Not syncing | Home | FIFA | numpy,pandas,matplotlib,seabor | Visualizing FIFA Player Data - Clau

localhost:8888/notebooks/numpy%2Cpandas%2Cmatplotlib%2Cseaborn%20assign.ipynb

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 29 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab

Python 3 (ipykernel)

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:
```
#1.Create a numpy array containing the numbers from 1 to 10, and then reshape it to a 2x5 matrix.
```

[22]:
```
#  Create and reshape array
arr1 = np.arange(1, 11)  # Create array from 1 to 10
reshaped_arr = arr1.reshape(2, 5)  # Reshape to 2x5
print(" Reshaped array:")
print(reshaped_arr)
print("\n")
```

```
 Reshaped array:
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]]
```

[ ]:
```
#2.Create a numpy array containing the numbers from 1 to 20, and then extract the elements between the 5th and 15th index.
```

[21]:
```
 #Array slicing
arr2 = np.arange(1, 21)  # Create array from 1 to 20
sliced_arr = arr2[5:15]  # Extract elements between 5th and 15th index
print(" Sliced array:")
print(sliced_arr)
```

```
 Sliced array:
[ 6  7  8  9 10 11 12 13 14 15]
```

[ ]:
```
#3.Create a Pandas series with the following data: {'apples': 3, 'bananas': 2, 'oranges': 1}. Then, add a new item to the series with the key 'pears' and
```

ENG IN 03:34 PM 26-10-2024

Not syncing | Home | FIFA | numpy,pandas,matplotlib,seabor | Visualizing FIFA Player Data - Clau

localhost:8888/notebooks/numpy%2Cpandas%2Cmatplotlib%2Cseaborn%20assign.ipynb

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 29 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab

Python 3 (ipykernel)

[ ]:
```
#3.Create a Pandas series with the following data: {'apples': 3, 'bananas': 2, 'oranges': 1}. Then, add a new item to the series with the key 'pears' and
```

[20]:
```
#  Create and modify series
fruit_series = pd.Series({'apples': 3, 'bananas': 2, 'oranges': 1})
fruit_series['pears'] = 4  # Add new item
print("Fruit series:")
print(fruit_series)
print("\n")
```

```
Fruit series:
apples     3
bananas    2
oranges    1
pears      4
dtype: int64
```

[ ]:
```
#4.Create a dataframe with the following columns: name, age, and gender. The dataframe should have 10 rows of data.
```

[19]:
```
#Create DataFrame
names = ['John', 'Emma', 'Alex', 'Sarah', 'Michael', 'Lisa', 'David', 'Anna', 'James', 'Emily']
ages = np.random.randint(25, 45, size=10)
genders = ['M', 'F', 'M', 'F', 'M', 'F', 'M', 'F', 'M', 'F']

df = pd.DataFrame({
    'name': names,
    'age': ages,
    'gender': genders
})

print("Initial DataFrame:")
```

ENG IN 03:35 PM 26-10-2024

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 29 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code JupyterLab Python 3 (ipykernel)

```
print("Initial DataFrame:")
print(df)
print("\n")
```

```
Initial DataFrame:
      name  age gender
0     John   44      M
1     Emma   30      F
2     Alex   29      M
3    Sarah   28      F
4  Michael   33      M
5     Lisa   43      F
6    David   32      M
7     Anna   39      F
8    James   30      M
9    Emily   31      F
```

[ ]:
```
#5.Add a new column to the data frame created in question 1, called occupation. The values for this column should be Programmer, Manager, and Analyst, co
```

[18]:
```
# Add occupation column
occupations = ['Programmer', 'Manager', 'Analyst'] * 4
occupations = occupations[:10]  # Take only first 10
df['occupation'] = occupations

print(" DataFrame with occupation:")
print(df)
print("\n")
```

```
 DataFrame with occupation:
      name  age gender  occupation
0     John   41      M  Programmer
1     Emma   40      F     Manager
2     Alex   35      M     Analyst
```

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 29 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
print(" DataFrame with occupation:")
print(df)
print("\n")
```

