Name : Devansh Dhopte

Branch : IT

Roll No : 29

Teachers Assesment - 1 of Tools for data Science

```
--prof Ashwini Gote
```

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.

```
import pandas as pd

df = pd.read_csv('Devansh_OE/data.csv')

print(df.head())  #print the few upper portion of data
```

```
   house_id  size_sqft  bedrooms  price_usd     location
0         1     1500.0       3.0   250000.0     New York
1         2     2000.0       4.0   320000.0  Los Angeles
2         3     1200.0       2.0   180000.0      Chicago
3         4     1800.0       3.0   280000.0      Houston
4         5     2500.0       4.0   400000.0      Phoenix
```

```
print(df)  ##print whole data
```

```
    house_id  size_sqft  bedrooms  price_usd          location
0          1     1500.0       3.0   250000.0          New York
1          2     2000.0       4.0   320000.0       Los Angeles
2          3     1200.0       2.0   180000.0           Chicago
3          4     1800.0       3.0   280000.0           Houston
4          5     2500.0       4.0   400000.0           Phoenix
5          6     1600.0       3.0   210000.0      Philadelphia
6          7     2200.0       4.0   330000.0       San Antonio
7          8     1900.0       3.0   290000.0         San Diego
8          9     2100.0       4.0   350000.0            Dallas
```

```
9          10     2300.0       4.0   380000.0          San Jose
10         11     1700.0       NaN   270000.0            Austin
11         12        NaN       3.0   240000.0      Jacksonville
12         13     2000.0       4.0        NaN   "San Francisco"
13         14     2100.0       3.0   310000.0          Columbus
14         15     2400.0       4.0   360000.0        Fort Worth
```

```
# Check for missing values
print(df.isnull().sum())
```

```
house_id     0
size_sqft    1
bedrooms     1
price_usd    1
location     0
dtype: int64
```

```
# Impute missing values with median
median_size = df['size_sqft'].median()
median_bedrooms = df['bedrooms'].median()
median_price = df['price_usd'].median()

df['size_sqft'].fillna(median_size, inplace=True)
df['bedrooms'].fillna(median_bedrooms, inplace=True)
df['price_usd'].fillna(median_price, inplace=True)

# Verify if missing values are handled
print(df.isnull().sum())
```

