## Name : Prathmesh Hande

## Branch : IT

## Roll No : 46

## Teachers Assesment - 1 of Tools for data Science

--prof Ashwini Gote

# 1.Data Analysis with Pandas and Matplotlib. (1.5)

### Objective: Perform data analysis on a given dataset using Pandas and visualize the results using Matplotlib.

### Requirements:

Choose a dataset (e.g., CSV, Excel, or any other format) related to a topic of interest (e.g., finance, sports, health). Use Pandas to load and clean the data. Perform basic statistical analysis (mean, median, standard deviation). Create meaningful visualizations using Matplotlib (e.g., bar chart, line plot, scatter plot).

Provide insights or conclusions based on the analysis.

In [14]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [19]:
```
# Import the dataset
ds = pd.read_csv(r"C:\Users\prasa\OneDrive\Desktop\TA TOOLS FOR DS\EmployeeSampleData\100 Sales Records.csv")
```

In [20]:
```
# Display structure of the data
print(ds.head())
```

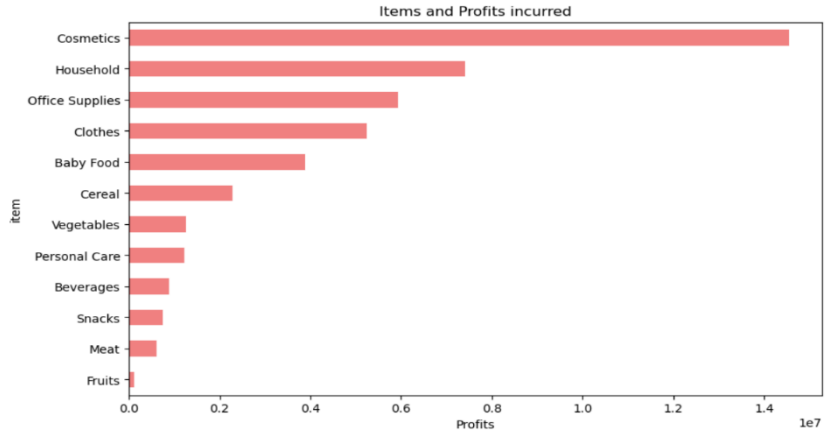
```
                           Region               Country        Item Type  \
0              Australia and Oceania                Tuvalu        Baby Food
1  Central America and the Caribbean               Grenada           Cereal
2                          Europe                Russia  Office Supplies
3                  Sub-Saharan Africa  Sao Tome and Principe           Fruits
4                  Sub-Saharan Africa                Rwanda  Office Supplies

  Sales Channel Order Priority Order Date   Order ID  Ship Date  Units Sold  \
0       Offline              H  5/28/2010  669165933  6/27/2010        9925
1        Online              C  8/22/2012  963881480  9/15/2012        2804
2       Offline              L   5/2/2014  341417157   5/8/2014        1779
3        Online              C  6/20/2014  514321792   7/5/2014        8102
4       Offline              L   2/1/2013  115456712   2/6/2013        5062

   Unit Price  Unit Cost  Total Revenue  Total Cost  Total Profit
0      255.28     159.42     2533654.00  1582243.50     951410.50
1      205.70     117.11      576782.80   328376.44     248406.36
2      651.21     524.96     1158502.59   933903.84     224598.75
3        9.33       6.92       75591.66    56065.84      19525.82
4      651.21     524.96     3296425.02  2657347.52     639077.50
```

In [28]:
```
mean_Total_Revenue = ds['Total Revenue'].mean()
median_Units_Sold = ds['Units Sold'].median()
std_deviation = ds['Units Sold'].std()
```

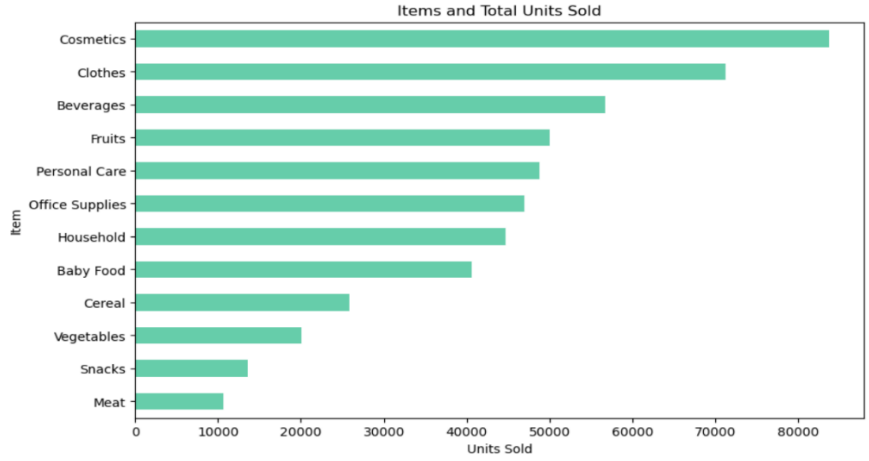
In [29]:
```
print(mean_Total_Revenue)
print(median_Units_Sold)
print(std_deviation)
```

```
1373487.6831
5382.5
2794.4845616956904
```

