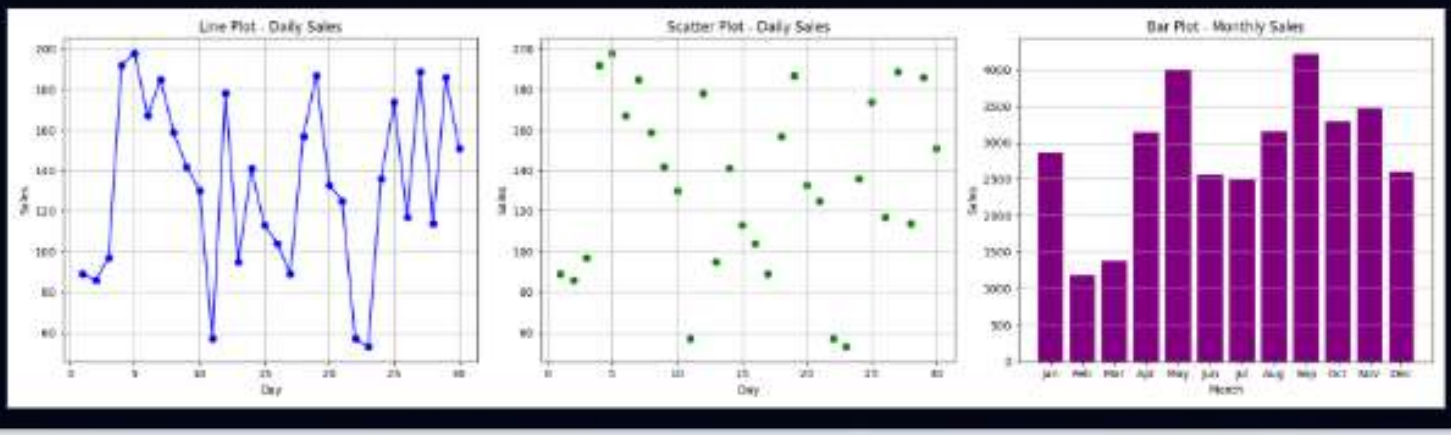
You have unsaved changes. To share your code, click the "Fork" button to generate a permanent link.

Python

```python
import matplotlib.pyplot as plt
import numpy as np

# Example data for 30 days (daily sales)
days = np.arange(1, 31)
daily_sales = np.random.randint(50, 200, size=30)  # Replace with real daily data

# Example data for 12 months (monthly sales)
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
          'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
monthly_sales = np.random.randint(1000, 5000, size=12)  # Replace with real monthly data

# Create a figure with subplots
plt.figure(figsize=(18, 5))

# Line Plot
plt.subplot(1, 3, 1)
plt.plot(days, daily_sales, marker='o', linestyle='-', color='blue')
plt.title("Line Plot - Daily Sales")
plt.xlabel("Day")
plt.ylabel("Sales")
plt.grid(True)

# Scatter Plot
plt.subplot(1, 3, 2)
plt.scatter(days, daily_sales, color='green')
plt.title("Scatter Plot - Daily Sales")
plt.xlabel("Day")
plt.ylabel("Sales")
plt.grid(True)

# Bar Plot
plt.subplot(1, 3, 3)
plt.bar(months, monthly_sales, color='purple')
plt.title("Bar Plot - Monthly Sales")
plt.xlabel("Month")
plt.ylabel("Sales")
plt.grid(axis='y')

# Layout and show all plots
plt.tight_layout()
plt.show()
```