```
 DataFrame with occupation:
      name  age gender  occupation
0     John   41      M  Programmer
1     Emma   40      F     Manager
2     Alex   35      M     Analyst
3    Sarah   41      F  Programmer
4  Michael   44      M     Manager
5     Lisa   40      F     Analyst
6    David   42      M  Programmer
7     Anna   33      F     Manager
8    James   25      M     Analyst
9    Emily   36      F  Programmer
```

[ ]:
```
#6.Select the rows of the dataframe where the age is greater than or equal to 30.
```

[11]:
```
#Filter by age
filtered_df = df[df['age'] >= 30]
print(" Filtered DataFrame (age >= 30):")
print(filtered_df)
print("\n")
```

```
 Filtered DataFrame (age >= 30):
      name  age gender  occupation
0     John   41      M  Programmer
1     Emma   40      F     Manager
2     Alex   35      M     Analyst
3    Sarah   41      F  Programmer
4  Michael   44      M     Manager
5     Lisa   40      F     Analyst
6    David   42      M  Programmer
7     Anna   33      F     Manager
```

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 29 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
6    David   42      M  Programmer
7     Anna   33      F     Manager
9    Emily   36      F  Programmer
```

[ ]:
```
#7.Convert this dataframe to a csv file and read that csv file, finally display the contents.
```

[13]:
```
#.Save and read CSV
df.to_csv('employee_data.csv', index=False)
read_df = pd.read_csv('employee_data.csv')
print(" Read from CSV:")
print(read_df)
```

```
 Read from CSV:
      name  age gender  occupation
0     John   41      M  Programmer
1     Emma   40      F     Manager
2     Alex   35      M     Analyst
3    Sarah   41      F  Programmer
4  Michael   44      M     Manager
5     Lisa   40      F     Analyst
6    David   42      M  Programmer
7     Anna   33      F     Manager
8    James   25      M     Analyst
9    Emily   36      F  Programmer
```

[ ]:
```
#8.Create a line plot using matplotlib pyplot that displays the population of four different cities over time. Each city should have its own line, and th
The data for the four cities is provided below:
City A: [500000, 550000, 600000, 650000, 700000, 750000, 800000]
City B: [800000, 850000, 900000, 950000, 1000000, 1050000, 1100000]
City C: [1000000, 1050000, 1100000, 1150000, 1200000, 1250000, 1300000]
City D: [1200000, 1250000, 1300000, 1350000, 1400000, 1450000, 1500000]
```

[14]:
```
#.Line plot of city populations
```

```
7     Anna   33      F     Manager
8    James   25      M     Analyst
9    Emily   36      F  Programmer
```