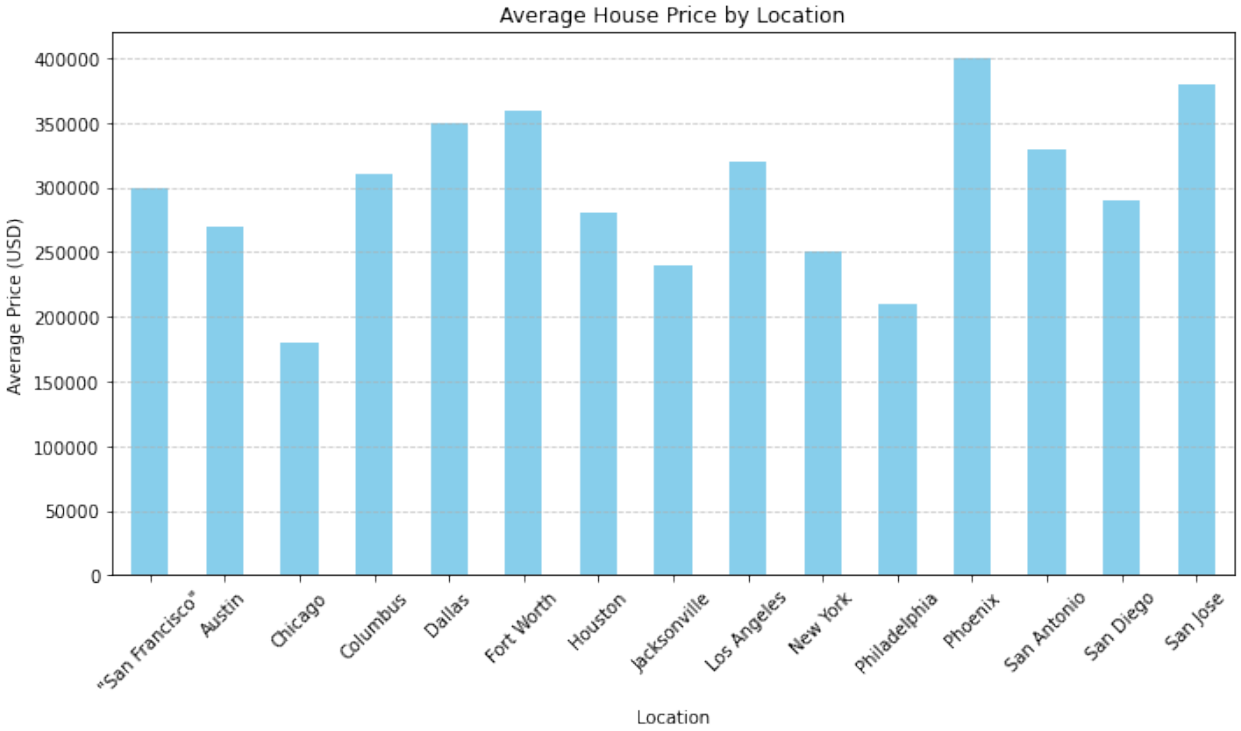
```
house_id     0
size_sqft    0
bedrooms     0
price_usd    0
location     0
dtype: int64
```

```
# Perform basic statistical analysis
mean_size = df['size_sqft'].mean()
median_size = df['size_sqft'].median()
std_dev_size = df['size_sqft'].std()

mean_bedrooms = df['bedrooms'].mean()
median_bedrooms = df['bedrooms'].median()
std_dev_bedrooms = df['bedrooms'].std()

mean_price = df['price_usd'].mean()
median_price = df['price_usd'].median()
std_dev_price = df['price_usd'].std()

# Print the results
print("Size_sqft:")
```

