting at cell A1. The table has columns labeled Product, Category, Sales, Cost, and Profit. The "Profit" column for row 4 (Cell E4) is highlighted in red. The table contains 21 rows of data, with rows 1 through 4 being headers and rows 5 through 21 being data points. The "Sheet1" tab is selected at the bottom left. The status bar at the bottom right shows "Average: 925.6410256 Count: 135 Sum: 72200".

2. Go to the "Insert" tab on the ribbon.

3. Click on "PivotTable."

The screenshot shows the same Microsoft Excel window as before, but now the "Insert" tab is selected in the ribbon. In the "Tables" section of the ribbon, the "PivotTables" button is highlighted. The main area of the screen shows the same dataset as the previous screenshot. The "Sheet1" tab is still selected at the bottom left. The status bar at the bottom right shows "Average: 925.6410256 Count: 135 Sum: 72200".

#### 4. Choose where you want to place the PivotTable (e.g., new worksheet).

A screenshot of Microsoft Excel showing a PivotTable named "PivotTable1" in the range A1:W21. The PivotTable has columns for Product, Category, Sales, Cost, and Profit. The Profit column is highlighted in red. The PivotTable ribbon tab is selected.

#### 5. Drag "Category" to the Rows area.

#### 6. Drag "Sales" to the Values area, choosing the sum function.

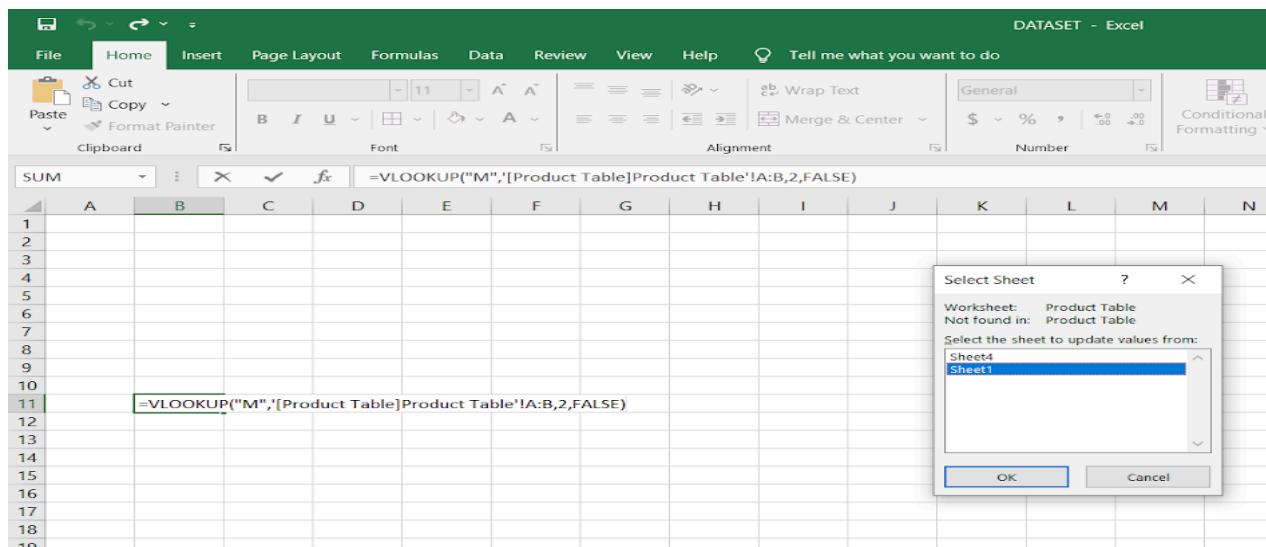
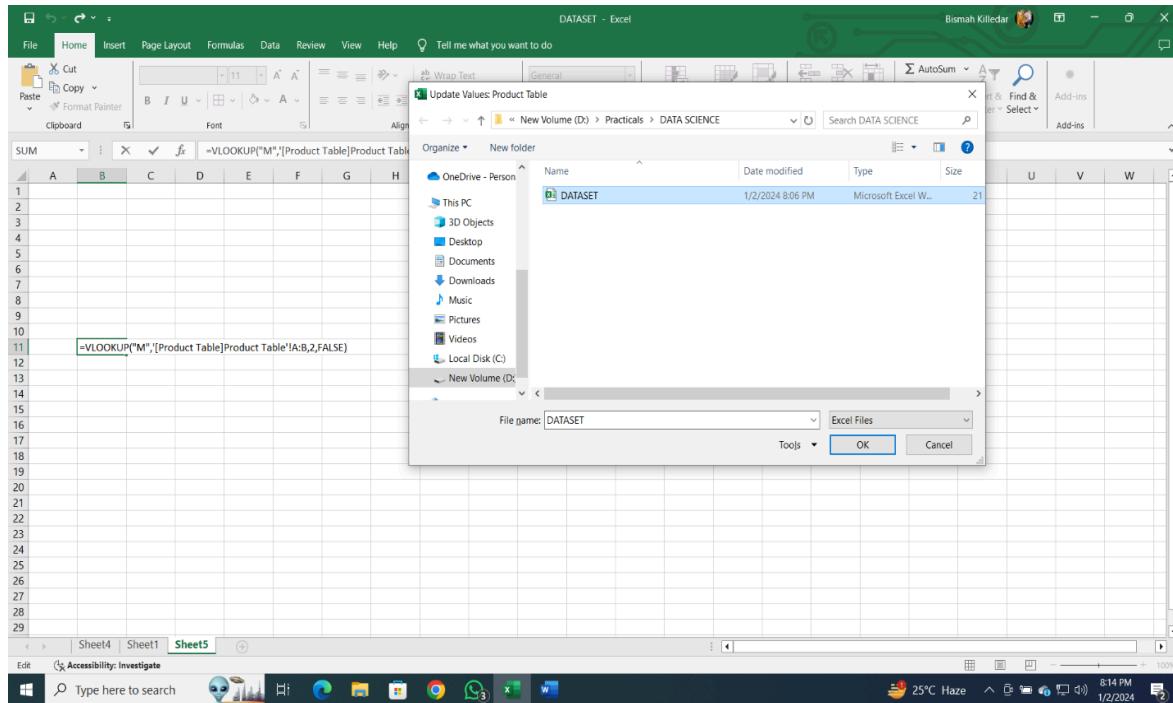
A screenshot of Microsoft Excel showing a bar chart titled "Sum of Sales" based on the PivotTable data. The chart shows sales for Clothing, Electronics, and Home Appliances. The PivotChart ribbon tab is selected. The chart is located in the range A1:Q16.

- **Use VLOOKUP function to retrieve information from a different worksheet or table.**

Use the VLOOKUP function to retrieve the category of "Product M" from a separate worksheet named "Product Table" using following steps:

**Steps:**

1. Assuming your "Product Table" is in a different worksheet.
2. In a cell in your main dataset, enter the formula:  
**=VLOOKUP("M", 'Product Table'!A:B, 2, FALSE)**



DATASET - Excel

Bismah Killeadar

File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do

C8 A B C D E F G H I J K L M N O P Q R S T U V W

1  
2  
3  
4  
5  
6  
7  
8 Clothing  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29

Sheet4 | Sheet1 | Sheet5 | +

Type here to search 25°C Haze 8:15 PM 1/2/2024

- **Perform what-if analysis using Goal Seek to determine input values for desired output.**

Use Goal Seek to find the required sales for "Product P" to achieve a profit of 1000 using the following steps.

### Steps:

1. Identify the cell containing the formula for "Profit" for "Product P" (let's assume it's in cell E17).
2. Go to the "Data" tab on the ribbon.
3. Click on "What-If Analysis" and select "Goal Seek."