Click Run or press shift + ENTER to run code

Enable code completions

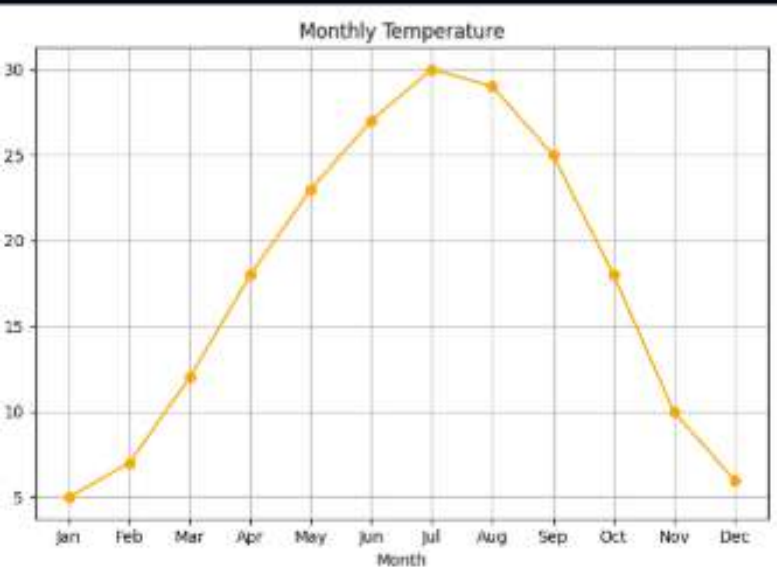


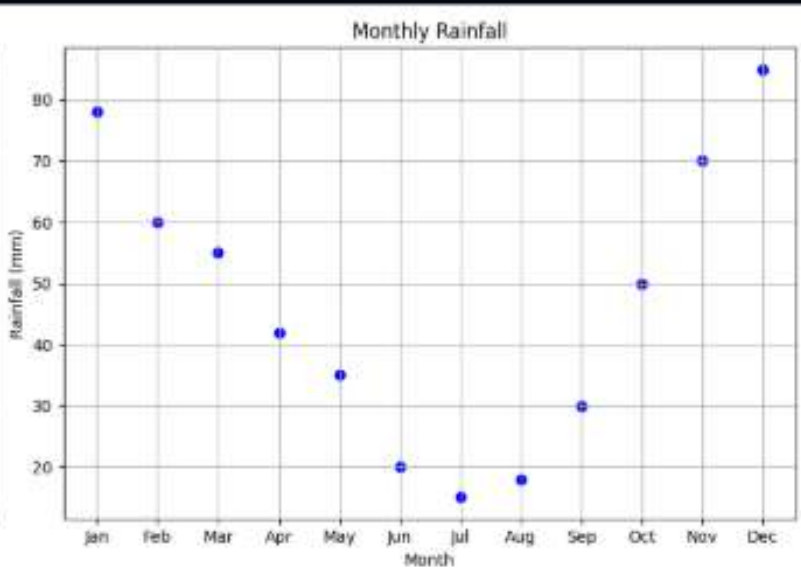
Python

```python
import matplotlib.pyplot as plt
import numpy as np

# Sample data for 12 months
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
          'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

# Example temperature (°C) and rainfall (mm) data
temperature = [5, 7, 12, 18, 23, 27, 30, 29, 25, 18, 10, 6]     # Replace with actual tem
rainfall = [78, 60, 55, 42, 35, 20, 15, 18, 30, 50, 70, 85]     # Replace with actual rai

# Create figure and subplots
plt.figure(figsize=(14, 5))

# 1. Line Plot - Monthly Temperature
plt.subplot(1, 2, 1)
plt.plot(months, temperature, marker='o', color='orange', linestyle='-')
plt.title("Monthly Temperature")
plt.xlabel("Month")
plt.ylabel("Temperature (°C)")
plt.grid(True)

# 2. Scatter Plot - Monthly Rainfall
plt.subplot(1, 2, 2)
plt.scatter(months, rainfall, color='blue')
plt.title("Monthly Rainfall")
plt.xlabel("Month")
plt.ylabel("Rainfall (mm)")
plt.grid(True)

# Layout adjustment and display
plt.tight_layout()
plt.show()
```

ck Run or press shift + ENTER to run code

Enable code completions





main.py

Share | Run

```python
import string
from collections import Counter

try:
    # Step 1: Read the text file safely
    with open("sample_text.txt", "r", encoding="utf-8", errors='ignore') as
        file:
        text = file.read()

    # Step 2: Preprocess the text
    text = text.lower()  # convert to lowercase
    text = text.translate(str.maketrans('', '', string.punctuation))  # remove
        punctuation
    words = text.split()  # split into words

    # Step 3: Count word frequencies
    word_freq = Counter(words)

    # Step 4: Display the frequency distribution
    print("Word Frequency Distribution:\n")
    for word, freq in word_freq.most_common():
        print(f"{word}: {freq}")

except FileNotFoundError:
    print("Error: The file 'sample_text.txt' was not found. Please check the
        filename and location.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

Output | Clear

```
ERROR!
Error: The file 'sample_text.txt' was not found. Please check the filename and
    location.