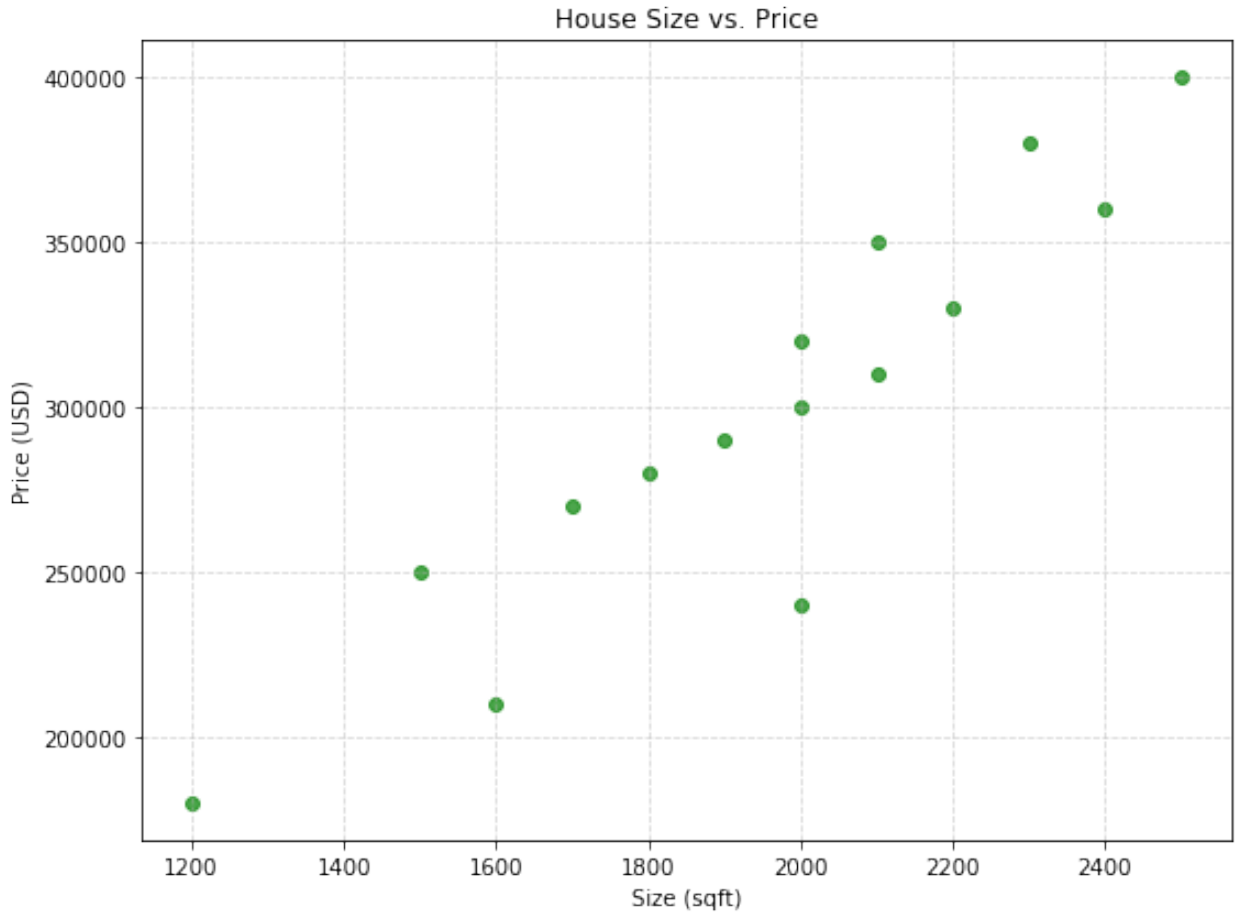
```
print("Mean:", mean_size)
print("Median:", median_size)
print("Standard Deviation:", std_dev_size)
print("\nBedrooms:")
print("Mean:", mean_bedrooms)
print("Median:", median_bedrooms)
print("Standard Deviation:", std_dev_bedrooms)
print("\nPrice_usd:")
print("Mean:", mean_price)
print("Median:", median_price)
print("Standard Deviation:", std_dev_price)
```

```
Size_sqft:
Mean: 1953.3333333333333
Median: 2000.0
Standard Deviation: 350.2380143083653

Bedrooms:
Mean: 3.433333333333333
Median: 3.5
Standard Deviation: 0.622972903178973

Price_usd:
Mean: 298000.0
Median: 300000.0
Standard Deviation: 62013.82334378591
```

