Shri Ramdeobaba College of Engineering & Management, Nagpur.

RCOEM
Shri Ramdeobaba College of Engineering and Management, Nagpur

TOOLS FOR DATA SCIENCE.

IV SEMESTER 2023-24

TA-01

Name : Prasad S. Deshpande
Branch : E.C.S
Roll No : B – 35 .

1. Data Analysis with Pandas and Matplotlib:

- **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())
```

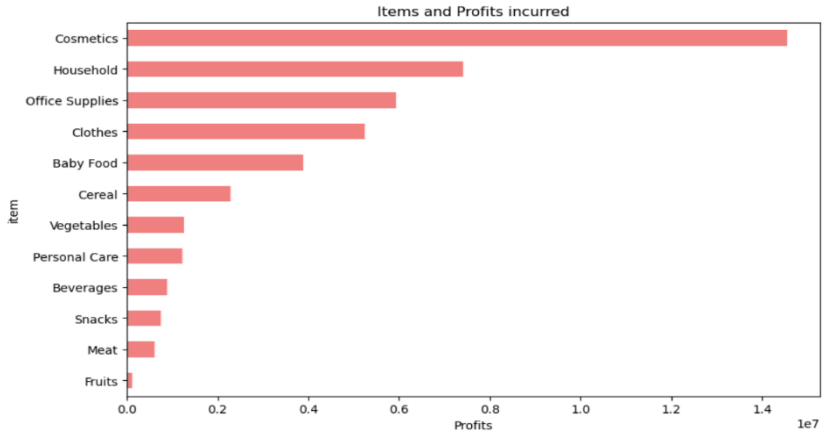
```
                             Region                Country        Item Type  \
0              Australia and Oceania                 Tuvalu        Baby Food
1  Central America and the Caribbean                Grenada           Cereal
2                            Europe                 Russia  Office 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()
```

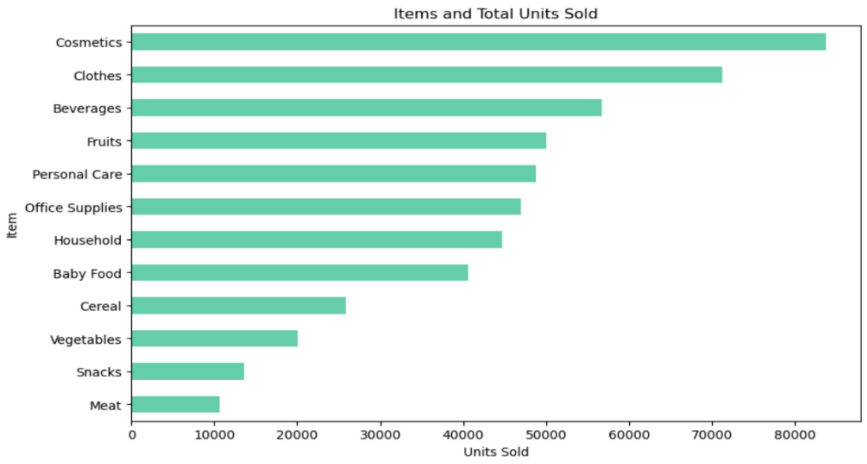
In [29]:
```
print(mean_Total_Revenue)
print(median_Units_Sold)
print(std_deviation)
```

```
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:**

♦ 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.