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Product	Category	Sales	Cost	Profit										
2	A	Electronics	1000	500	500										
3	B	Clothing	1500	750	750										
4	C	Electronics	2000	1000	1000										
5	D	Home Appliances	1200	600	600										
6	E	Clothing	1800	900	900										
7	F	Electronics	900	450	450										
8	G	Home Appliances	1300	650	650										
9	H	Clothing	1600	800	800										
10	I	Electronics	1100	550	550										
11	J	Clothing	1400	700	700										
12	K	Home Appliances	1000	500	500										
13	L	Electronics	1900	950	950										
14	M	Clothing	2200	1100	1100										
15	N	Electronics	800	400	400										
16	O	Home Appliances	1700	850	850										
17	P	Clothing	1200	600	600										
18	Q	Electronics	1300	650	650										
19	R	Home Appliances	1600	800	800										
20	S	Clothing	900	450	450										
21	T	Electronics	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700	700										
		Clothing	1400	700	700										
		Home	1400	700	700										
		Appliances	1400	700	700										
		Electronics	1400	700</											

Screenshot of Microsoft Excel showing the Goal Seek dialog box. The formula bar shows =C17-D17. The Goal Seek dialog box is open, set to find a value of 1000 in cell E17 by changing cell \$C\$17.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Product	Category	Sales	Cost	Profit																		
2	A	Electronics	1000	500	500																		
3	B	Clothing	1500	750	750																		
4	C	Electronics	2000	1000	1000																		
5	D	Home Appliances	1200	600	600																		
6	E	Clothing	1800	900	900																		
7	F	Electronics	900	450	450																		
8	G	Home Appliances	1300	650	650																		
9	H	Clothing	1600	800	800																		
10	I	Electronics	1100	550	550																		
11	J	Clothing	1400	700	700																		
12	K	Home Appliances	1000	500	500																		
13	L	Electronics	1900	950	950																		
14	M	Clothing	2200	1100	1100																		
15	N	Electronics	800	400	400																		
16	O	Home Appliances	1700	850	850																		
17	P	Clothing	1200	600	600																		
18	Q	Electronics	1300	650	650																		
19	R	Home Appliances	1600	800	800																		
20	S	Clothing	900	450	450																		
21	T	Electronics	1400	700	700																		
		Home	1000	500	500																		

5. Click "OK" to let Excel determine the required sales.

Screenshot of Microsoft Excel showing the Goal Seek Status dialog box. It indicates that the goal seeking process has found a solution, with the target value being 1000 and the current value also being 1000.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	Product	Category	Sales	Cost	Profit																		
2	A	Electronics	1000	500	500																		
3	B	Clothing	1500	750	750																		
4	C	Electronics	2000	1000	1000																		
5	D	Home Appliances	1200	600	600																		
6	E	Clothing	1800	900	900																		
7	F	Electronics	900	450	450																		
8	G	Home Appliances	1300	650	650																		
9	H	Clothing	1600	800	800																		
10	I	Electronics	1100	550	550																		
11	J	Clothing	1400	700	700																		
12	K	Home Appliances	1000	500	500																		
13	L	Electronics	1900	950	950																		
14	M	Clothing	2200	1100	1100																		
15	N	Electronics	800	400	400																		
16	O	Home Appliances	1700	850	850																		
17	P	Clothing	1600	600	1000																		
18	Q	Electronics	1300	650	650																		
19	R	Home Appliances	1600	800	800																		
20	S	Clothing	900	450	450																		
21	T	Electronics	1400	700	700																		
		Home	1000	500	500																		

## Practical -2

### Aim: Data Frames and Basic Data Pre-processing

- Read data from CSV and JSON files into a data frame.
- Perform basic data pre-processing tasks such as handling missing values and outliers.
- Manipulate and transform data using functions like filtering, sorting, and grouping.

Requirements: Dataset : companydata.csv,

Platform: Google colab

### Dataset:

Job Position	Years of Experience	Salary (in USD per year)
CEO	5	100000
Senior manager	4	80000
Junior manager	3	NaN
Employee	NaN	40000
Assistant staff	1	20000

### Code & Output:

1. Read data from CSV and JSON files into a data frame.

```
import pandas as pd
```

```
#To create data set
```

```
#df = pd.DataFrame({'Job Position': ['CEO', 'Senior Manager', 'Junior Manager', 'Employee',  
'Assistant Staff'], 'Years of Experience':[5, 4, 3, None, 1],  
'Salary':[100000,80000,None,40000, 20000]})
```

```
#Read the data
```

```
df= pd.read_csv("companydata.csv")  
df
```

→

	Job Position	Years of Experience	Salary
0	CEO	5.0	100000.0
1	Senior Manager	4.0	80000.0
2	Junior Manager	3.0	NaN
3	Employee	NaN	40000.0
4	Assistant Staff	1.0	20000.0

## 2. Perform basic data pre-processing tasks such as handling missing values and outliers

```
# Handling missing values
```

```
df['Years of Experience'].fillna(df['Years of Experience'].median(), inplace=True)
```

```
df['Salary'].fillna(df['Salary'].mean(), inplace=True)
```

```
df
```

→

	Job Position	Years of Experience	Salary
0	CEO	5.0	100000.0
1	Senior Manager	4.0	80000.0
2	Junior Manager	3.0	60000.0
3	Employee	3.5	40000.0
4	Assistant Staff	1.0	20000.0

```
# Handling outliers in the 'Salary' column using winsorizing
```

```
df['Salary'] = winsorize(df['Salary'], limits=[0.05, 0.05])
```

```
df['Salary']
```

```
→ 0    100000.0  
 1    80000.0  
 2    60000.0  
 3    40000.0  
 4    20000.0  
Name: Salary, dtype: float64
```

## 3. Manipulate and transform data using functions like filtering, sorting, and grouping.

```
# Filtering: Selecting rows where 'Years of Experience' is not null
```

```
df_filtered = df[df['Years of Experience'].notnull()]
```

```
# Sorting: Sorting the DataFrame by 'Salary' in descending order
```

```

df_sorted = df.sort_values(by='Salary', ascending=False)

# Grouping: Calculating the average salary for each job position
df_grouped = df.groupby('Job Position')['Salary'].mean().reset_index()

# Viewing the results
print("Original DataFrame:")
print(df.head())
print("\nFiltered DataFrame (Rows with non-null 'Years of Experience'):")
print(df_filtered)
print("\nSorted DataFrame (Descending order by 'Salary'):")
print(df_sorted)
print("\nGrouped DataFrame (Average salary for each job position):")
print(df_grouped)

```

Original DataFrame:

	Job Position	Years of Experience	Salary
0	CEO	5.0	100000.0
1	Senior Manager	4.0	80000.0
2	Junior Manager	3.0	60000.0
3	Employee	3.5	40000.0
4	Assistant Staff	1.0	20000.0

Filtered DataFrame (Rows with non-null 'Years of Experience'):

	Job Position	Years of Experience	Salary
0	CEO	5.0	100000.0
1	Senior Manager	4.0	80000.0
2	Junior Manager	3.0	60000.0
3	Employee	3.5	40000.0
4	Assistant Staff	1.0	20000.0

Sorted DataFrame (Descending order by 'Salary'):

	Job Position	Years of Experience	Salary
0	CEO	5.0	100000.0
1	Senior Manager	4.0	80000.0
2	Junior Manager	3.0	60000.0
3	Employee	3.5	40000.0
4	Assistant Staff	1.0	20000.0

Grouped DataFrame (Average salary for each job position):

	Job Position	Salary
0	Assistant Staff	20000.0
1	CEO	100000.0
2	Employee	40000.0
3	Junior Manager	60000.0
4	Senior Manager	80000.0

## Practical - 3

### Aim: Feature Scaling and Dummification

- Apply feature-scaling techniques like standardization and normalization to numerical features.
- Perform feature dummification to convert categorical variables into numerical representations.