=== Code Execution Successful ===
```

main.py

Share Run

```python
import pandas as pd

# Sample data - replace this with your actual DataFrame
data = {
    'customer_id': [101, 102, 103, 104, 105, 106, 107],
    'age': [25, 30, 25, 40, 30, 25, 35],
    'purchase_amount': [100, 150, 80, 120, 90, 60, 200]
}

# Create DataFrame
df = pd.DataFrame(data)

# Frequency distribution of customer ages
age_distribution = df['age'].value_counts().sort_index()

# Display results
print("Frequency Distribution of Customer Ages:\n")
print(age_distribution)

```

Output Clear

```
Frequency Distribution of Customer Ages:

age
25    3
30    2
35    1
40    1
Name: count, dtype: int64

=== Code Execution Successful ===
```

Programiz
Python Online Compiler

main.py

Share Run

```python
import pandas as pd

# Sample data - replace this with your actual dataset
data = {
    'post_id': [1, 2, 3, 4, 5, 6, 7, 8],
    'likes': [10, 20, 10, 30, 20, 10, 40, 30]
}

# Create DataFrame
df = pd.DataFrame(data)

# Frequency distribution of likes
likes_distribution = df['likes'].value_counts().sort_index()

# Display results
print("Frequency Distribution of Likes:\n")
print(likes_distribution)
```

Output Clear

```
Frequency Distribution of Likes:

likes
10    3
20    2
30    2
40    1
Name: count, dtype: int64

=== Code Execution Successful ===
```

main.py

Share | Run

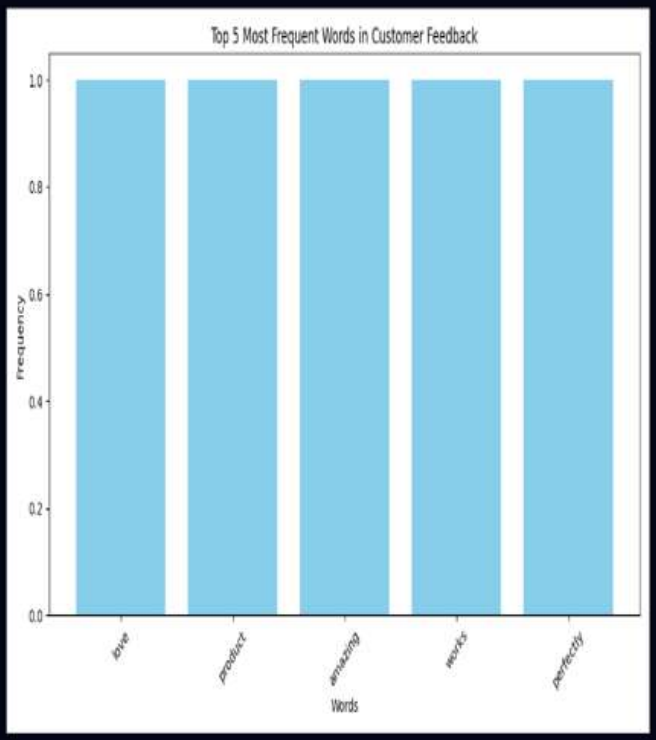
```python
import pandas as pd
import string
from collections import Counter

# Sample data - replace this with your actual dataset
data = {
    'review_id': [1, 2, 3, 4, 5],
    'review': [
        'This product is amazing! I love it.',
        'The product is good, but it could be better.',
        'I did not like the product, it is not worth the price.',
        'Amazing quality, highly recommend it to others!',
        'The product is okay, but not great.'
    ]
}

# Create DataFrame
df = pd.DataFrame(data)

# Step 1: Preprocess the reviews (convert to lowercase, remove punctuation)
def preprocess_review(review):
    review = review.lower()  # Convert to lowercase
    review = review.translate(str.maketrans('', '', string.punctuation))  # Remove punctuation
    return review

# Apply preprocessing to all reviews
df['processed_review'] = df['review'].apply(preprocess_review)

# Step 2: Tokenize reviews into words
words = ' '.join(df['processed_review']).split()

# Step 3: Count word frequencies
word_freq = Counter(words)

# Step 4: Display the results
print("Word Frequency Distribution:\n")
for word, freq in word_freq.most_common():
    print(f"{word}: {freq}")
```

Output | Clear

```
Word Frequency Distribution:

product: 4
is: 4
it: 4
the: 4
not: 3
amazing: 2
i: 2
but: 2
this: 1
love: 1
good: 1
could: 1
be: 1
better: 1
did: 1
like: 1
worth: 1
price: 1
quality: 1
highly: 1
recommend: 1
to: 1
others: 1
okay: 1
great: 1

=== Code Execution Successful ===
```

```python
import pandas as pd
import string
import matplotlib.pyplot as plt
from collections import Counter
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import nltk

# Download NLTK stopwords and punkt if you don't have them
nltk.download('punkt')
nltk.download('stopwords')

# Function to preprocess the feedback text
def preprocess_text(text):
    # Convert text to lowercase
    text = text.lower()