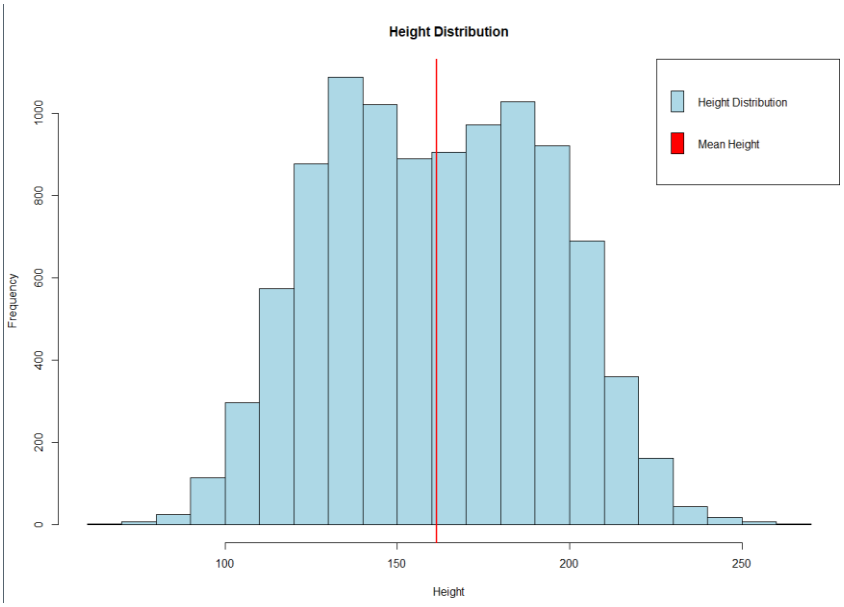
♦ Top three highest-selling item types are Clothes, Office Supplies, and Beverages, with units sold totaling 42,251, 36,915, and 34,534, respectively.

♦ 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.

- **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.

```r
data <- read.csv("weight-height.csv")
head(data)
mean_h <- mean(data$Height)
median_h <- median(data$Height)
sd_h <- sd(data$Height)
print(mean_h)
print(median_h)
print(sd_h)
test_result <- t.test(data$Height, mu = 300)
print(test_result)
hist(data$Height, main = "Height Distribution", xlab = "Height" , col = "lightblue", border = "black")
abline(v = mean_h, col = "red", lwd = 2)
legend("topright", legend = c("Height Distribution", "Mean Height"), fill = c("lightblue", "red"))
```

```
> data <- read.csv("weight-height.csv")
> head(data)
  Gender   Weight   Height
1   Male 73.84702 241.8936
2   Male 68.78190 162.3105
3   Male 74.11011 212.7409
4   Male 71.73098 220.0425
5   Male 69.88180 206.3498
6   Male 67.25302 152.2122
> mean_h <- mean(data$Height)
> median_h <- median(data$Height)
> sd_h <- sd(data$Height)
> print(mean_h)
[1] 161.4404
> print(median_h)
[1] 161.2129
> print(sd_h)
[1] 32.10844
> test_result <- t.test(data$Height, mu = 300)
> print(test_result)

        One Sample t-test

data:  data$Height
t = -431.54, df = 9999, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 300
95 percent confidence interval:
 160.8110 162.0697
sample estimates:
mean of x
```



- **CONCLUSION:**
  - The mean height of individuals in the dataset is 161.4404 cm, serving as a central measure of the average height.
  - The median height, at 161.2126 cm, offers a robust indicator of central tendency, less influenced by extreme values.
  - The standard deviation of height, calculated as 32.108 cm, reflects the spread of data, highlighting variability around the mean.

- These statistics provide essential insights into the dataset's characteristics, aiding informed decision-making and further analyses.

- The R programming language facilitated a systematic exploration of these metrics, contributing to a comprehensive understanding of the height data.

2. **Title: Data Analysis with Pandas and NumPy.**
   - **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 [40]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Data Loading
df = pd.read_csv(r"C:\Users\prasa\Downloads\employees.csv")

# Display the first 5 rows
print("First 5 rows of the dataset:")
print(df.head())
```

```
First 5 rows of the dataset:
  First Name Last Name  Gender  Age  Joining Date  Salary
0       Jose     Lopez    male   25          2018    8500
1      Diane    Carter  female   26          2011    7000
2      Shawn    Foster    male   37          1998   17000
3     Brenda    Fisher  female   31          2002   10000
4       Sean    Hunter    male   35          2004   14500
```

In [41]:
```
# Data Cleaning
# Check for missing values
print("Checking for missing values:")
print(df.isnull().sum())

