w7terwftz

March 6, 2024

0.1 Name : Abhinav Anpan

0.2 Branch : IT

0.3 Roll No : 16

0.4 Teachers Assesment - 1 of Tools for data Science

--pr

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.

[24]:
```
import pandas as pd
```

[25]:
```
df = pd.read_csv('Abhinav_OE_/data.csv')
```

[26]:
```
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
```

[27]:
```
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
```

[28]:
```python
# Check for missing values
print(df.isnull().sum())
```

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

[29]:
```python
# 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
```

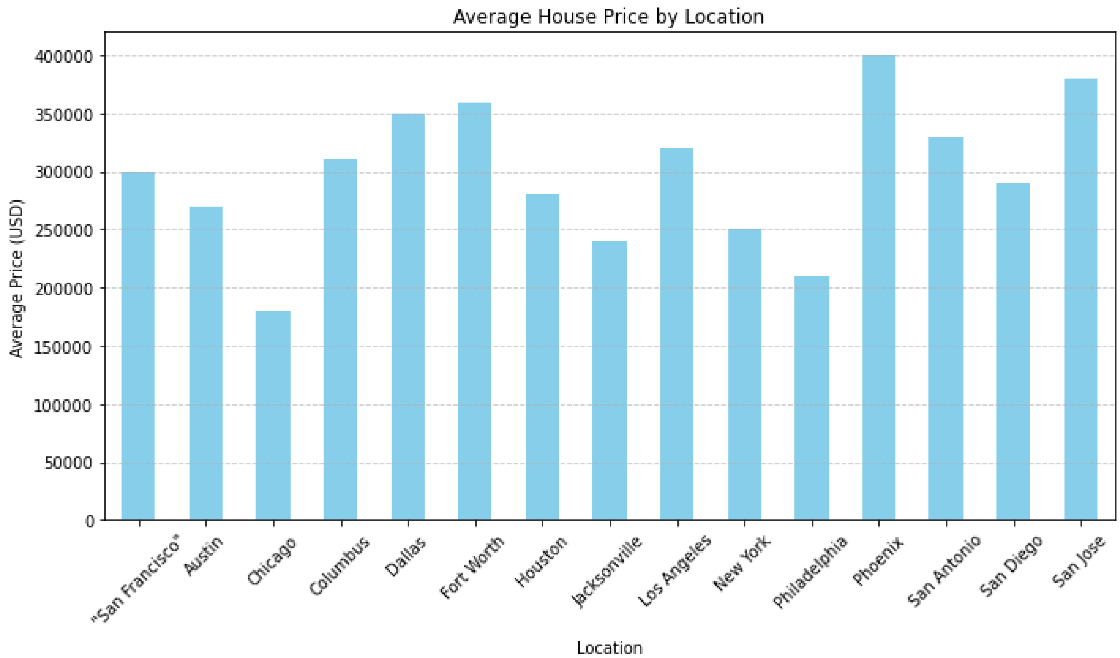
[30]:
```python
# Perform basic statistical analysis
mean_size = df['size_sqft'].mean()
median_size = df['size_sqft'].median()
std_dev_size = df['size_sqft'].std()

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)
```