    # Remove punctuation
    text = text.translate(str.maketrans('', '', string.punctuation))

    # Tokenize the text (split into words)
    words = word_tokenize(text)

    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    words = [word for word in words if word not in stop_words]

    return words

# Create a sample dataframe with a 'feedback' column to replace missing
# 'data.csv'
data = {
    'feedback': [
        "I love the product! It's amazing and works perfectly.",
        "The service was terrible, very disappointed.",
        "Great experience, will buy again.",
        "Not satisfied with the quality.",
        "Excellent customer support and fast delivery."
    ]
}
df = pd.DataFrame(data)

# Preprocess all feedback comments
all_words = []
for feedback in df['feedback']:
    all_words.extend(preprocess_text(feedback))

# Calculate the frequency distribution of words
word_freq = Counter(all_words)

# Set N directly since input() is not suitable here
N = 5

# Get the top N most common words
top_n_words = word_freq.most_common(N)
```

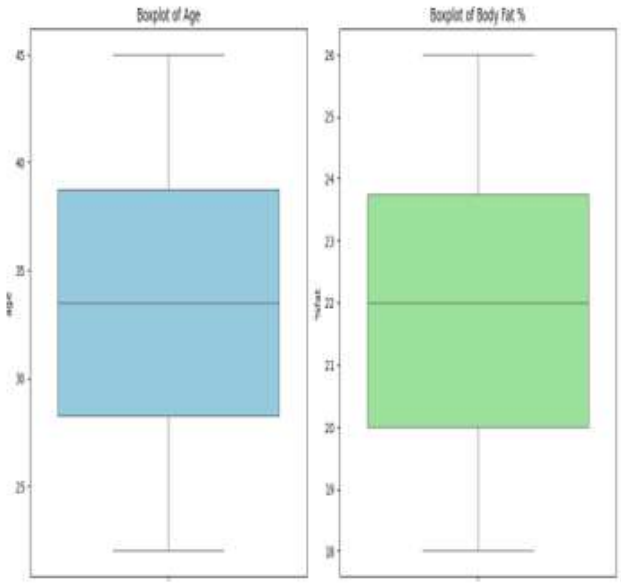
```
Top 5 Most Frequent Words:
love: 1
product: 1
amazing: 1
works: 1
perfectly: 1
```

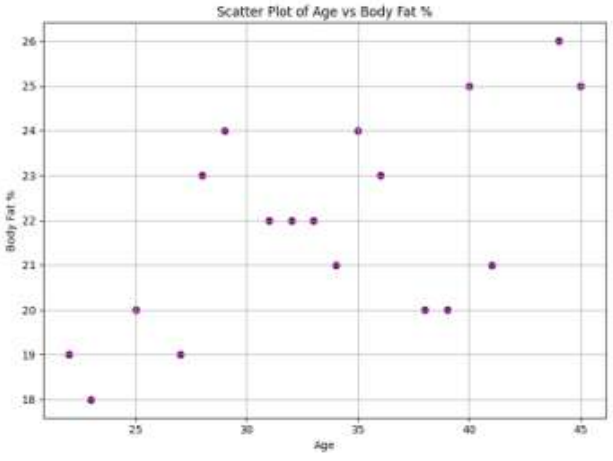


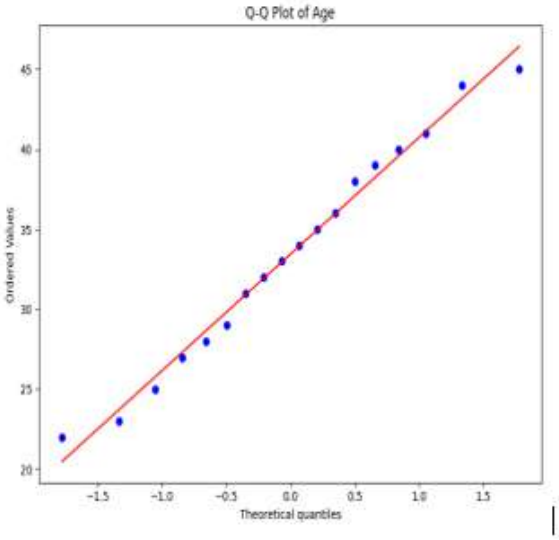
+ Code + Markdown

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats

# Sample data (age and body fat % of 10 randomly selected adults)
data = {
    'age': [23, 25, 31, 40, 22, 35, 41, 28, 39, 32, 27, 34, 44, 33, 38, 29, 45, 36],
    '%fat': [18, 20, 22, 25, 19, 24, 26, 23, 26, 25, 19, 21, 26, 25, 26, 24, 25, 21]
}

# Create DataFrame
df = pd.DataFrame(data)

# Calculate the mean, median, and standard deviation
age_mean = df['age'].mean()
age_median = df['age'].median()
age_std = df['age'].std()

fat_mean = df['%fat'].mean()
fat_median = df['%fat'].median()
fat_std = df['%fat'].std()

# Display the results
print(f"Age - Mean: {age_mean}, Median: {age_median}, Std: {age_std}")
print(f"Body Fat % - Mean: {fat_mean}, Median: {fat_median}, Std: {fat_std}")

# Draw boxplots for Age and %Fat
plt.figure(figsize=(12, 8))

plt.subplot(1, 2, 1)
sns.boxplot(data=df['age'], color='skyblue')
plt.title('Boxplot of Age')

plt.subplot(1, 2, 2)
sns.boxplot(data=df['%fat'], color='lightgreen')
plt.title('Boxplot of Body Fat %')

plt.tight_layout()
plt.show()

# Scatter plot of Age vs %Fat
plt.figure(figsize=(8, 6))
plt.scatter(df['age'], df['%fat'], color='purple')
plt.title('Scatter Plot of Age vs Body Fat %')
plt.xlabel('Age')
plt.ylabel('Body fat %')
plt.grid(True)
plt.show()

# Q-Q plot for Age
plt.figure(figsize=(8, 6))
stats.probplot(df['age'], dist="norm", plot=plt)
plt.title('Q-Q Plot of Age')
plt.show()

# Q-Q plot for Body Fat %
plt.figure(figsize=(8, 6))
stats.probplot(df['%fat'], dist="norm", plot=plt)
plt.title('Q-Q Plot of Body Fat %')
plt.show()
```