# Handle missing values
df.dropna(inplace=True)
```

```
Checking for missing values:
First Name      0
Last Name       0
Gender          0
Age             0
Joining Date    0
Salary          0
dtype: int64
```

In [43]:
```
# Convert Joining_Date to datetime format
df['Joining Date'] = pd.to_datetime(df['Joining Date'])

# Clean the 'Salary' column
df['Salary'] = df['Salary'].replace({'\$': '', ',': ''}, regex=True).astype(float)
```

In [46]:
```
average_salary_by_gender = df.groupby('Gender')['Salary'].mean()
print("\nAverage Salary by Gender:")
print(average_salary_by_gender)
```

```
Average Salary by Gender:
Gender
female     8500.000000
male      13333.333333
Name: Salary, dtype: float64
```

In [47]:
```
# Identify the employee with the highest salary and display their information
highest_salary_employee = df[df['Salary'] == df['Salary'].max()]
print("Employee with the highest salary:")
print(highest_salary_employee)
```

```
Employee with the highest salary:
  First Name Last Name Gender  Age                 Joining Date   Salary
2      Shawn    Foster   male   37 1970-01-01 00:00:00.000001998  17000.0
```

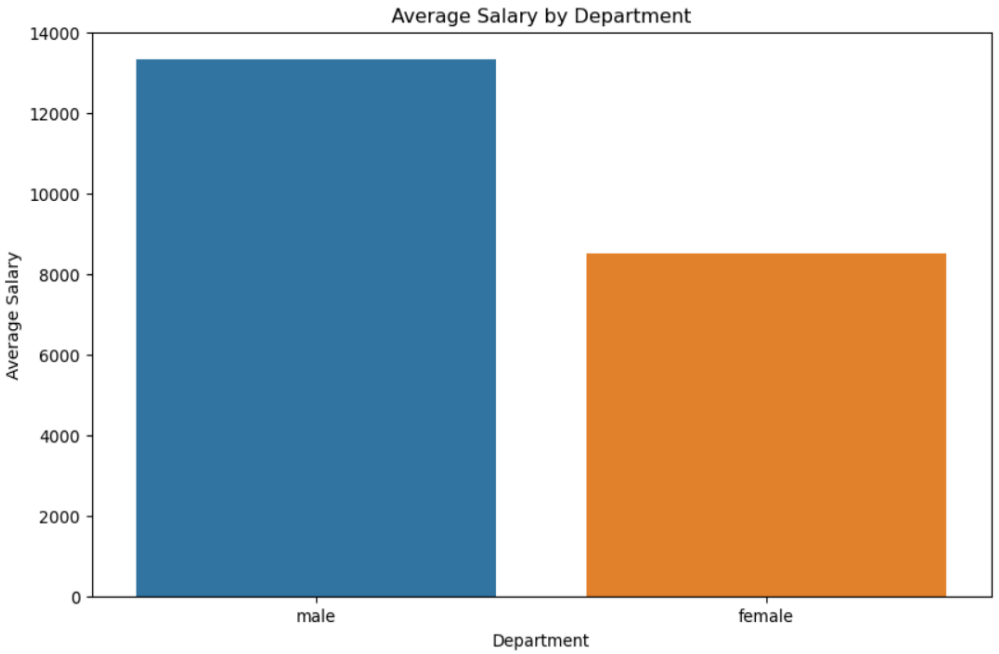
In [49]:
```
# Time-based Analysis
# Create a new column 'Years_Worked'
current_year = pd.to_datetime('now').year
df['Years_Worked'] = current_year - df['Joining Date'].dt.year