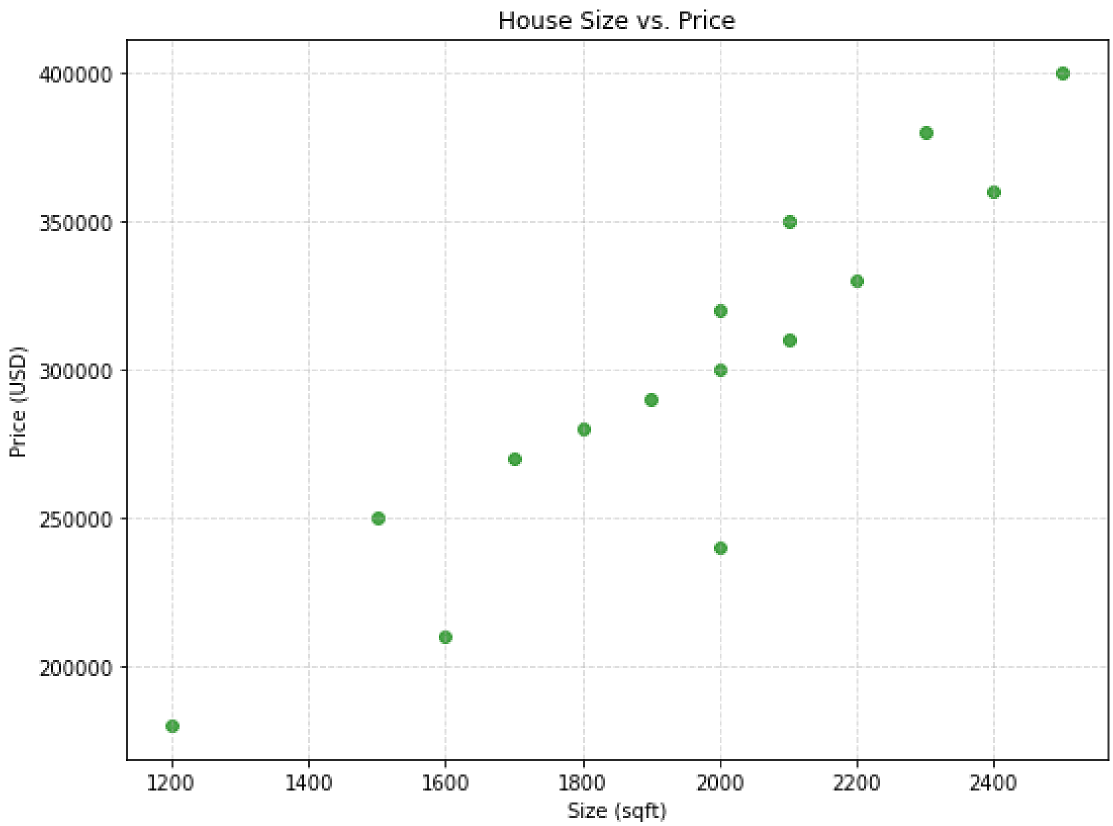
```
Size_sqft:
Mean: 1953.3333333333333
Median: 2000.0
Standard Deviation: 350.2380143083653

Bedrooms:
Mean: 3.4333333333333333
Median: 3.5
Standard Deviation: 0.622972903178973

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

[31]:
```
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()
```

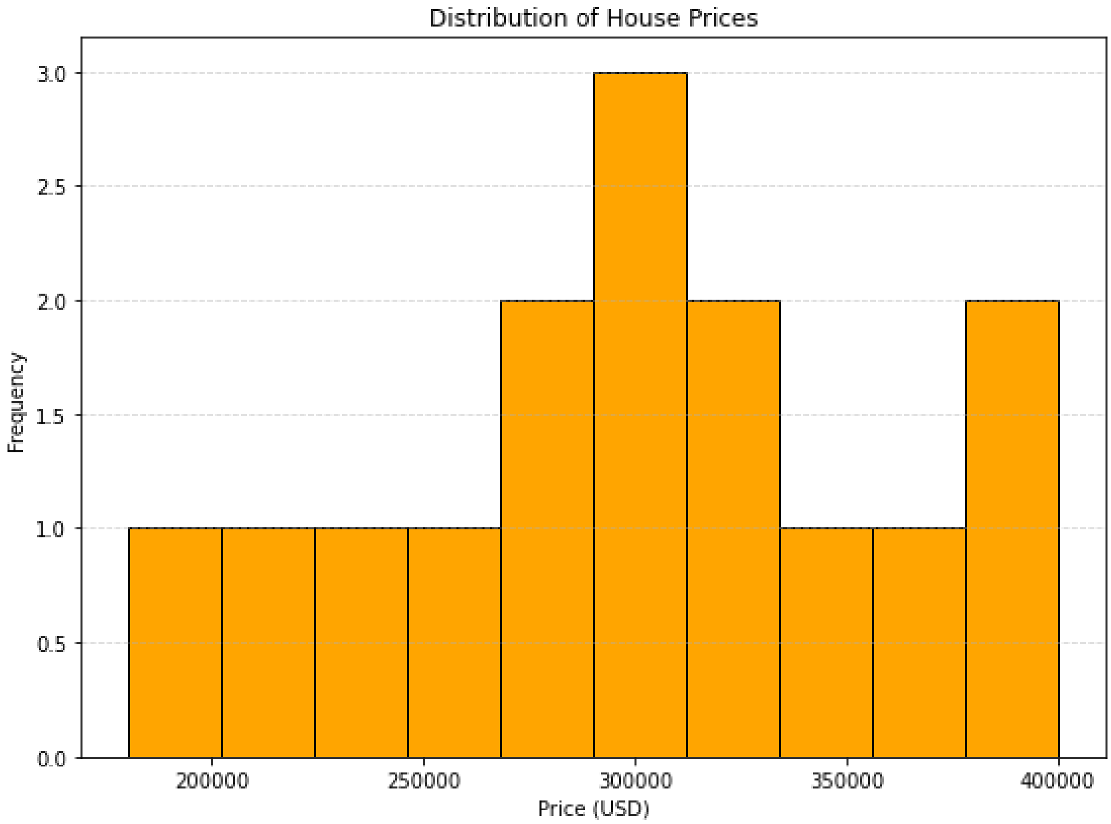


[32]:
```
# Plot scatter plot for size_sqft vs. price_usd
plt.figure(figsize=(8, 6))
plt.scatter(df['size_sqft'], df['price_usd'], color='green', alpha=0.7)
plt.title('House Size vs. Price')
plt.xlabel('Size (sqft)')
plt.ylabel('Price (USD)')
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