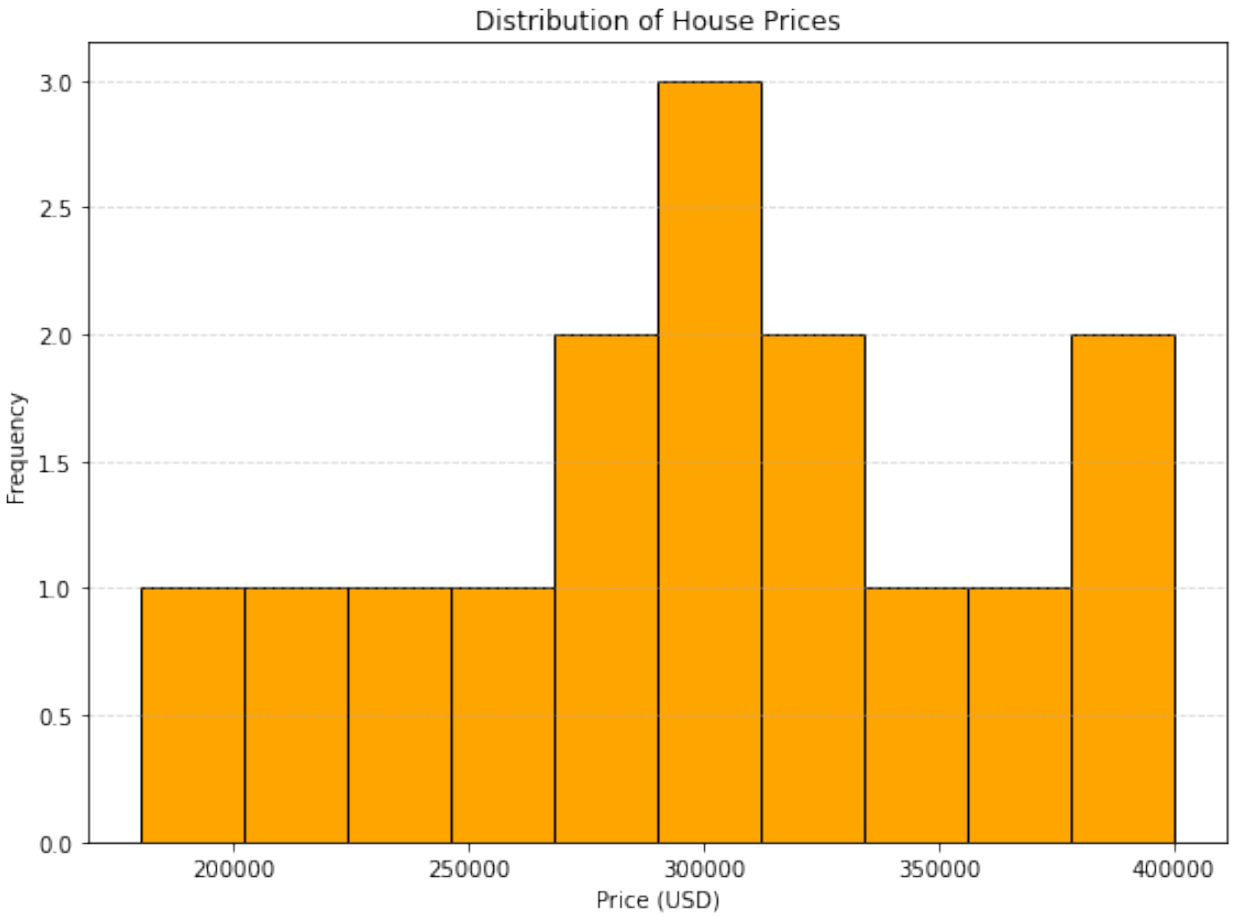
```
import matplotlib.pyplot as plt

# Group the data by location and calculate the mean price for each
location
mean_price_by_location = df.groupby('location')['price_usd'].mean()

# Plot the bar chart
plt.figure(figsize=(10, 6))
mean_price_by_location.plot(kind='bar', color='skyblue')
plt.title('Average House Price by Location')
plt.xlabel('Location')
plt.ylabel('Average Price (USD)')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
# Plot scatter plot for size_sqft vs. price_usd
plt.figure(figsize=(8, 6))
plt.scatter(df['size_sqft'], df['price_usd'], color='green',
alpha=0.7)
plt.title('House Size vs. Price')
plt.xlabel('Size (sqft)')
plt.ylabel('Price (USD)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```




```
# Plot histogram for house prices
plt.figure(figsize=(8, 6))
plt.hist(df['price_usd'], bins=10, color='orange', edgecolor='black')
plt.title('Distribution of House Prices')
plt.xlabel('Price (USD)')
plt.ylabel('Frequency')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



Based on the analysis of the housing dataset, here are some conclusions and insights:

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:

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.

```
import pandas as pd

# Load the dataset into a Pandas DataFrame
employee_df = pd.read_csv('Devansh_OE/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
```

```
print(employee_df.isnull().sum())
```