# Calculate average salary based on the number of years worked
average_salary_by_years = df.groupby('Years_Worked')['Salary'].mean()
print("Average Salary by Years Worked:")
print(average_salary_by_years)
```

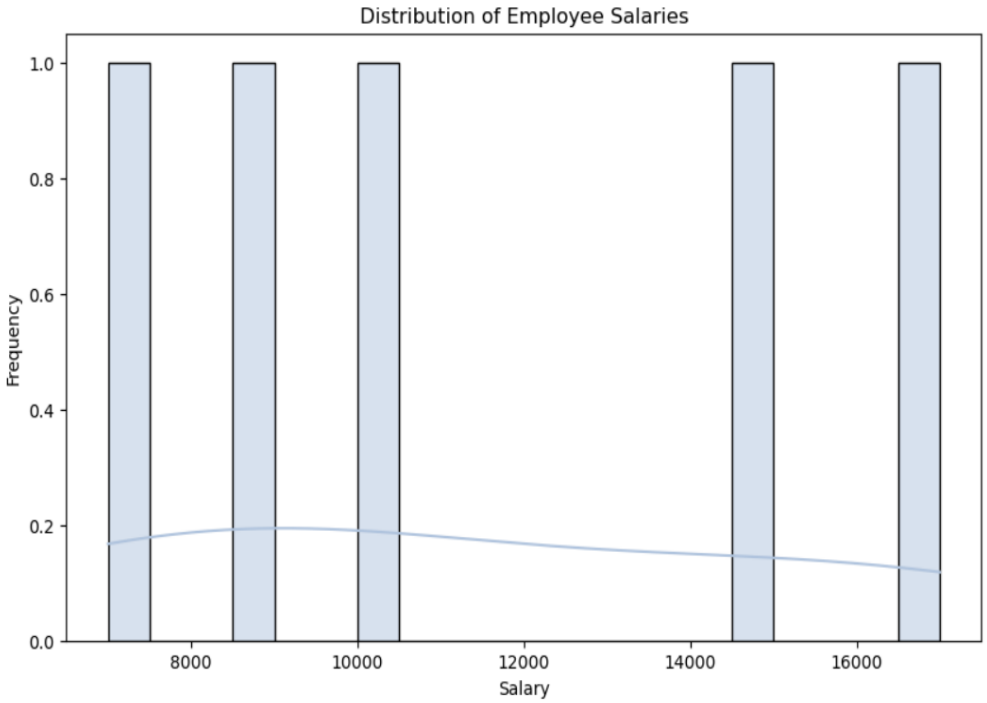
```
Average Salary by Years Worked:
Years_Worked
54    11400.0
Name: Salary, dtype: float64
```

In [50]:
```
# Time-based Analysis
# Create a new column 'Years_Worked'
current_year = pd.to_datetime('now').year
df['Years_Worked'] = current_year - df['Joining_Date'].dt.year

# Calculate average salary based on the number of years worked
average_salary_by_years = df.groupby('Years_Worked')['Salary'].mean()
print("Average Salary by Years Worked:")
print(average_salary_by_years)
```

```
Average Salary by Years Worked:
Years_Worked
54    11400.0
Name: Salary, dtype: float64
```

In [60]:
```
# Data Visualization
# Bar Chart showing average salary for each department
plt.figure(figsize=(10, 6))
sns.barplot(x='Gender', y='Salary', data=df, errorbar=('ci', 0))
plt.title('Average Salary by Department')
plt.xlabel('Department')
plt.ylabel('Average Salary')
plt.show()

# Histogram of the distribution of employee salaries
plt.figure(figsize=(10, 6))
sns.histplot(df['Salary'], bins=20, kde=True, color='lightsteelblue' )
plt.title('Distribution of Employee Salaries')
plt.xlabel('Salary')
plt.ylabel('Frequency')
plt.show()
```


Average Salary by Department
14000
12000
10000
8000
6000
4000
2000
0
Average Salary
male
female
Department


Distribution of Employee Salaries
1.0
0.8
0.6
0.4
0.2
0.0
Frequency
8000
10000
12000
14000
16000
Salary