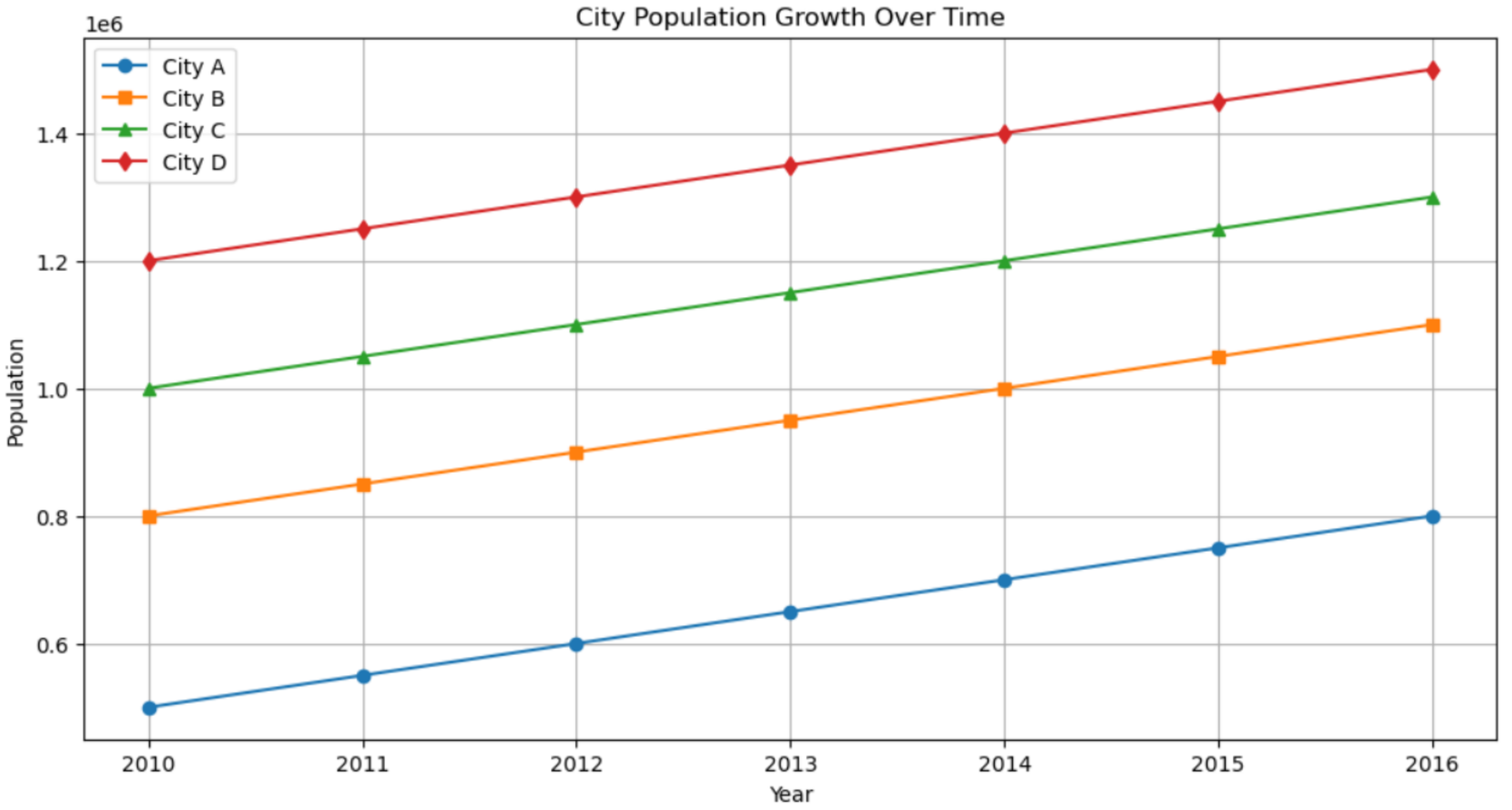
[ ]:
```
#8.Create a line plot using matplotlib pyplot that displays the population of four different cities over time. Each city should have its own line, and th
The data for the four cities is provided below:
City A: [500000, 550000, 600000, 650000, 700000, 750000, 800000]
City B: [800000, 850000, 900000, 950000, 1000000, 1050000, 1100000]
City C: [1000000, 1050000, 1100000, 1150000, 1200000, 1250000, 1300000]
City D: [1200000, 1250000, 1300000, 1350000, 1400000, 1450000, 1500000]
```

[14]:
```
#.Line plot of city populations
plt.figure(figsize=(12, 6))

years = range(2010, 2017)
city_a = [500000, 550000, 600000, 650000, 700000, 750000, 800000]
city_b = [800000, 850000, 900000, 950000, 1000000, 1050000, 1100000]
city_c = [1000000, 1050000, 1100000, 1150000, 1200000, 1250000, 1300000]
city_d = [1200000, 1250000, 1300000, 1350000, 1400000, 1450000, 1500000]

plt.plot(years, city_a, marker='o', label='City A')
plt.plot(years, city_b, marker='s', label='City B')
plt.plot(years, city_c, marker='^', label='City C')
plt.plot(years, city_d, marker='d', label='City D')

plt.title('City Population Growth Over Time')
plt.xlabel('Year')
plt.ylabel('Population')
plt.legend()
plt.grid(True)
plt.show()
```



[ ]:
```
#9.Create a scatter plot using seaborn that shows the relationship between the number of hours studied and the test scores obtained by a group of student.
Hours Studied: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Test Scores: [93, 57, 61, 54, 51, 53, 87, 81, 83, 85]
```