In [33]:
```
# Bar chart
plt.figure(figsize=(10, 6))
ds.groupby('Item Type')['Total Profit'].sum().sort_values().plot(kind='barh', color='lightcoral')
plt.title('Items and Profits incurred')
plt.xlabel('Profits')
plt.ylabel('item')
plt.show()
```




In [32]:
```
# Bar chart
plt.figure(figsize=(10, 6))
ds.groupby('Item Type')['Units Sold'].sum().sort_values().plot(kind='barh', color='mediumaquamarine')
plt.title('Items and Total Units Sold')
plt.xlabel('Units Sold')
plt.ylabel('Item')
plt.show()
```




# CONCLUSIONS:

1. Top three most profitable item types are Cosmetics, Household, and Office Supplies, with profits of $1,717,540.03, $808,643.42, and $539,811.25, respectively.

2. Top three highest-selling item types are Clothes, Office Supplies, and Beverages, with units sold totaling 42,251, 36,915, and 34,534, respectively.

3. Allocate additional marketing resources to promote these high-profit items. Consider targeted advertising campaigns, bundling deals, or loyalty programs to enhance customer engagement. Implement strategies to capitalize on the popularity of these items. Introduce limited-time promotions, discounts, or exclusive offers for Clothes, Office Supplies, and Beverages to stimulate demand. Explore cross-selling opportunities, where customers purchasing one of these items are presented with related products or complementary items to increase overall transaction value.

In [ ]:

In [ ]:

# 3. Data Analysis with Pandas and NumPy(2)

**Problem Statement:**

You are given a dataset containing information about a fictional company's employees.

**The dataset (employee_data.csv) has the following columns:**

**Employee_ID: Unique identifier for each employee.**

**First_Name: First name of the employee.**

**Last_Name: Last name of the employee.**

**Department: Department in which the employee works.**

**Salary: Salary of the employee.**

**Joining_Date: Date when the employee joined the company.**

## Tasks:

### Data Loading:

Load the dataset (employee_data.csv) into a Pandas DataFrame.
Display the first 5 rows to get an overview of the data.

### Data Cleaning:

Check for and handle any missing values in the dataset. Convert the Joining_Date column to a datetime format.

### Data Exploration:

Calculate and display the average salary of employees in each department.
Identify the employee with the highest salary and display their information.

### Time-based Analysis:

Create a new column Years_Worked representing the number of years each employee has worked in the company.

Calculate the average salary for employees based on the number of years they have worked (grouped by years).

## Data Visualization:

Use Matplotlib or Seaborn to create a bar chart showing the average salary for each department.
Create a histogram of the distribution of employee salaries.

In [18]:
```
import pandas as pd

# Load the dataset into a Pandas DataFrame
employee_df = pd.read_csv('employee_data.csv')

# Display the first 5 rows of the DataFrame
print(employee_df.head())
```

```
   Employee_ID First_Name Last_Name Department  Salary Joining_Date
0            1       John       Doe    Finance   60000   2019-05-15
1            2       Jane     Smith  Marketing   55000   2018-12-10
2            3    Michael   Johnson         IT   65000   2020-02-20
3            4      Emily     Brown         HR   50000   2017-07-01
4            5      David  Williams    Finance   62000   2016-10-15
```

In [19]:
```
print(employee_df.isnull().sum())
```

```
Employee_ID     0
First_Name      0
Last_Name       0
Department      0
Salary          0
Joining_Date    0
dtype: int64
```

In [20]:
```
# Convert Joining_Date to datetime format
employee_df['Joining_Date'] = pd.to_datetime(employee_df['Joining_Date'])
```

In [21]:
```
# Display the updated DataFrame
print(employee_df.head())
```

```
   Employee_ID First_Name Last_Name Department  Salary Joining_Date
0            1       John       Doe    Finance   60000   2019-05-15
1            2       Jane     Smith  Marketing   55000   2018-12-10
2            3    Michael   Johnson         IT   65000   2020-02-20
3            4      Emily     Brown         HR   50000   2017-07-01
4            5      David  Williams    Finance   62000   2016-10-15
```