[33]:

```
# 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()
```



Conclusion :

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

1.Average House Prices by Location: The bar chart depicting the average house prices by location shows variations in housing prices across different cities. For example, San Francisco and Los Angeles have relatively higher average prices compared to other cities in the dataset.

2.Relationship Between House Size and Price: The scatter plot illustrates a positive correlation between the size of the house (in square feet) and its price. Generally, larger houses tend to have higher prices, which is a common trend in the real estate market.

3.Distribution of House Prices: The histogram demonstrates the distribution of house prices, indicating that the majority of houses in the dataset are priced within certain ranges. However, there are also some higher-priced houses, as evidenced by the tail of the distribution.

4.Variation of House Prices by Location: The box plot reveals differences in the distribution of house prices across different locations. Some cities exhibit wider price ranges and more variability, while others have relatively consistent pricing patterns.

Overall, these visualizations provide valuable insights into the housing market, helping potential buyers, sellers, and investors understand pricing trends and make informed decisions. Additionally, further analysis could be conducted to explore other factors influencing house prices, such as the number of bedrooms, proximity to amenities, and economic indicators specific to each location.

[ ]:

```

```

[ ]:

```

```

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.

0.5 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.

[34]:

```python
import pandas as pd

# Load the dataset into a Pandas DataFrame
```

```
employee_df = pd.read_csv('Abhinav_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
```

[35]:
```
print(employee_df.isnull().sum())
```

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

[36]:
```
# Convert Joining_Date to datetime format
employee_df['Joining_Date'] = pd.to_datetime(employee_df['Joining_Date'])
```

[37]:
```
# 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
```

[38]:
```
# 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
```

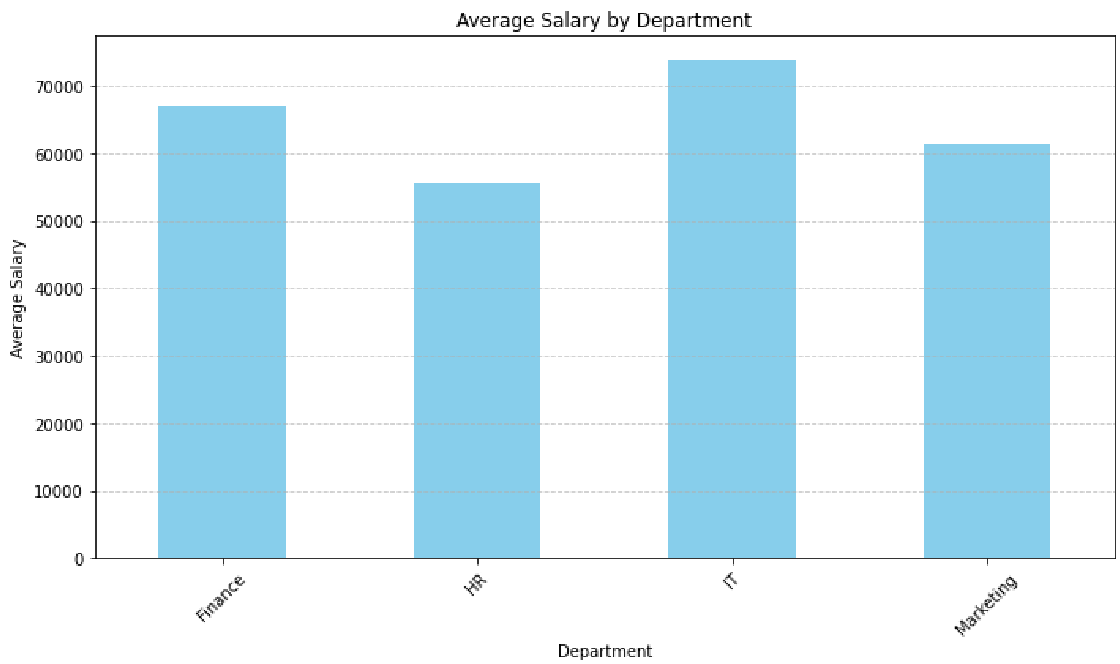
```
Finance      66923.076923
HR           55500.000000
IT           73692.307692
Marketing    61416.666667
Name: Salary, dtype: float64

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
```