### Feature Scaling:

Feature scaling is a preprocessing technique used to standardize the range of independent variables or features of the data. It is essential for certain machine learning algorithms that are sensitive to the scale of input features, ensuring that all features contribute equally to the learning process.

### Feature Dummification:

Feature dummification or one-hot encoding is a technique used to convert categorical variables into numerical representations. This is necessary because many machine learning algorithms require numerical input, and representing categorical variables as binary vectors helps maintain their information.

### Steps:

1. **Load and Explore Data:** Load the dataset and explore its structure, identify numeric and categorical features.
2. **Feature Scaling:** Apply standardization and normalization to numeric features.
3. **Feature Dummification:** Convert categorical variables into numerical representations using one-hot encoding.
4. **Combine Features:** Combine scaled numeric features with one-hot encoded categorical features.
5. **Display Resulting Dataset:** Display the final dataset after both feature scaling and dummification.

### Code:

```
import pandas as pd
from sklearn.preprocessing import StandardScaler, MinMaxScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Define the data
data = {
    'Product': ['Apple_Juice', 'Banana_Smoothie', 'Orange_Jam', 'Grape_Jelly', 'Kiwi_Parfait',
    'Mango_Chutney', 'Pineapple_Sorbet', 'Strawberry_Yogurt', 'Blueberry_Pie', 'Cherry_Salsa'],
    'Category': ['Apple', 'Banana', 'Orange', 'Grape', 'Kiwi', 'Mango', 'Pineapple', 'Strawberry', 'Blueberry',
    'Cherry'],
    'Sales': [1200, 1700, 2200, 1400, 2000, 1000, 1500, 1800, 1300, 1600],
```

```

'Cost': [600, 850, 1100, 700, 1000, 500, 750, 900, 650, 800],
'Profit': [600, 850, 1100, 700, 1000, 500, 750, 900, 650, 800]
}

# Create a DataFrame
df = pd.DataFrame(data)

# Display the original dataset
print("Original Dataset:")
print(df)

# Step 1: Feature Scaling (Standardization and Normalization)
numeric_columns = ['Sales', 'Cost', 'Profit']
scaler_standardization = StandardScaler()
scaler_normalization = MinMaxScaler()
df_scaled_standardized = pd.DataFrame(scaler_standardization.fit_transform(df[numeric_columns]),
columns=numeric_columns)
df_scaled_normalized = pd.DataFrame(scaler_normalization.fit_transform(df[numeric_columns]),
columns=numeric_columns)

# Combine the scaled numeric features with the categorical features
df_scaled = pd.concat([df_scaled_standardized, df.drop(numeric_columns, axis=1)], axis=1)

# Display the dataset after feature scaling
print("\nDataset after Feature Scaling:")
print(df_scaled)

# Step 2: Feature Dummification
# Identify categorical columns
categorical_columns = ['Product', 'Category']

# Create a column transformer for dummification
preprocessor = ColumnTransformer(
    transformers=[
        ('categorical', OneHotEncoder(), categorical_columns)
    ],
    remainder='passthrough'
)

# Apply the column transformer to the dataset
df_dummified = pd.DataFrame(preprocessor.fit_transform(df))

# Display the dataset after feature dummification
print("\nDataset after Feature Dummification:")
print(df_dummified)

```

## Output:

→ Original Dataset:

	Product	Category	Sales	Cost	Profit
0	Apple_Juice	Apple	1200	600	600
1	Banana_Smoothie	Banana	1700	850	850
2	Orange_Jam	Orange	2200	1100	1100
3	Grape_Jelly	Grape	1400	700	700
4	Kiwi_Parfait	Kiwi	2000	1000	1000
5	Mango_Chutney	Mango	1000	500	500
6	Pineapple_Sorbet	Pineapple	1500	750	750
7	Strawberry_Yogurt	Strawberry	1800	900	900
8	Blueberry_Pie	Blueberry	1300	650	650
9	Cherry_Salsa	Cherry	1600	800	800

Dataset after Feature Scaling:

	Sales	Cost	Profit	Product	Category
0	-1.058873	-1.058873	-1.058873	Apple_Juice	Apple
1	0.372036	0.372036	0.372036	Banana_Smoothie	Banana
2	1.802946	1.802946	1.802946	Orange_Jam	Orange
3	-0.486509	-0.486509	-0.486509	Grape_Jelly	Grape
4	1.230582	1.230582	1.230582	Kiwi_Parfait	Kiwi
5	-1.631237	-1.631237	-1.631237	Mango_Chutney	Mango
6	-0.200327	-0.200327	-0.200327	Pineapple_Sorbet	Pineapple
7	0.658218	0.658218	0.658218	Strawberry_Yogurt	Strawberry
8	-0.772691	-0.772691	-0.772691	Blueberry_Pie	Blueberry
9	0.085855	0.085855	0.085855	Cherry_Salsa	Cherry

Dataset after Feature Dummification:

	0
0	(0, 0)\t1.0\n (0, 10)\t1.0\n (0, 20)\t1200...
1	(0, 1)\t1.0\n (0, 11)\t1.0\n (0, 20)\t1700...
2	(0, 7)\t1.0\n (0, 17)\t1.0\n (0, 20)\t2200...
3	(0, 4)\t1.0\n (0, 14)\t1.0\n (0, 20)\t1400...
4	(0, 5)\t1.0\n (0, 15)\t1.0\n (0, 20)\t2000...
5	(0, 6)\t1.0\n (0, 16)\t1.0\n (0, 20)\t1000...
6	(0, 8)\t1.0\n (0, 18)\t1.0\n (0, 20)\t1500...
7	(0, 9)\t1.0\n (0, 19)\t1.0\n (0, 20)\t1800...
8	(0, 2)\t1.0\n (0, 12)\t1.0\n (0, 20)\t1300...
9	(0, 3)\t1.0\n (0, 13)\t1.0\n (0, 20)\t1600...

## Practical - 4

### Aim: Hypothesis Testing

- Formulate null and alternative hypotheses for a given problem.
- Conduct a hypothesis test using appropriate statistical tests (e.g., t-test, chisquare test).
- Interpret the results and draw conclusions based on the test outcomes.

### Hypothesis Testing:

Hypothesis testing is a statistical method used to make inferences about population parameters based on sample data. It involves the formulation of a null hypothesis ( $H_0$ ) and an alternative hypothesis ( $H_1$ ), and the collection of sample data to assess the evidence against the null hypothesis. The goal is to determine whether there is enough evidence to reject the null hypothesis in favor of the alternative hypothesis.

#### 1. Formulate Hypotheses:

- Null Hypothesis ( $H_0$ ): The average caffeine content per serving is 80 mg ( $\mu=80$ ).
- Alternative Hypothesis ( $H_1$ ): The average caffeine content per serving is different from 80 mg ( $\mu\neq80$ ).

#### 2. Statistical Test:

- A t-test is appropriate since you are comparing a sample mean to a known population mean, and the sample size is small.

#### 3. Data Collection:

- Randomly select 30 cans of the energy drink and measure the caffeine content in each.

#### 4. Conducting the Hypothesis Test:

##### a. Collect Data:

- Calculate the sample mean ( $\bar{x}$ ) and standard deviation ( $s$ ) from the 30 samples.

##### b. Set Significance Level ( $\alpha$ ):

- Choose a significance level ( $\alpha=0.05, 0.01, 0.10$ ).

##### c. Calculate the Test Statistic (t-value):

- Use the formula  $t=\frac{\bar{x}-\mu}{s/\sqrt{n}}$ .

##### d. Determine Degrees of Freedom:

- For a one-sample t-test, degrees of freedom ( $df$ ) is  $n-1$ .