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

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

# 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
```

```
# Calculate average salary of employees in each department
average_salary_by_department = employee_df.groupby('Department')
['Salary'].mean()
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)
```

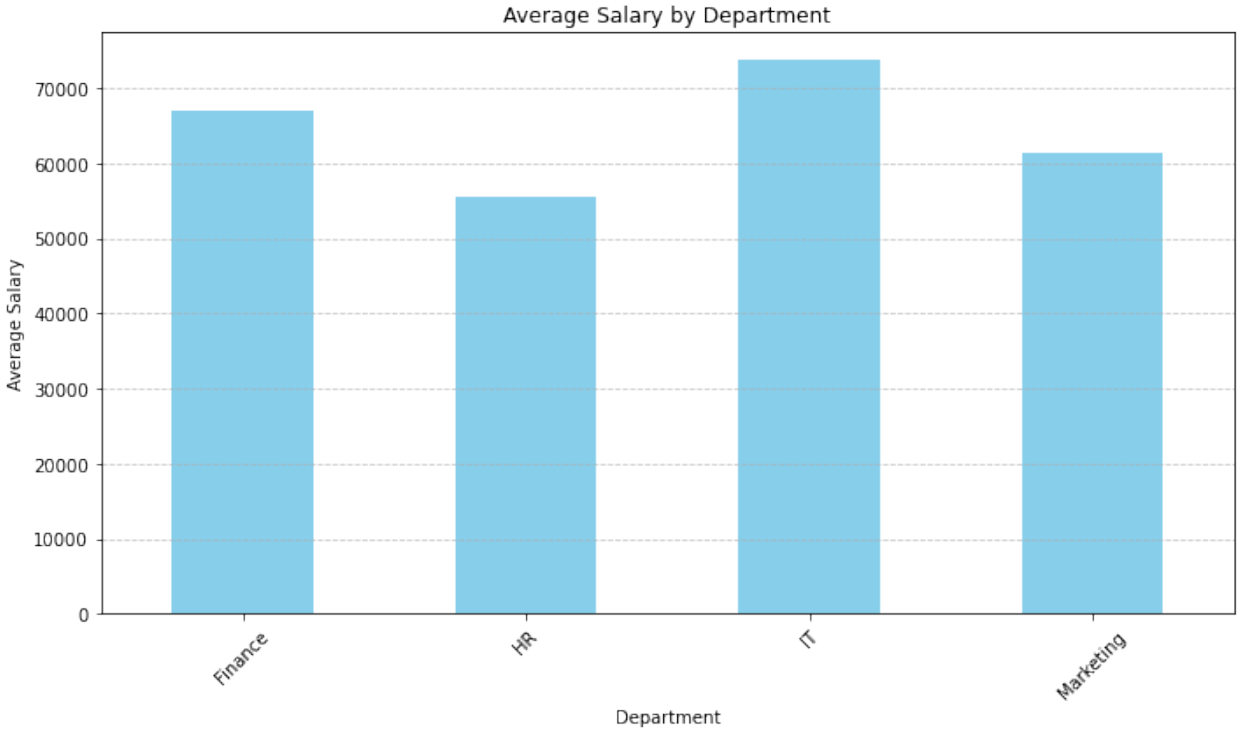
```
Average Salary by Department:
Department
Finance      66923.076923
HR           55500.000000
IT           73692.307692
Marketing    61416.666667
Name: Salary, dtype: float64