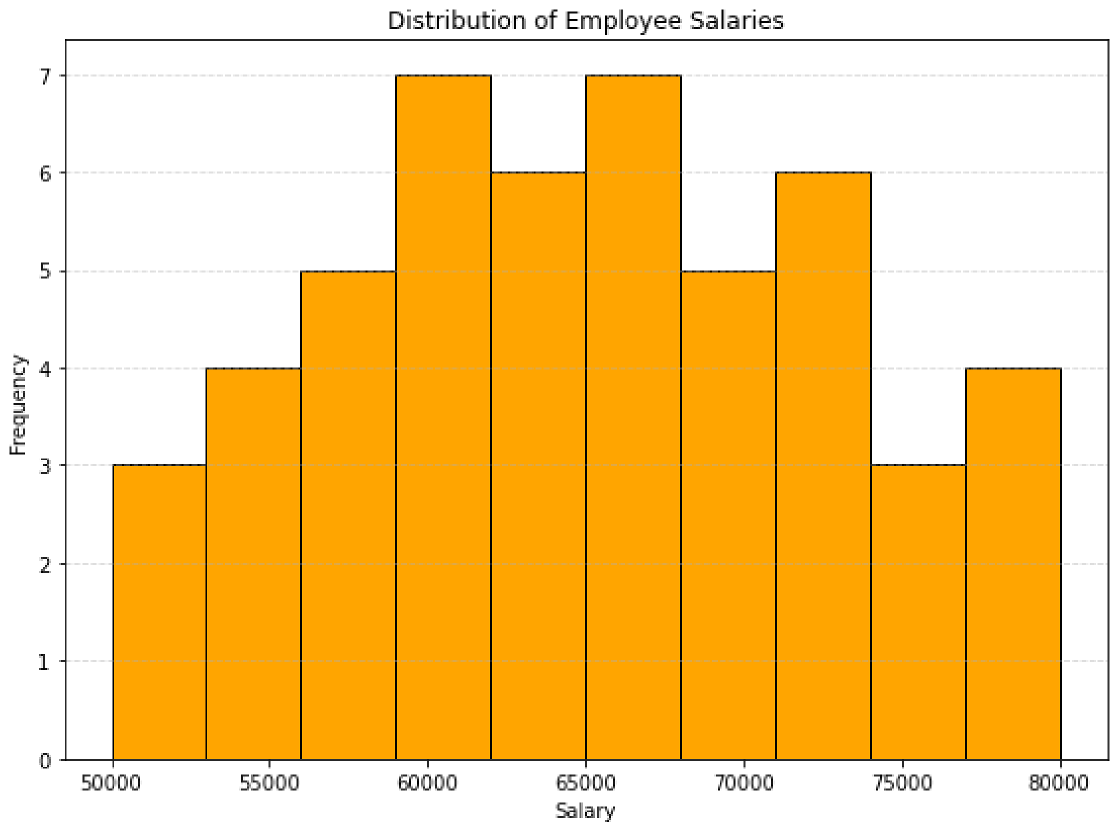
[39]:
```
# Calculate the number of years each employee has worked in the company
current_year = pd.to_datetime('today').year
employee_df['Years_Worked'] = current_year - employee_df['Joining_Date'].dt.year

# 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
```