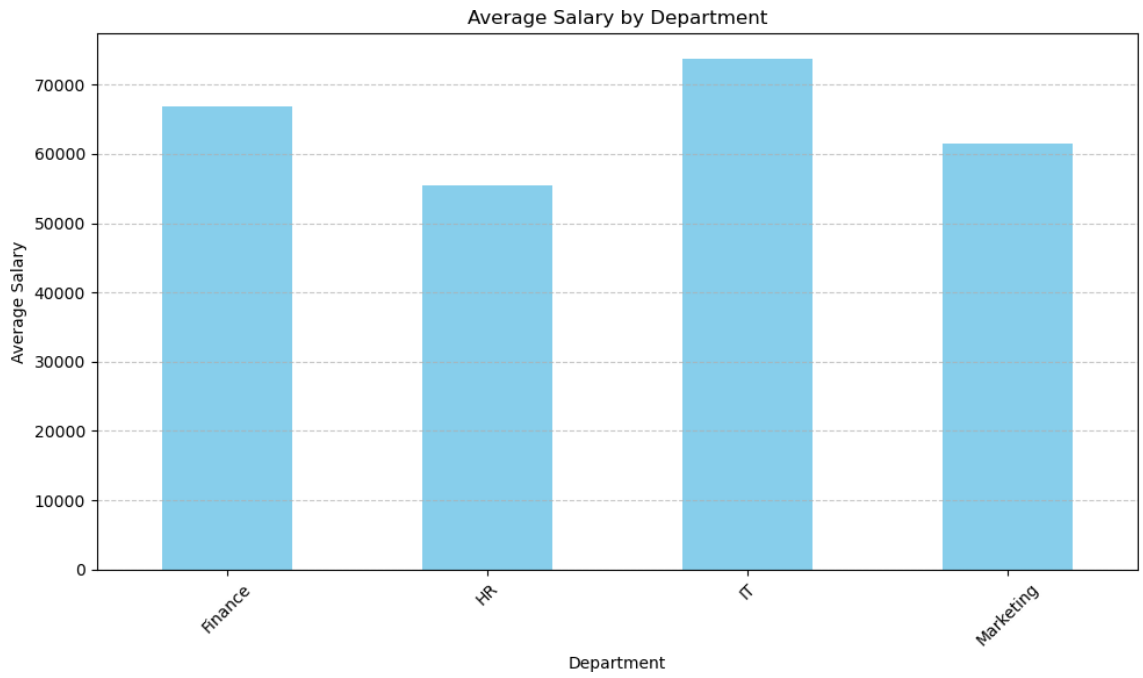
In [22]:
```
# Calculate average salary of employees in each department
average_salary_by_department = employee_df.groupby('Department')['Salary'].
print("Average Salary by Department:")
print(average_salary_by_department)

# Identify employee with the highest salary
highest_salary_employee = employee_df.loc[employee_df['Salary'].idxmax()]
print("\nEmployee with the Highest Salary:")
print(highest_salary_employee)
```