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
```

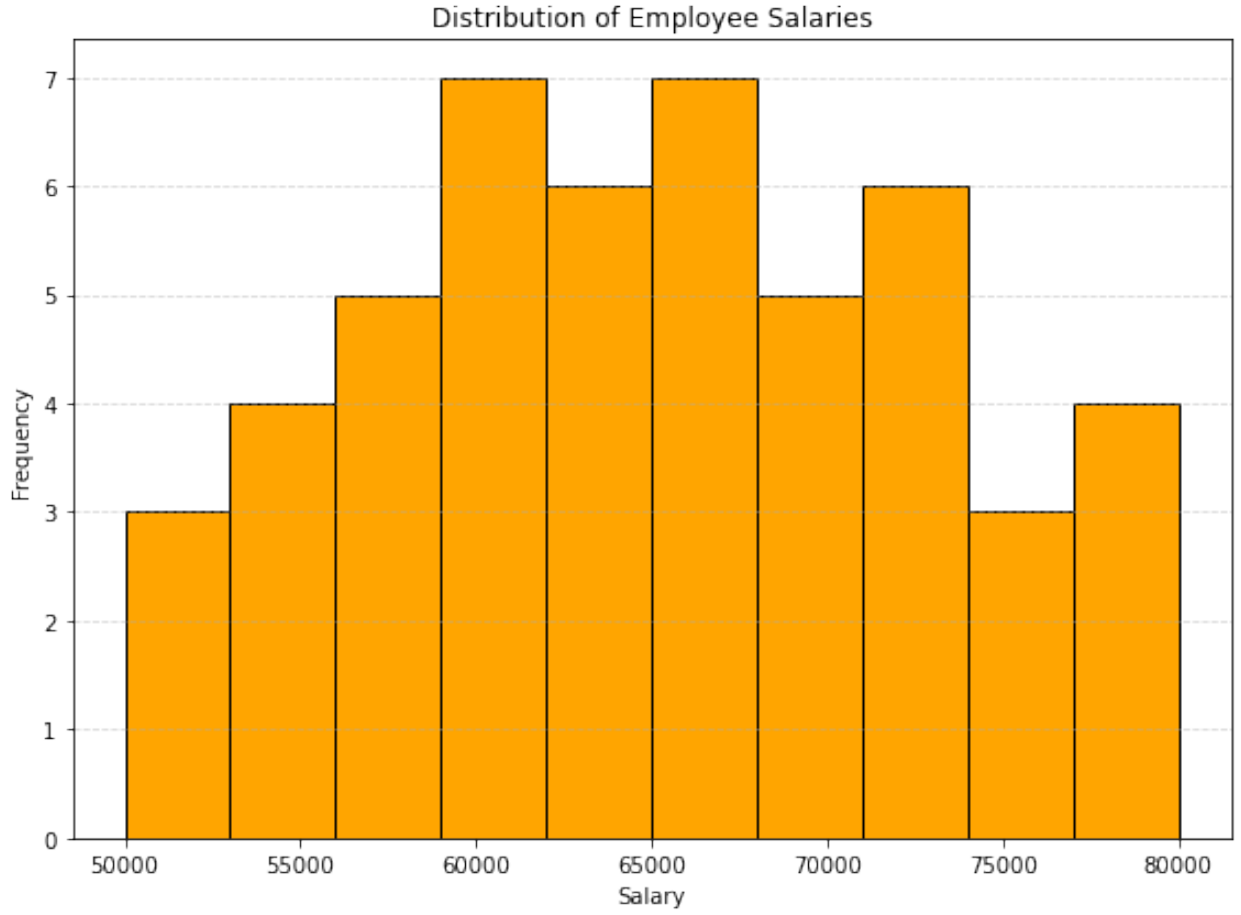
```
# Calculate the number of years each employee has worked in the
company
current_year = pd.to_datetime('today').year
employee_df['Years_Worked'] = current_year -
employee_df['Joining_Date'].dt.year

# Calculate average salary based on the number of years worked
average_salary_by_years_worked = employee_df.groupby('Years_Worked')
```

```
['Salary'].mean()
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
```

```
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()
```


Average Salary by Department
Average Salary
0
10000
20000
30000
40000
50000
60000
70000
Finance
HR
IT
Marketing
Department


Distribution of Employee Salaries
Frequency
0
1
2
3
4
5
6
7
50000
55000
60000
65000
70000
75000
80000
Salary

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