##### e. Find Critical Values or P-value:

- Use a t-table or statistical software to find the critical t-values for a two-tailed test at the chosen significance level.

##### f. Make a Decision:

- If the t-value falls outside the critical region, reject the null hypothesis. If it falls inside, fail to reject.

##### g. Interpretation:

- If you reject the null hypothesis, there is enough evidence to suggest that the average caffeine content per serving is different from 80 mg. If you fail to reject the null hypothesis, there is not enough evidence to suggest a difference in the average caffeine content.

## 5. Conclusion:

- Draw conclusions about the energy drink's caffeine content, considering both statistical and practical significance. Consider decisions relevant to the context of the problem.

### Code:

```
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt

# Generate two samples for demonstration purposes
np.random.seed(42)
sample1 = np.random.normal(loc=10, scale=2, size=30)
sample2 = np.random.normal(loc=12, scale=2, size=30)

# Perform a two-sample t-test
t_statistic, p_value = stats.ttest_ind(sample1, sample2)

# Set the significance level
alpha = 0.05
print("Results of Two-Sample t-test:")
print(f"t-statistic: {t_statistic}")
print(f"p-value: {p_value}")
print(f"Degrees of Freedom: {len(sample1) + len(sample2) - 2}")

# Plot the distributions
plt.figure(figsize=(10, 6))
plt.hist(sample1, alpha=0.5, label='Sample 1', color='blue')
plt.hist(sample2, alpha=0.5, label='Sample 2', color='orange')
plt.axvline(np.mean(sample1), color='blue', linestyle='dashed', linewidth=2)
plt.axvline(np.mean(sample2), color='orange', linestyle='dashed', linewidth=2)
plt.title('Distributions of Sample 1 and Sample 2')
plt.xlabel('Values')
plt.ylabel('Frequency')
plt.legend()

# Highlight the critical region if null hypothesis is rejected
if p_value < alpha:
    critical_region = np.linspace(min(sample1.min(), sample2.min()), max(sample1.max(), sample2.max()), 1000)
    plt.fill_between(critical_region, 0, 5, color='red', alpha=0.3, label='Critical Region')

# Show the observed t-statistic
plt.text(11, 5, f'T-statistic: {t_statistic:.2f}', ha='center', va='center', color='black', backgroundcolor='white')

# Show the plot
plt.show()

# Draw Conclusions
if p_value < alpha:
    if np.mean(sample1) > np.mean(sample2):
        print("Conclusion: There is significant evidence to reject the null hypothesis.")
```

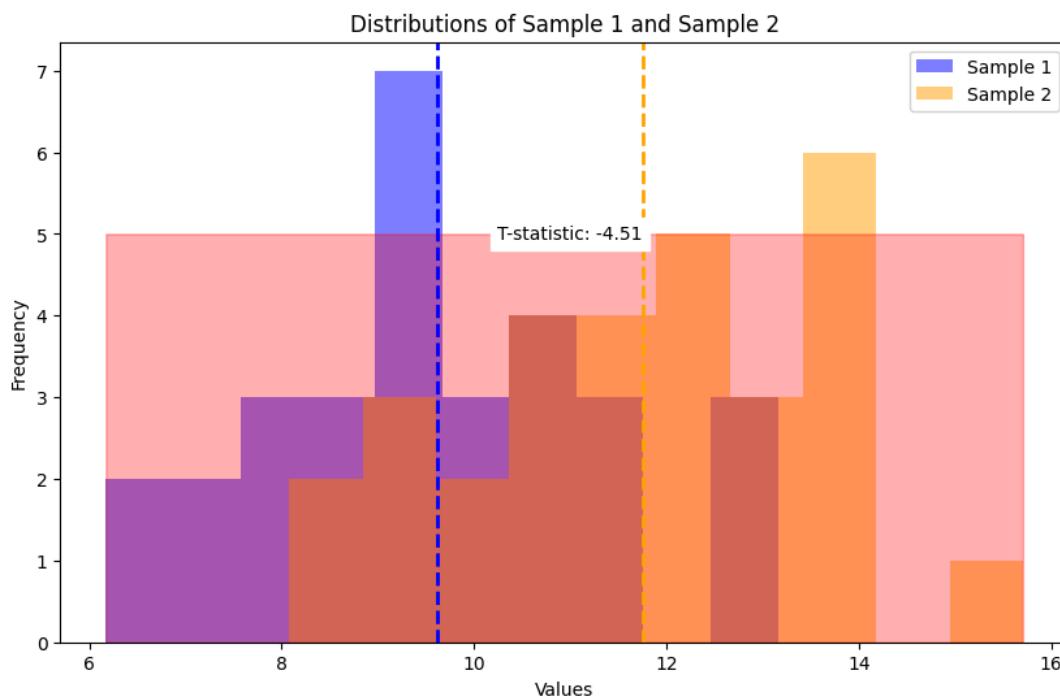
```

print("Interpretation: The mean caffeine content of Sample 1 is significantly higher than that of Sample
2.")
else:
    print("Conclusion: There is significant evidence to reject the null hypothesis.")
    print("Interpretation: The mean caffeine content of Sample 2 is significantly higher than that of Sample
1.")
else:
    print("Conclusion: Fail to reject the null hypothesis.")
    print("Interpretation: There is not enough evidence to claim a significant difference between the means.")

```

## Output:

Results of Two-Sample t-test:  
t-statistic: -4.512913234547555  
p-value: 3.176506547470154e-05  
Degrees of Freedom: 58



Conclusion: There is significant evidence to reject the null hypothesis.  
Interpretation: The mean caffeine content of Sample 2 is significantly higher than that of Sample 1.

## Practical - 5

### Aim: ANOVA (Analysis of Variance)

- Perform one-way ANOVA to compare means across multiple groups.
- Conduct post-hoc tests to identify significant differences between group means.

Requirements: RStudio, WholesaleCustomersData.csv

Steps:

Download the “WholesaleCustomersData.csv” from Kaggle

Open RStudio > Create A New Project > Import Dataset > From Text (base)