[40]:
```
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()
```

```python
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.

[ ]:

[ ]:

[ ]:

```

```

[ ]:

```

```

[ ]:

```

```

Name: Abhinav Anpan

Branch: IT

Roll No: 16

Tools for data science

Teacher Assessment

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:-

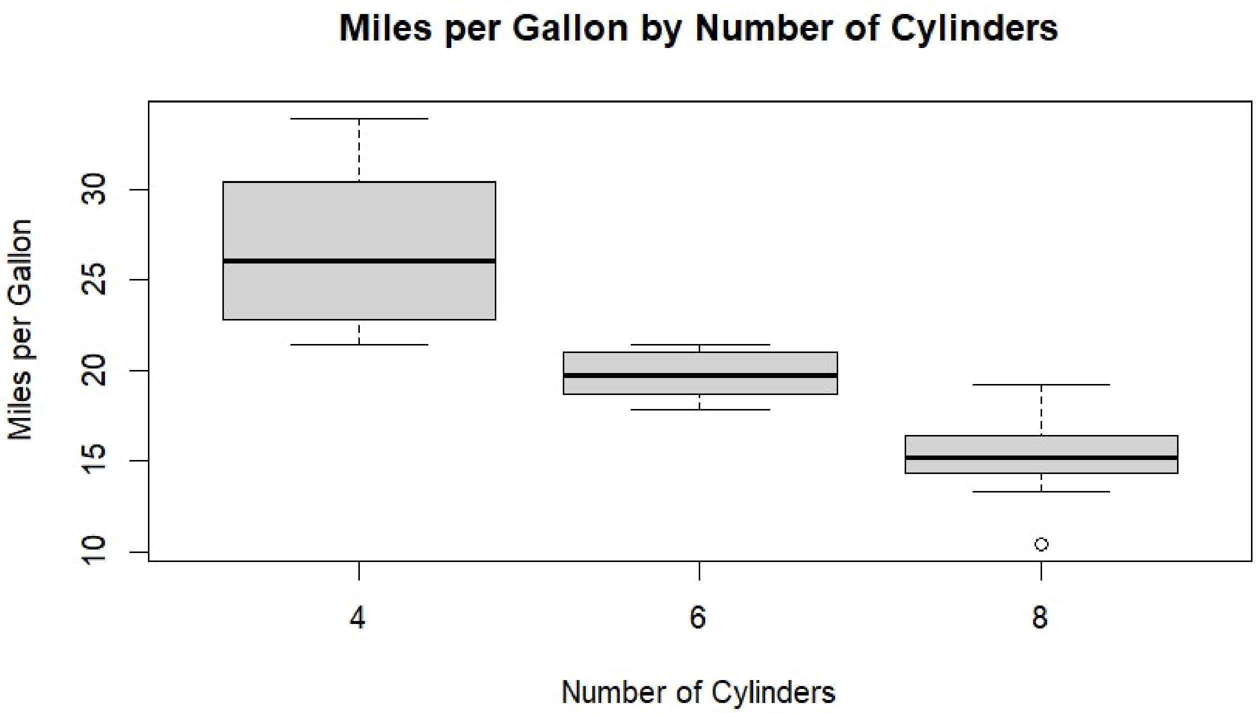
```
# Load the mtcars dataset
# here the mtcars data set is built in data set of R programming language
# Now we will be performing out operations on it

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)
```

```r
# 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:-

```
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> data(mtcars)
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
> summary(mtcars$mpg)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  10.40   15.43   19.20   20.09   22.80   33.90
> summary(mtcars$hp)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   52.0    96.5   123.0   146.7   180.0   335.0
> summary(mtcars$cyl)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  4.000   4.000   6.000   6.188   8.000   8.000
> # Conduct ANOVA test to compare means of mpg between different numbers of cylinders
> anova_result <- aov(mpg ~ cyl, data = mtcars)
> summary(anova_result)
            Df Sum Sq Mean Sq F value   Pr(>F)
cyl          1  817.7   817.7   79.56 6.11e-10 ***
Residuals   30  308.3    10.3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```



CONCLUSION :->

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.