OUT PUT:







main.py

Share | Run

```python
import pandas as pd

# Load the "sales_data.csv" into a Pandas DataFrame
try:
    df = pd.read_csv('sales_data.csv')
except FileNotFoundError:
    print("Error: The file 'sales_data.csv' was not found. Please check the file path.")
    exit()

# Check the first few rows of the data
print(df.head())

# Step 1: Create a new column "Total Sales"
df['Total Sales'] = df['Quantity Sold'] * df['Unit Price']

# Step 2: Calculate the profit by applying a 20% profit margin on the "Total Sales"
df['Profit'] = df['Total Sales'] * 0.20

# Step 3: Calculate the total sales and profit for each product
product_sales_profit = df.groupby('Product').agg(
    total_sales=('Total Sales', 'sum'),
    total_profit=('Profit', 'sum')
).reset_index()

# Step 4: Sort the products by total profit and display the top 5 most profitable products
top_5_profitable_products = product_sales_profit.sort_values(by='total_profit', ascending
    =False).head(5)

# Display the top 5 most profitable products
print("\nTop 5 Most Profitable Products:")
print(top_5_profitable_products)
```

Output

Clear

```
ERROR!
Error: The file 'sales_data.csv' was not found. Please check the file path.

--- Code Execution Successful ---
```

main.py

Share | Run | Output

```python
import pandas as pd

# Load the "customer_data.csv" into a Pandas DataFrame
try:
    df = pd.read_csv('customer_data.csv')
except FileNotFoundError:
    print("Error: The file 'customer_data.csv' was not found. Please check the file path.")
    exit()

# Check the first few rows of the data
print(df.head())

# Step 1: Segment customers based on Total Spending
# Let's define the thresholds for segmentation: Low, Medium, High spenders
spending_thresholds = df['Total Spending'].quantile([0.33, 0.66])

# Function to segment based on Total Spending
def categorize_spending(row):
    if row['Total Spending'] <= spending_thresholds[0.33]:
        return 'Low Spender'
    elif row['Total Spending'] <= spending_thresholds[0.66]:
        return 'Medium Spender'
    else:
        return 'High Spender'

# Apply the function to create a new column 'Spending Segment'
df['Spending Segment'] = df.apply(categorize_spending, axis=1)

# Step 2: Calculate the average age of customers in each spending segment
average_age_per_segment = df.groupby('Spending Segment')['Age'].mean().reset_index()

# Display the average age for each spending segment
print("\nAverage Age of Customers in Each Spending Segment:")
print(average_age_per_segment)
```

```
ERROR!
Error: The file 'customer_data.csv' was not found. Please check the file path.

=== Code Execution Successful ===
```