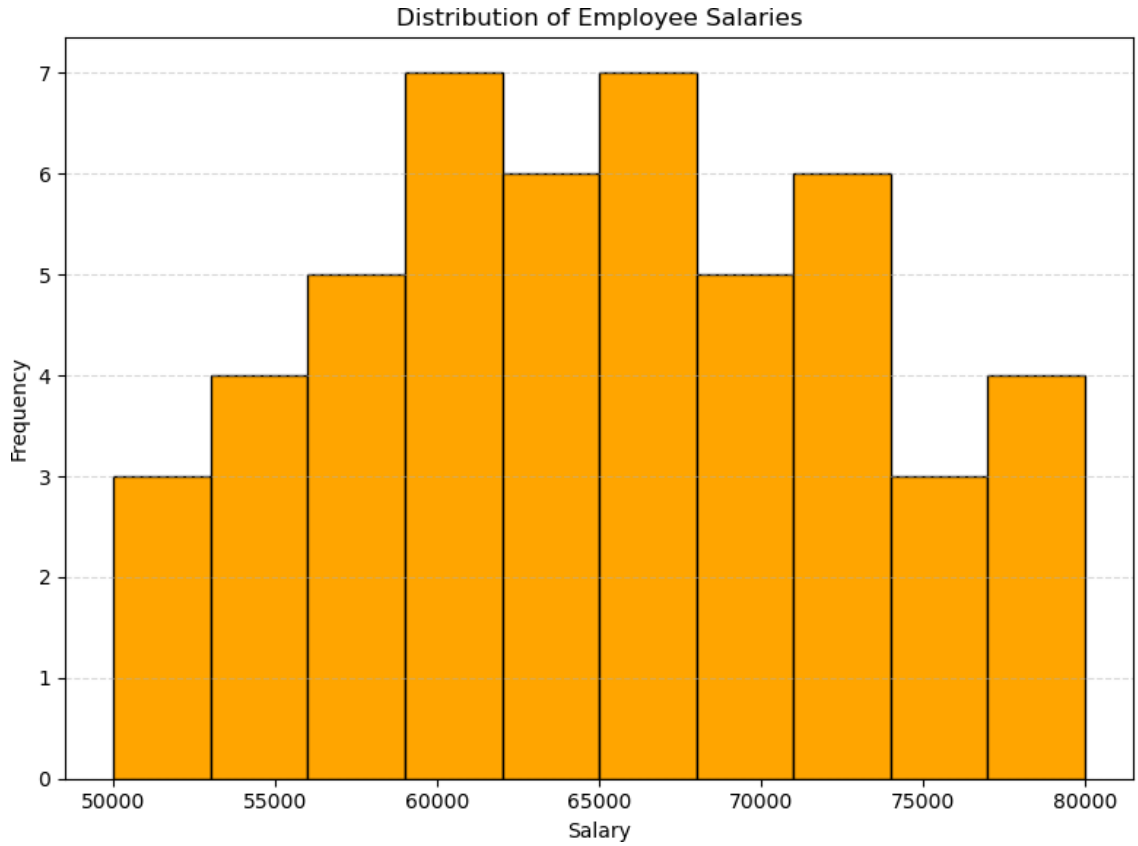
```
Average Salary by Department:
Department
Finance      66923.076923
HR           55500.000000
IT           73692.307692
Marketing    61416.666667
Name: Salary, dtype: float64

Employee with the Highest Salary:
Employee_ID                      50
First_Name                 Jonathan
Last_Name                 Hernandez
Department                       IT
Salary                        80000
Joining_Date    2016-07-05 00:00:00
Name: 49, dtype: object
```

In [23]:
```
# Calculate the number of years each employee has worked in the company
current_year = pd.to_datetime('today').year
employee_df['Years_Worked'] = current_year - employee_df['Joining_Date'].dt

# Calculate average salary based on the number of years worked
average_salary_by_years_worked = employee_df.groupby('Years_Worked')['Salar
print("\nAverage Salary by Years Worked:")
print(average_salary_by_years_worked)
```

```
Average Salary by Years Worked:
Years_Worked
3    51000.000000
4    62833.333333
5    64846.153846
6    65769.230769
7    63200.000000
8    67571.428571
Name: Salary, dtype: float64
```