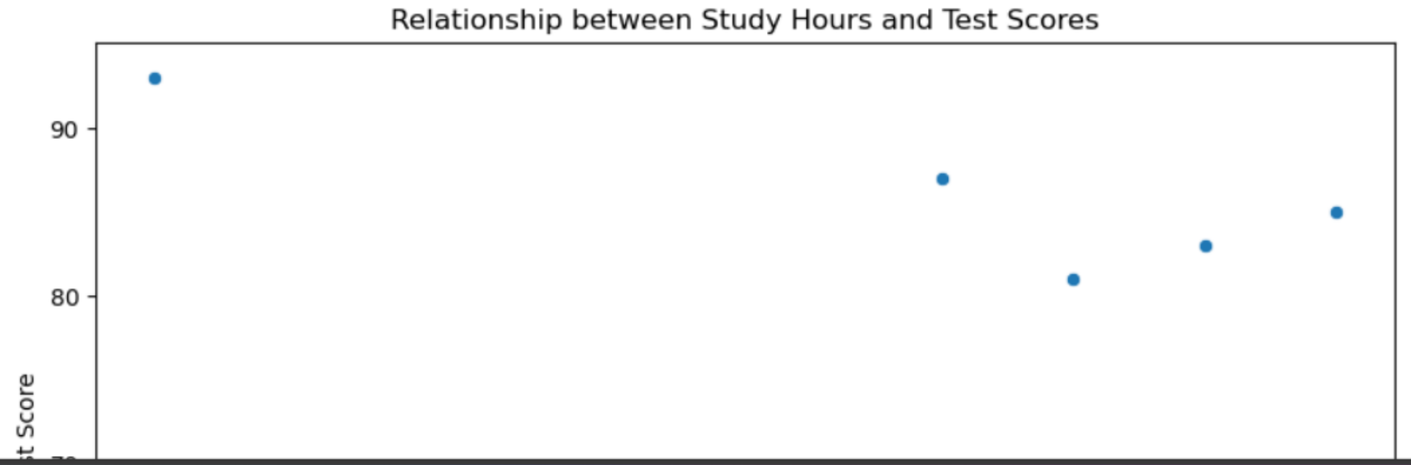
2010 2011 2012 2013 2014 2015 2016
Year

[ ]:
```
#9.Create a scatter plot using seaborn that shows the relationship between the number of hours studied and the test scores obtained by a group of student
Hours Studied: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Test Scores: [93, 57, 61, 54, 51, 53, 87, 81, 83, 85]
```

[15]:
```
 #.Scatter plot of study hours vs test scores
plt.figure(figsize=(10, 6))

hours = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
scores = [93, 57, 61, 54, 51, 53, 87, 81, 83, 85]

sns.scatterplot(x=hours, y=scores)
plt.title('Relationship between Study Hours and Test Scores')
plt.xlabel('Hours Studied')
plt.ylabel('Test Score')
plt.show()
```



jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 30 minutes ago

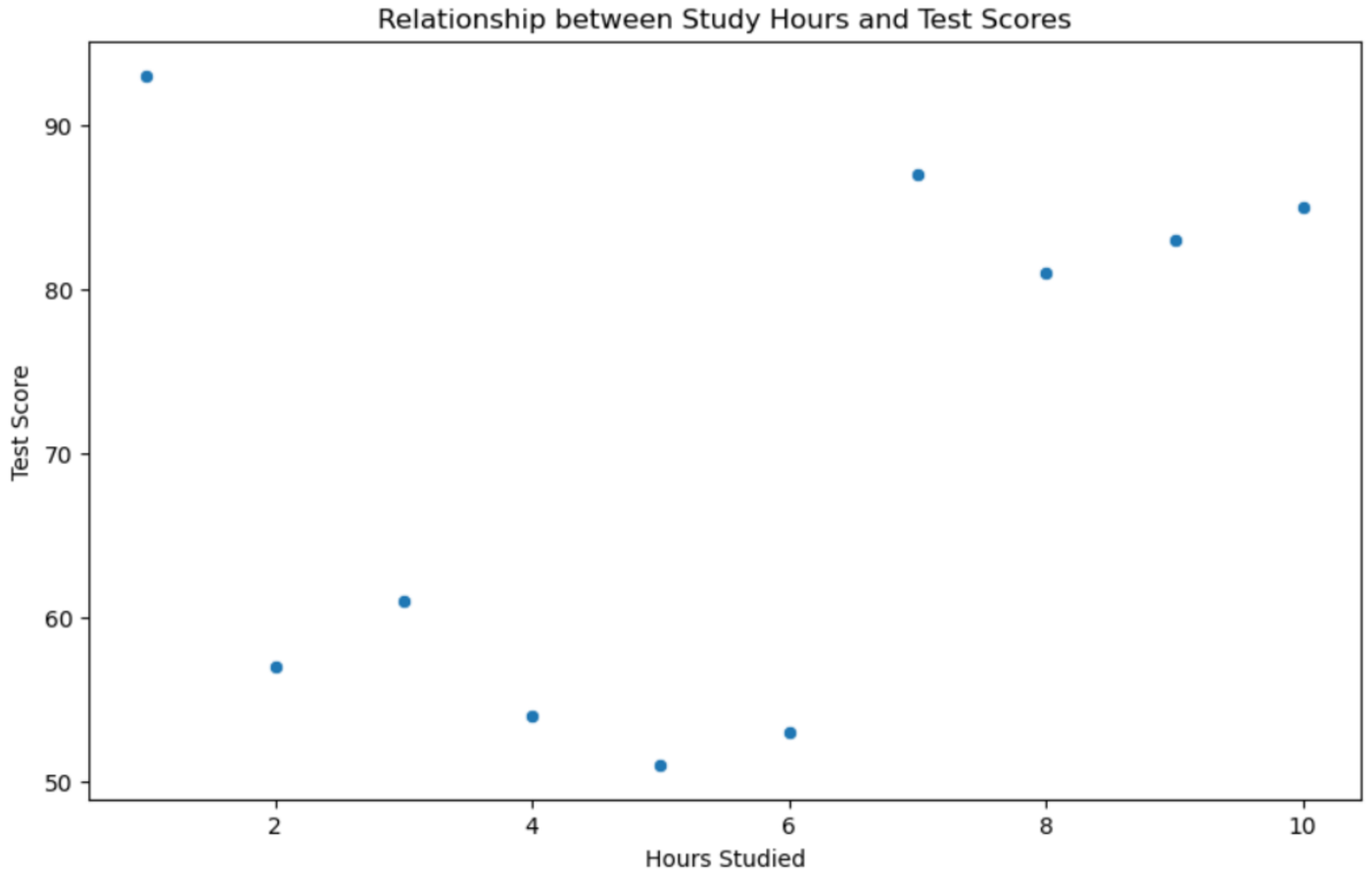
File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
plt.xlabel('Hours Studied')
plt.ylabel('Test Score')
plt.show()
```



[ ]: #10.Create a bar chart using matplotlib pyplot that shows the total sales for each month of the year. Use the following data:

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 30 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[ ]:
```
#10.Create a bar chart using matplotlib pyplot that shows the total sales for each month of the year. Use the following data:
Month: ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]
Sales: [11860, 10480, 4997, 5523, 13965, 6011, 13158, 9533, 5158, 9058, 11346, 6675]
```

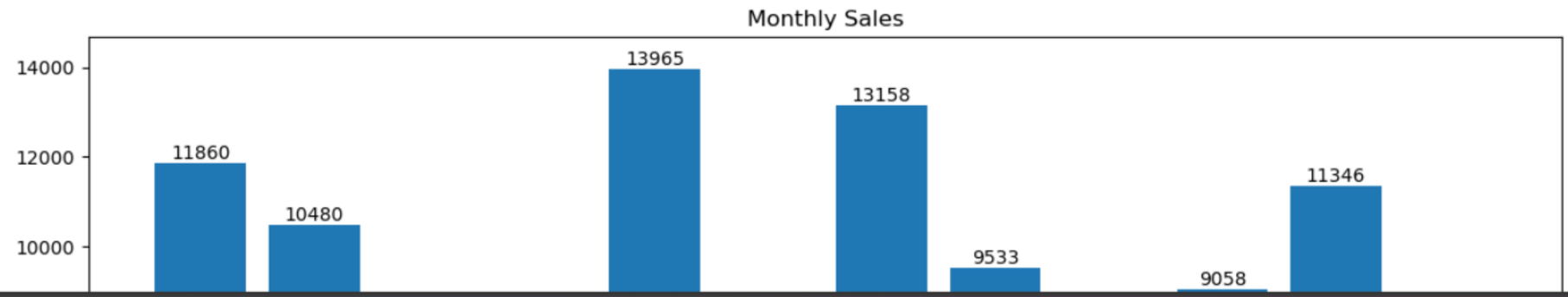
[17]:
```
#.Bar chart of monthly sales
plt.figure(figsize=(12, 6))

months = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]
sales = [11860, 10480, 4997, 5523, 13965, 6011, 13158, 9533, 5158, 9058, 11346, 6675]

plt.bar(months, sales)
plt.title('Monthly Sales')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.xticks(rotation=45)