In the Console, Type the following Command-

**WholesaleCustomerData\$Region <- factor(WholesaleCustomerData\$Region)**

Here, “WholesaleCustomerData” is the name of the dataset

# Perform the ANOVA

**one.way <- aov(Grocery ~ Region, data = WholesaleCustomerData)**

# Check the summary

**summary(one.way)**

# Try Tukey's HSD test

**TukeyHSD(one.way)**

---

### Comparing Multiple Groups

```
> wholesalecustomerData$Milk <- factor(wholesalecustomerData$Milk)
> one.way <- aov(Grocery ~ Milk, data = wholesalecustomerData)
> summary(one.way)
    Df  Sum Sq Mean Sq F value Pr(>F)
Milk     420 3.947e+10 93969190    9.97 2.75e-07 ***
Residuals 19 1.791e+08  9425030
---
Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> TukeyHSD(one.way)
  Tukey multiple comparisons of means
  95% family-wise confidence level

Fit: aov(formula = Grocery ~ Milk, data = wholesalecustomerData)

$Milk
      diff      lwr      upr      p adj
112-55 6.410000e+02 -25236.798668 26518.79867 1.0000000
134-55 8.100000e+01 -25796.798668 25958.79867 1.0000000
201-55 1.080000e+02 -25769.798668 25985.79867 1.0000000
254-55 4.730000e+02 -25404.798668 26350.79867 1.0000000
259-55 1.001000e+02 -24976.798668 26878.79867 1.0000000
```

```

> wholesaleCustomerData$Delicassen <- factor(wholesaleCustomerData$Delicassen)
> one.way <- aov(Detergents_Paper ~ Delicassen, data = wholesaleCustomerData)
> summary(one.way)
   Df  Sum Sq Mean Sq F value Pr(>F)
Delicassen 402 8.828e+09 21961336  0.706  0.942
Residuals   37 1.151e+09 31110332
> TukeyHSD(one.way)
  Tukey multiple comparisons of means
  95% family-wise confidence level

Fit: aov(formula = Detergents_Paper ~ Delicassen, data = wholesaleCustomerData)

$Delicassen
        diff      lwr      upr     p adj
7-3    -9.352500e+02 -34881.96178 33011.462 1.0000000
8-3    -1.337250e+03 -35283.96178 32609.462 1.0000000
11-3   -1.010250e+03 -34956.96178 32936.462 1.0000000
18-3   -6.492500e+02 -26944.25987 25645.760 1.0000000
22-3   -1.312250e+03 -35258.96178 32634.462 1.0000000
27-3   2.123750e+03 -31822.96178 36070.462 1.0000000

> wholesaleCustomerData$Fresh <- factor(wholesaleCustomerData$Fresh)
> one.way <- aov(Detergents_Paper ~ Fresh, data = wholesaleCustomerData)
> summary(one.way)
   Df  Sum Sq Mean Sq F value Pr(>F)
Fresh      432 9.888e+09 22889849  1.758  0.219
Residuals     7 9.112e+07 13017787
> TukeyHSD(one.way)
  Tukey multiple comparisons of means
  95% family-wise confidence level

Fit: aov(formula = Detergents_Paper ~ Fresh, data = wholesaleCustomerData)

$Fresh
        diff      lwr      upr     p adj
9-3    3.349000e+03 -30768.0902 37466.0902 1.0000000
18-3   4.678000e+03 -29439.0902 38795.0902 1.0000000
23-3   3.755000e+03 -30362.0902 37872.0902 1.0000000
37-3   6.628000e+03 -27489.0902 40745.0902 1.0000000

```

## Practical - 6

### Aim: Regression and Its Types

- Implement simple linear regression using a dataset.
- Explore and interpret the regression model coefficients and goodness-of-fit measures.
- Extend the analysis to multiple linear regression and assess the impact of additional predictors

Requirements: RStudio, LungCap.xls

Download the “LungCap.xls” from Kaggle

Open RStudio > Create A New Project > Import Dataset > From Excel > Browse > Import

In the Console, Type the following Command-

```
>LungCap <- read_excel("C:/Users/admin/Downloads/LungCap.xls")
> View(LungCap)
> attach(LungCap)
> names(LungCap)
> class(`LungCap(cc)`)
> class(`Age( years)`)
> class(`Height(inches)`)
> class(Smoke)
> class(Gender)
> class(Caesarean)
> plot(`Age( years)`, `LungCap(cc)`, main = "Scatterplot")
> cor(`Age( years)`, LungCap$`LungCap(cc)` )
> mod<-lm(LungCap$`LungCap(cc)`~`Age( years)` )
> summary(mod)
```

### Outputs:

```
> LungCap <- read_excel("C:/Users/admin/Downloads/LungCap.xls")
> View(LungCap)
> attach(LungCap)
```

```

> names(LungCap)
[1] "LungCap(cc)"      "Age( years)"      "Height(inches)" "Smoke"           "Gender"
[5] "Caesarean"
> class(`LungCap(cc)`)
[1] "numeric"
> class(`Age( years)` )
[1] "numeric"
> class(`Height(inches)` )
[1] "numeric"
> class(Smoke)
[1] "character"
> class(Gender)
[1] "character"
> class(Caesarean)
[1] "character"
> plot(`Age( years)` ,`LungCap(cc)` , main = "scatterplot")
> cor(`Age( years)` , LungCap$`LungCap(cc)` )
[1] 0.8196749
> mod<-lm(LungCap$`LungCap(cc)` ~ `Age( years)` )
> summary(mod)

```

call:  
`lm(formula = LungCap$`LungCap(cc)` ~ `Age( years)` )`

Residuals:

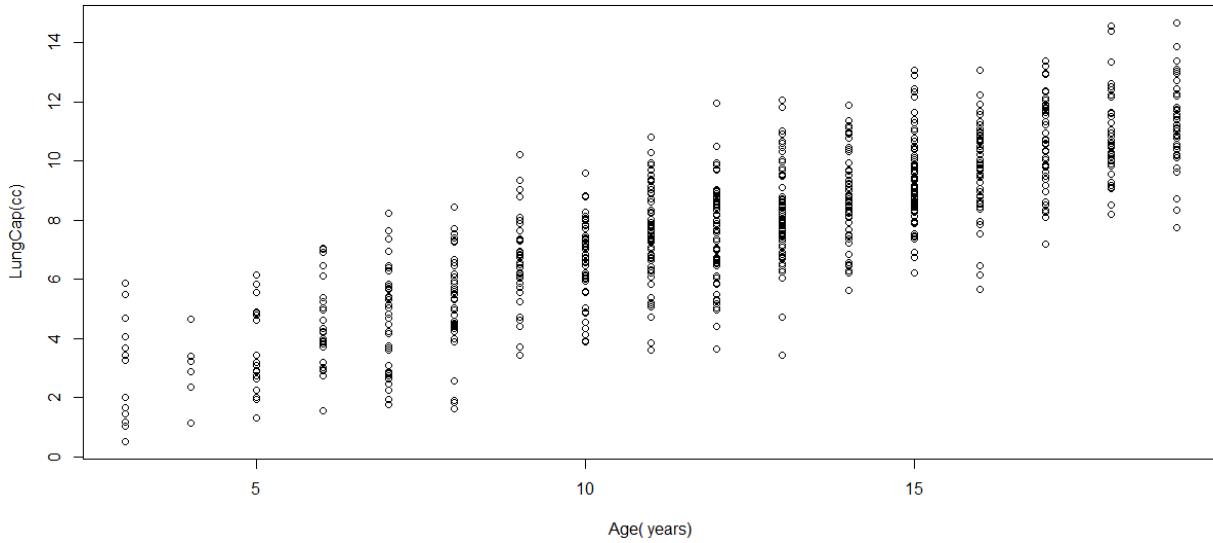
Min	1Q	Median	3Q	Max
-4.7799	-1.0203	-0.0005	0.9789	4.2650

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	1.14686	0.18353	6.249	7.06e-10 ***
`Age( years)`	0.54485	0.01416	38.476	< 2e-16 ***
---				
signif. codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'
	0.1 ' '	1		

Residual standard error: 1.526 on 723 degrees of freedom  
Multiple R-squared: 0.6719, Adjusted R-squared: 0.6714  
F-statistic: 1480 on 1 and 723 DF, p-value: < 2.2e-16

Scatterplot



## Practical 7

### Aim: Logistic Regression and Decision Tree

- Build a logistic regression model to predict a binary outcome.
- Evaluate the model's performance using classification metrics (e.g., accuracy, precision, recall).
- Construct a decision tree model and interpret the decision rules for classification