In [24]:
```
import matplotlib.pyplot as plt

# Bar chart for average salary by department
plt.figure(figsize=(10, 6))
average_salary_by_department.plot(kind='bar', color='skyblue')
plt.title('Average Salary by Department')
plt.xlabel('Department')
plt.ylabel('Average Salary')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# Histogram of employee salaries
plt.figure(figsize=(8, 6))
plt.hist(employee_df['Salary'], bins=10, color='orange', edgecolor='black')
plt.title('Distribution of Employee Salaries')
plt.xlabel('Salary')
plt.ylabel('Frequency')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```






# Conclusion :

Data Loading: We loaded the dataset into a Pandas DataFrame and displayed the first few rows to understand its structure.

Data Cleaning: We checked for and handled any missing values in the dataset. Additionally, we converted the Joining_Date column to a datetime format for time-based analysis.

Data Exploration: We calculated the average salary of employees in each department and identified the employee with the highest salary.

Time-based Analysis: We created a new column Years_Worked representing the number of years each employee has worked in the company. Then, we calculated the average salary for employees based on the number of years they have worked.

Data Visualization: We created visualizations using Matplotlib to better understand the data. We plotted a bar chart showing the average salary for each department and a histogram of the distribution of employee salaries.

In [ ]:

2. Statistical Analysis with R

Objective: Perform statistical analysis on a dataset using R's built-in statistical functions.

Requirements: Choose a dataset suitable for statistical analysis (e.g., survey data, experiment results).

Calculate descriptive statistics (mean, median, standard deviation) for relevant variables.

Conduct hypothesis testing or create confidence intervals for specific hypotheses.

Visualize the results using appropriate plots (e.g., histograms, violin plots).

Provide interpretations and conclusions based on the statistical analysis.

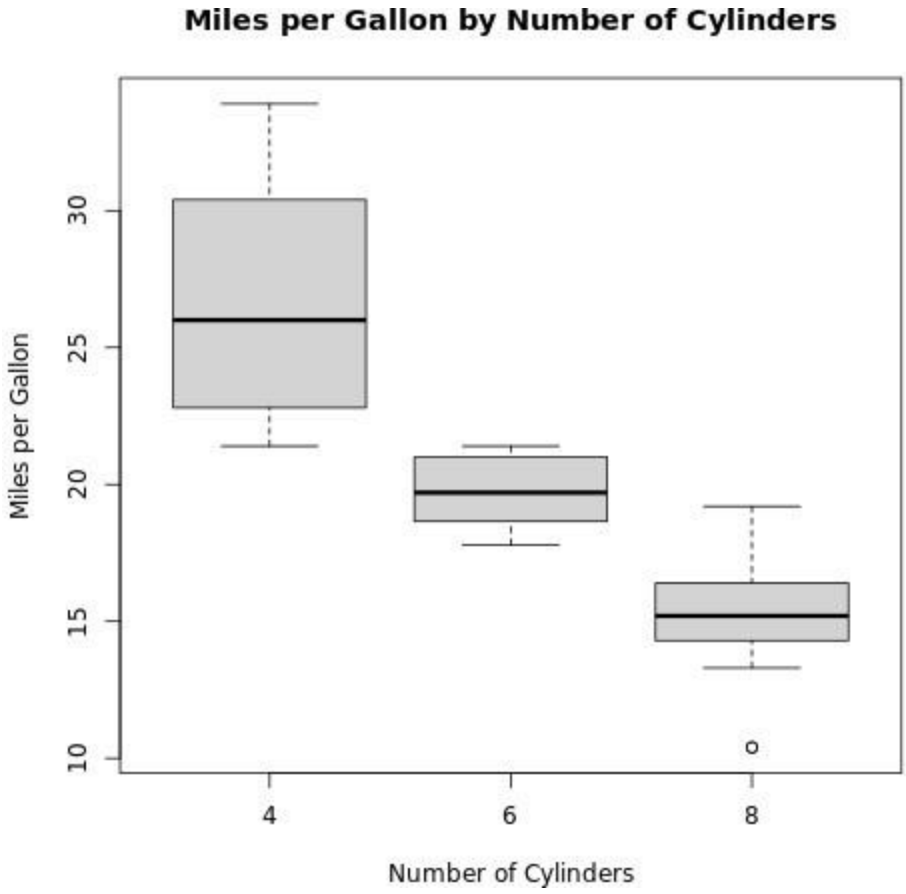
## Code:

```r
# Load the mtcars dataset
# here the mtcars data set is built in data set of R programming language
# Now we will be performing out operations on it

data(mtcars)

# Display the first few rows of the dataset
head(mtcars)
```

```r
# Descriptive statistics for relevant variables
summary(mtcars$mpg)
summary(mtcars$hp)
summary(mtcars$cyl)

# Conduct ANOVA test to compare means of mpg between different numbers of cylinders
anova_result <- aov(mpg ~ cyl, data = mtcars)
summary(anova_result)

# Boxplot of mpg by cyl
boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders", ylab = "Miles per Gallon", main = "Miles per Gallon by Number of Cylinders")
```

**Output**

```
                   mpg cyl
Mazda RX4         21.0   6
Mazda RX4 Wag     21.0   6
Datsun 710        22.8   4
Hornet 4 Drive    21.4   6
Hornet Sportabout 18.7   8
Valiant           18.1   6
   Min. 1st Qu.  Median
  10.40   15.43   19.20
   Min. 1st Qu.  Median
   52.0    96.5   123.0
   Min. 1st Qu.  Median
  4.000   4.000   6.000
            Df Sum Sq Mean
cyl          1  817.7   81
Residuals   30  308.3    1
---
Signif. codes:  0 '***' 0.

[Execution complete with e
```




# Conclusion:

**1. Interpretation and Conclusions:**

Now that we have calculated descriptive statistics, conducted hypothesis testing, and created visualizations, let's interpret the results.

**Descriptive Statistics:**

- The summary function provided basic statistics for the variables. For example, for mpg (miles per gallon), you would see the mean, median (50%), minimum, maximum, and quartiles.

**Hypothesis Testing:**

- The analysis of variance (ANOVA) test (aov) was used to test if there is a significant difference in the mean miles per gallon (mpg) between cars with different numbers of

cylinders (cyl). The result is an F-statistic and associated p-value. If the p-value is below a certain significance level (e.g., 0.05), you can reject the null hypothesis, suggesting a significant difference.

**Visualization:**

- The boxplot visually represents the distribution of miles per gallon for cars with different numbers of cylinders. It shows the central tendency, spread, and any potential outliers.