# Add value labels on top of each bar
for i, v in enumerate(sales):
    plt.text(i, v, str(v), ha='center', va='bottom')

plt.tight_layout()
plt.show()
```



Not syncing | Home | FIFA | numpy,pandas,matplotlib,seabor | Visualizing FIFA Player Data - Clau

localhost:8888/notebooks/numpy%2Cpandas%2Cmatplotlib%2Cseaborn%20assign.ipynb

jupyter numpy,pandas,matplotlib,seaborn assign Last Checkpoint: 30 minutes ago

File Edit View Run Kernel Settings Help

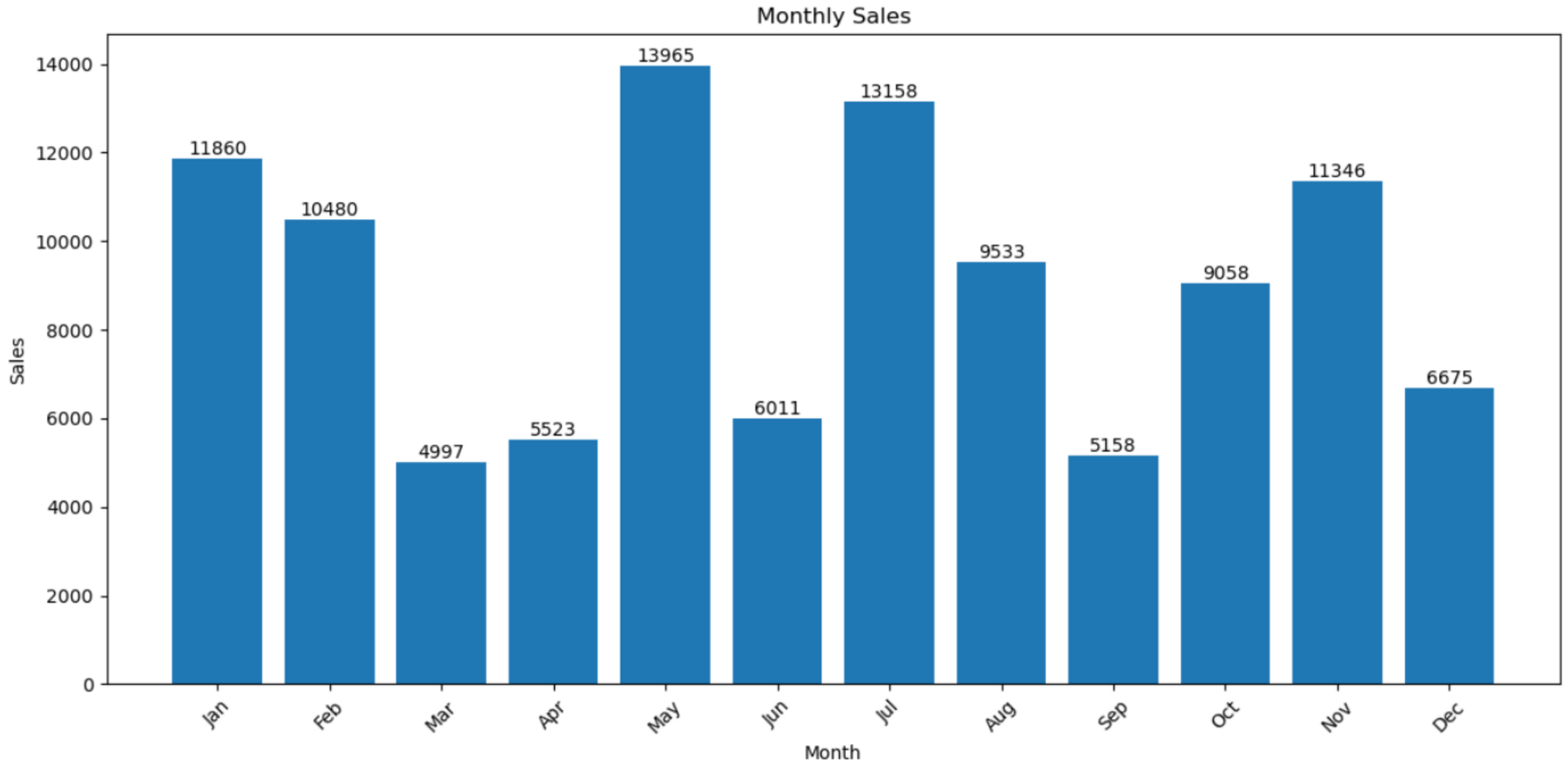
Trusted

Code

JupyterLab

Python 3 (ipykernel)

```
plt.show()
```



[ ]:

ENG IN 03:36 PM 26-10-2024