Requirements: RStudio, Iris.csv

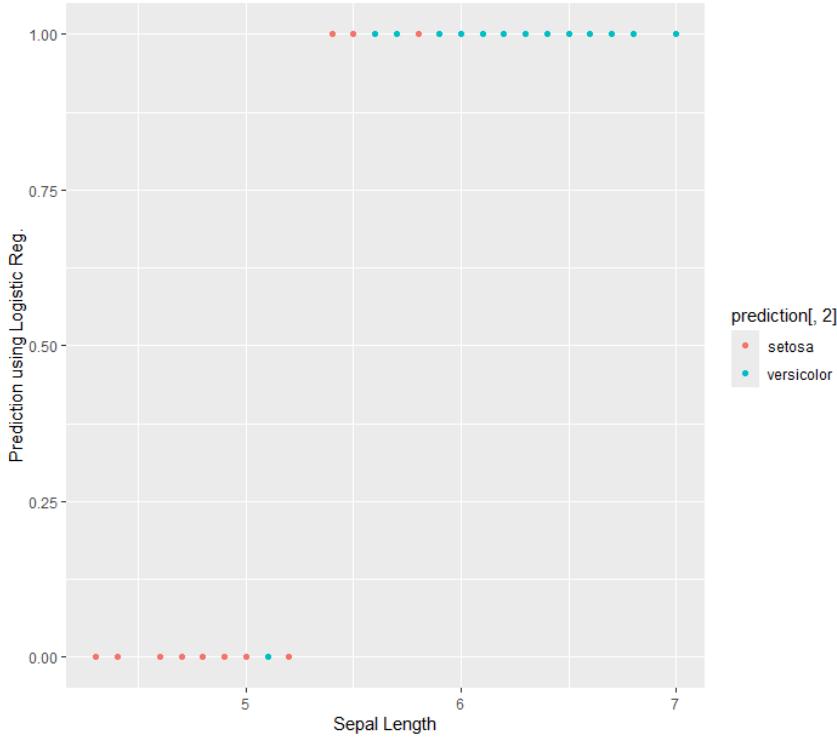
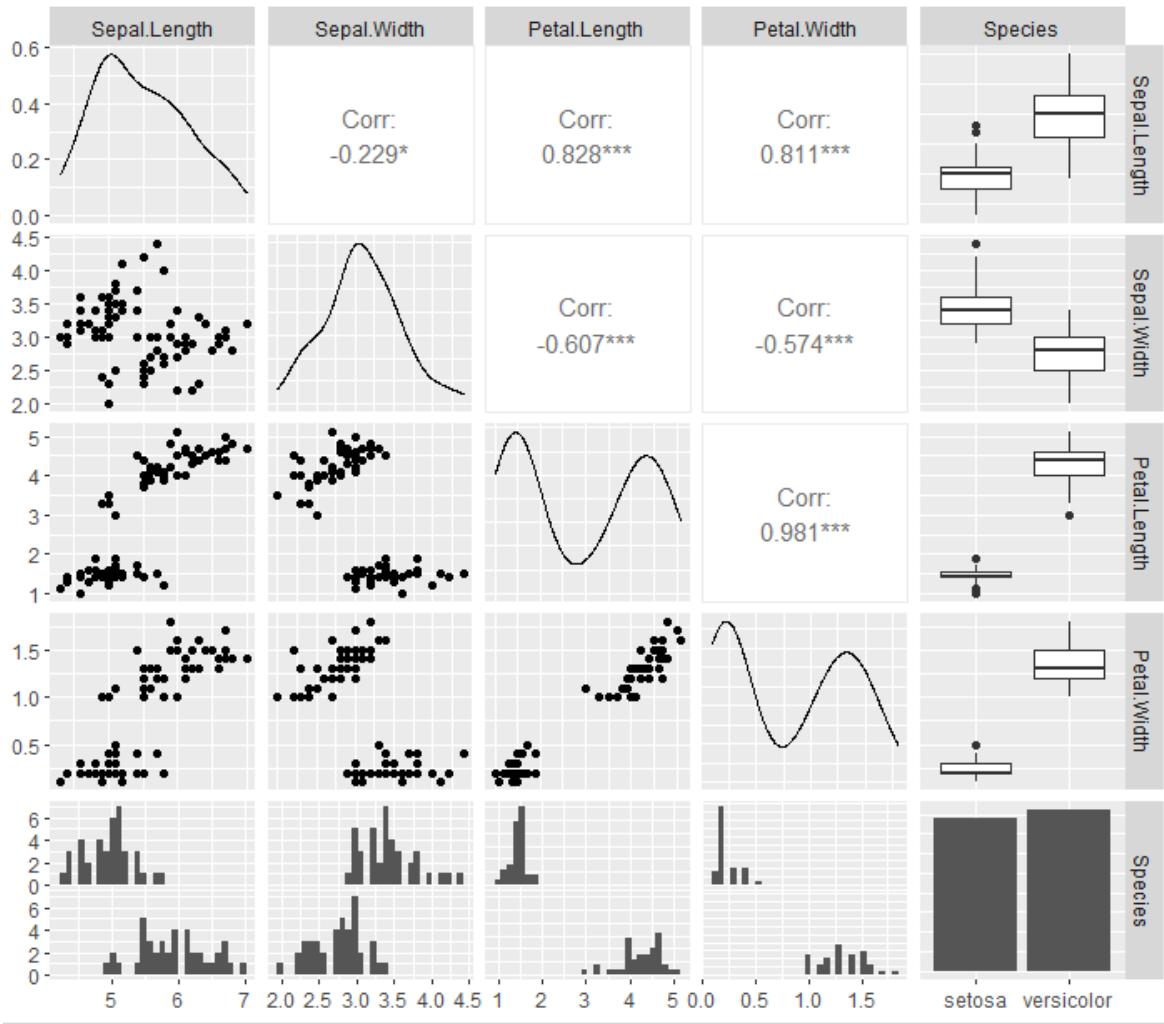
Steps: Download the “Iris.csv” from Kaggle

Open RStudio > Create A New Project > Import Dataset > From Text(Base) > Browse > Import

In the Console, Type the following Command-

```
> Iris <- read.csv("C:/Users/admin/Downloads/Iris.csv")
> View(Iris)
> library(datasets)
> ir_data<-iris
> head(ir_data)
> str(ir_data)
> levels(ir_data$Species)
> sum(is.na(ir_data))
> ir_data<-ir_data[1:100,]
> set.seed(100)
> samp <- sample(1:100, 80)
> ir_test<-ir_data[samp,]
> ir_ctrl<-ir_data[samp,]
> install.packages("ggplot2")
> library(ggplot2)
> install.packages("GGally")
> library(GGally)
> ggpairs(ir_test)
> y<-ir_test$Species; x<-ir_test$Sepal.Length
> glfit<-glm(y~x,family = 'binomial')
> summary(glfit)
> newdata<-data.frame(x=ir_ctrl$Sepal.Length)
> predicted_val<-predict(glfit,newdata, type = "response")
> prediction<-data.frame(ir_ctrl$Sepal.Length,ir_ctrl$Species,predicted_val)
> prediction
> qplot(prediction[,1],round(prediction[,3]),col=prediction[,2],xlab='Sepal Length', ylab='Prediction using Logistic Reg.')
)
```

Outputs:



## Practical - 8

### Aim: K-Means Clustering

- Apply the K-Means algorithm to group similar data points into clusters.
- Determine the optimal number of clusters using elbow method or silhouette analysis.
- Visualize the clustering results and analyze the cluster characteristics.

Requirements: RStudio, Iris.csv

Steps: Download the “WholesaleCustomerData.csv” from Kaggle

Open RStudio > Create A New Project > Import Dataset > From Text(Base) > Browse > Import

In the Console, Type the following Command-

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt
data= pd.read_csv('WholesaleCustomerData.csv')
data.head()
categorical_features = ['Channel','Region']
continuous_features = ['Fresh','Milk','Grocery','Frozen','Detergents_Paper','Delicassen']
data[continuous_features].describe()
for col in categorical_features:
    dummies = pd.get_dummies(data[col], prefix=col)
    data = pd.concat([data, dummies], axis=1)
    data.drop(col, axis=1, inplace=True)

data.head()
mms = MinMaxScaler()
mms.fit(data)
data_transformed = mms.transform(data)
from sklearn.cluster import KMeans
import numpy as np

Sum_of_squared_distances = []
K = range(1, 15)

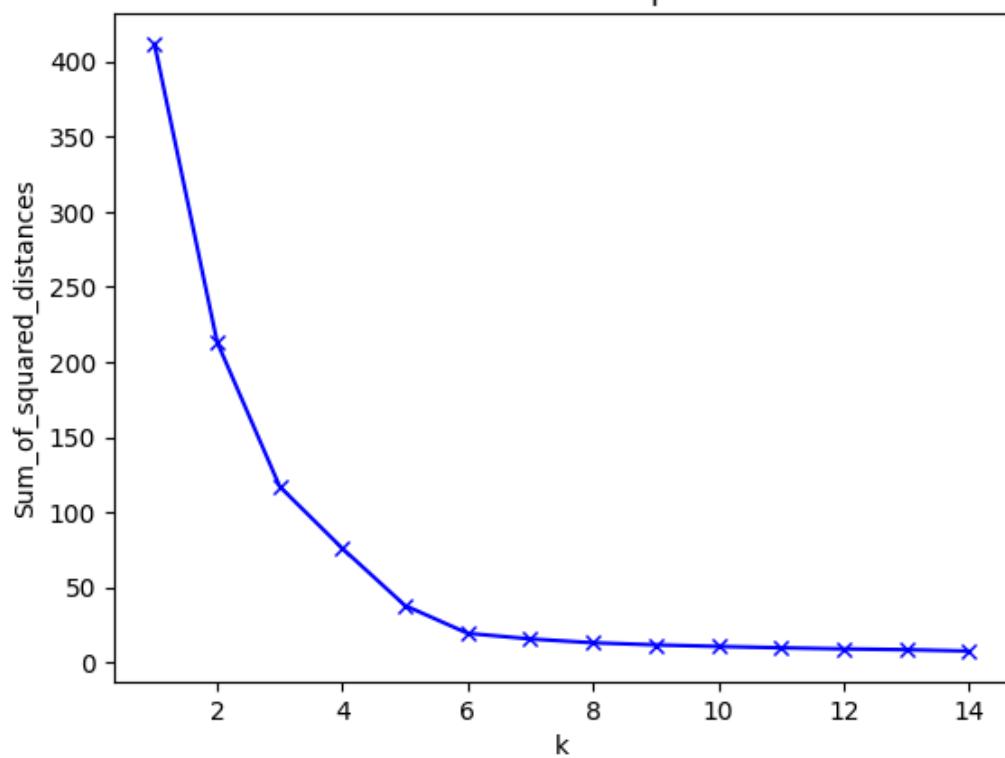
for k in K:
    km = KMeans(n_clusters=k)
    km = km.fit(data_transformed)
    Sum_of_squared_distances.append(km.inertia_)

plt.plot(K, Sum_of_squared_distances, 'bx-')
plt.xlabel('k')
plt.ylabel('Sum_of_squared_distances')
plt.title('Elbow Method For Optimal k')
```

```
plt.show()
```



Elbow Method For Optimal k



## Practical - 9

### Aim: Principal Component Analysis (PCA)

- Perform PCA on a dataset to reduce dimensionality.
- Evaluate the explained variance and select the appropriate number of principal components.
- Visualize the data in the reduced-dimensional space

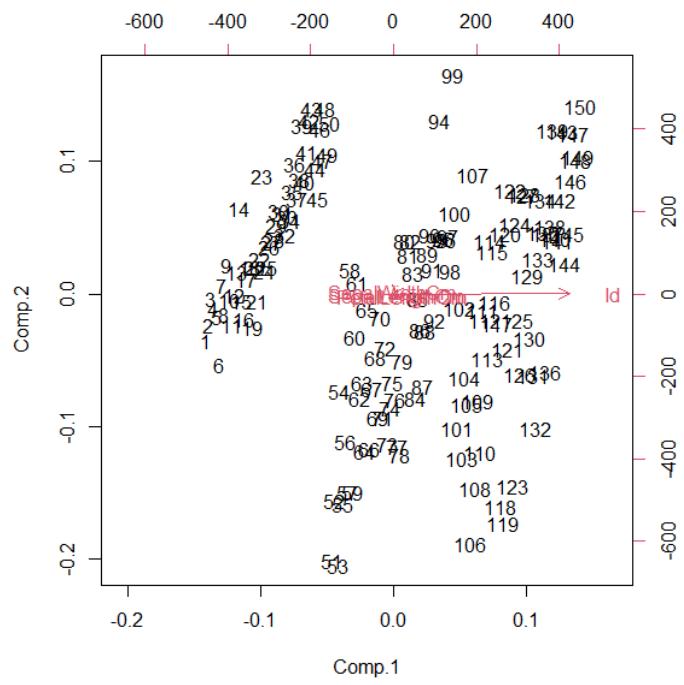
Requirements: RStudio, Iris.csv

Steps: Download the “Iris.csv” from Kaggle

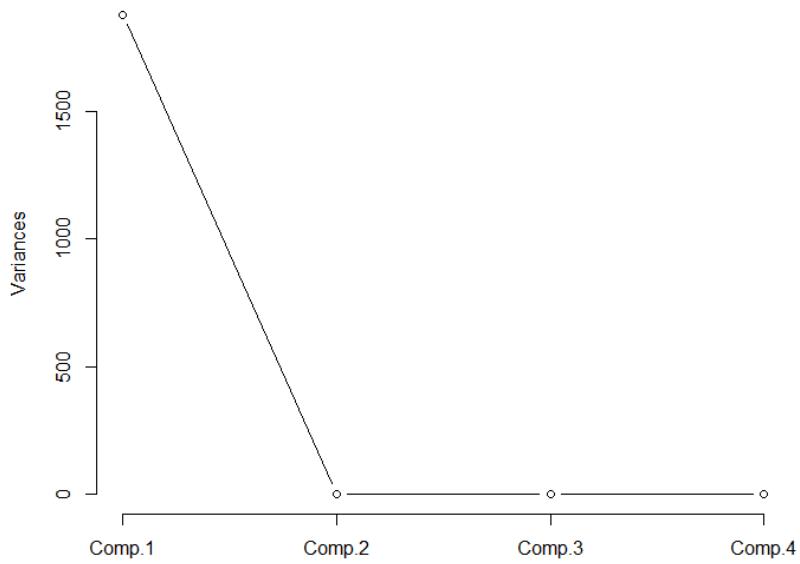
Open RStudio > Create A New Project > Import Dataset > From Text(Base) > Browse > Import

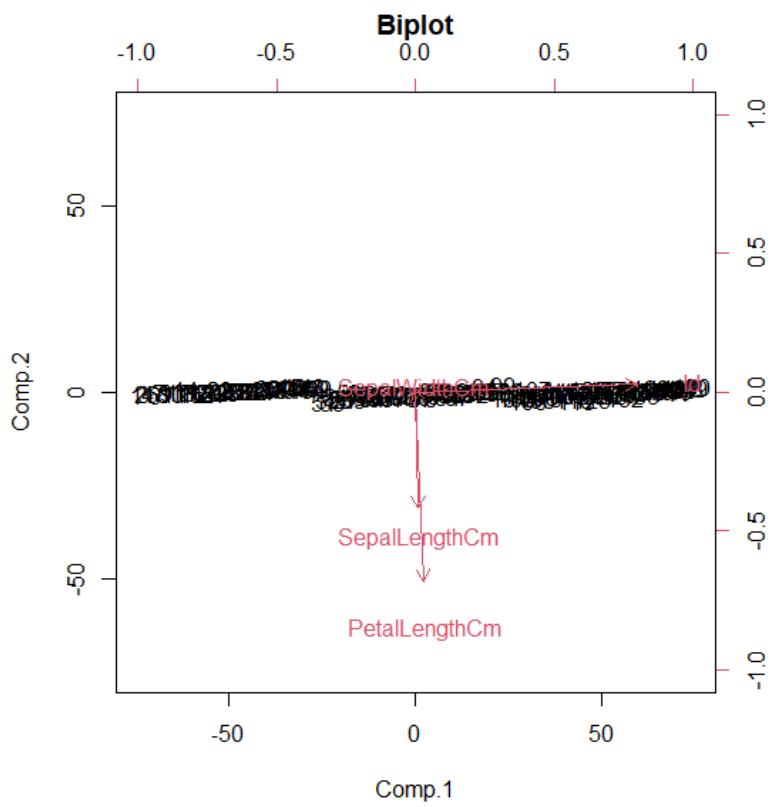
In the Console, Type the following Command-

```
> data_iris <- iris[1:4]
> cov_data <- cov(data_iris)
> Eigen_data <- eigen(cov_data)
> PCA_data <- princomp(data_iris,cor = "False")
> Eigen_data$values
> PCA_data$sdev^2
> PCA_data$loadings[,1:4]
> Eigen_data$vectors
> summary(PCA_data)
> biplot(PCA_data)
> screeplot(PCA_data,type = 'lines')
> model2 = PCA_data$loadings[,1]
> model2_scores <- as.matrix(data_iris)%%model2
> library(class)
> install.packages("e1071")
> library(e1071)
> mod1 <- naiveBayes(iris[,1:4],iris[,5])
> mod2 <- naiveBayes(model2_scores,iris[,5])
> table(predict(mod1,iris[,1:4]),iris[,5])
> table(predict(mod2,model2_scores),iris[,5])
> biplot(PCA_data, main = "Biplot", scale = 0)
```



**PCA\_data**



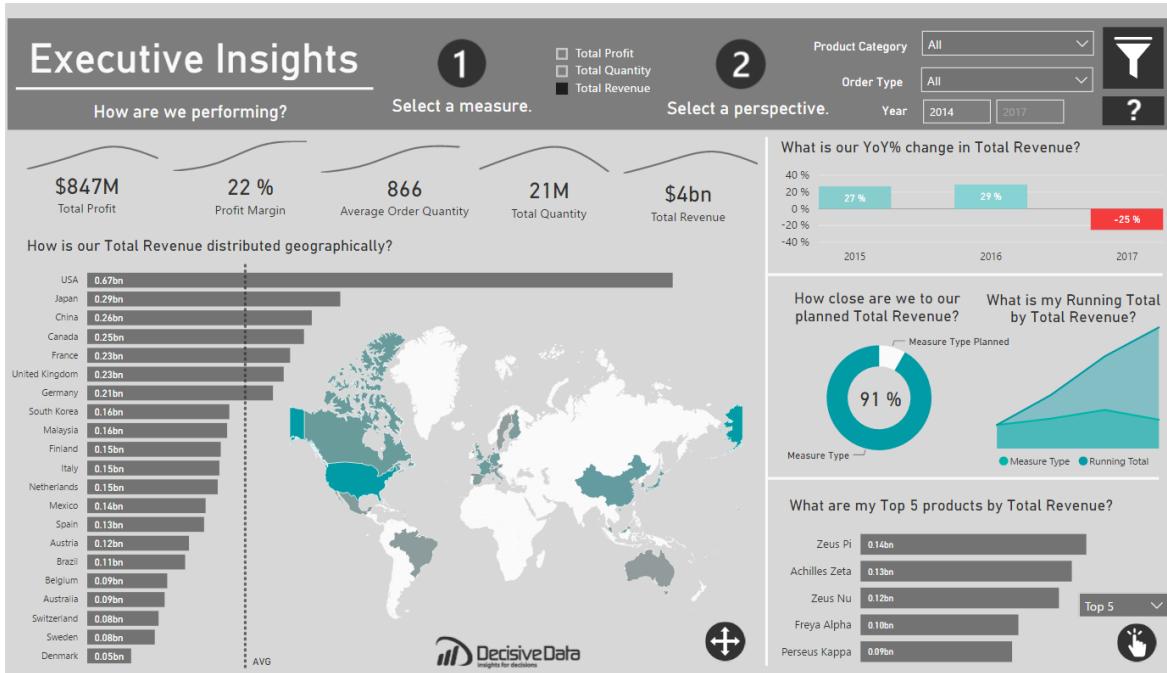


## Practical 10

### Aim: Data Visualization and Storytelling

- Create meaningful visualizations using data visualization tools
- Combine multiple visualizations to tell a compelling data story.
- Present the findings and insights in a clear and concise manner

### Executive Insights by Decisive Data



### Summary

Often asking, "How are we performing?" can be a question that cascades into a series of further questions, spinoffs and investigative research. This is especially true for globally minded companies. I wanted to create a report that preemptively addressed this kind of exploration. This report is meant to provide data-driven decision making, while emphasizing user-flexibility and visual analysis. I was able to achieve my goal to empower the user by leveraging dynamic visuals. Thus, this dashboard can scale as the needs of the global business changes.

### Approach

A dashboard is most valuable when you immediately understand what you can do. I looked into what attributes that can be influenced by the company:

- *What we are selling (products)*
- *When we are selling (year)*
- *Where we are selling (country)*
- *How we are selling (order type)*

The focus here is to view the business from multiple angles for these attributes, providing a holistic approach to the business, through dynamic parameters. I was able to provide deeper analysis with Top/Bottom products, YoY growth and Running totals to name a few. I leveraged bar charts, line charts, donut charts and custom visuals for visual analysis.

### Pushing the Boundaries

There were two things I wished to achieve with this dashboard, attractive mapping and next-level user interactivity.

I wanted to have a map that was design oriented and subtle, like an art piece, meant to invite the user into the dashboard. *If users are going to be using a report all the time, why not make it pleasing to the eye?* I found that design piece with the new “Shape Map” feature, where one can import custom TopoJSON files. I edited and imported a custom world map from <http://mapstarter.com/>. The simple map was meant to ground the user geographically and complement the adjacent bar chart, which holds the same information.

Moreover, I am a firm believer that interactivity is empowering. Giving users the tools to investigate data on their own terms is liberating. As such, I wanted to provide *two* layers of dynamic parameters. This was done with a Top 5/Bottom 5 measure that reflects whatever measure the user has selected at the top of the dashboard. This was executed within DAX. The question I wanted to be able to answer was “What is my [Top/Bottom] products by [Value]?” in one, simple, clean visual.

For the first part of this method of creating dynamic measures, I was inspired by Sam McKay’s wonderful blog post on [Dynamic Visuals](#), within Power BI. This provided me the groundwork for my dynamic values, and how I would approach the rest of this problem.

Now came the real challenge. I had to build two ranks on a value that could change with a click. To do this I set my dynamic parameter as two rank fields, one Ascending and the other to Descending, for the respective ends of the product performance. These were calculated across my desired field, Products.

Next, I needed to filter each of these Ranks to only keep the Top 5 Products. This can be down with two IF statements, keeping the Rank with only five values or else appearing as BLANK(). Once the IFs are built, you can then build another selector table with McKay’s method with a “Top 5” and “Bottom 5”, which can eventually be used with a SWITCH function to include only the [Filtered Rank DESC] or the [Filtered Rank ASC].

The result means that instead of **six** individual bar charts, one for a Top or Bottom, across three measures, Quantity, Revenue and Profit, you end up with **one**, dynamic visual. The result empowers users and saves valuable real estate for other interesting insights.

Quality over quantity.

### Wrapping it up

The dynamic parameters and tables worked very well, but it took a little research to have the two dynamic parameters work in conjunction, as I had yet to see any specific topic utilize such before in Power BI. Another aspect that comes to mind, is whether one could make the number of values, in this case 5, also a parameter, so you could select how many TopN you are seeing. Food for thought. In the end, it’s great to know that great visuals can be created just with DAX, determination and some trial and error.

Overall, with the combination of the custom visuals, and dynamic parameters, I created a clean dashboard for analytical insights that’s also a pleasure to interact with.