How can I plot multiple DataFrames in subplots using Pandas?

Authored by

stats writer

June 27, 2024

Pandas is a popular Python library used for data manipulation and analysis. It offers various tools for data visualization, including the ability to plot multiple DataFrames in subplots. This allows for a more efficient and organized way of presenting data. To plot multiple DataFrames in subplots using Pandas, one can use the "subplot" function, which creates a grid of plots within a single figure. This function can be customized to specify the number of rows and columns, as well as the positioning of each subplot. By using Pandas' subplot function, users can easily compare and analyze multiple DataFrames simultaneously, making data visualization more effective and informative.

Pandas: Plot Multiple DataFrames in Subplots

You can use the following basic syntax to plot multiple pandas DataFrames in subplots:

```
import matplotlib.pyplotas plt

#define subplot layout
fig, axes = plt.subplots(nrows=2, ncols=2)

#add DataFrames to subplots
df1.plot(ax=axes)
df2.plot(ax=axes)
df3.plot(ax=axes)
df4.plot(ax=axes)
```

The following example shows how to use this syntax in practice.

Example: Plot Multiple Pandas DataFrames in Subplots

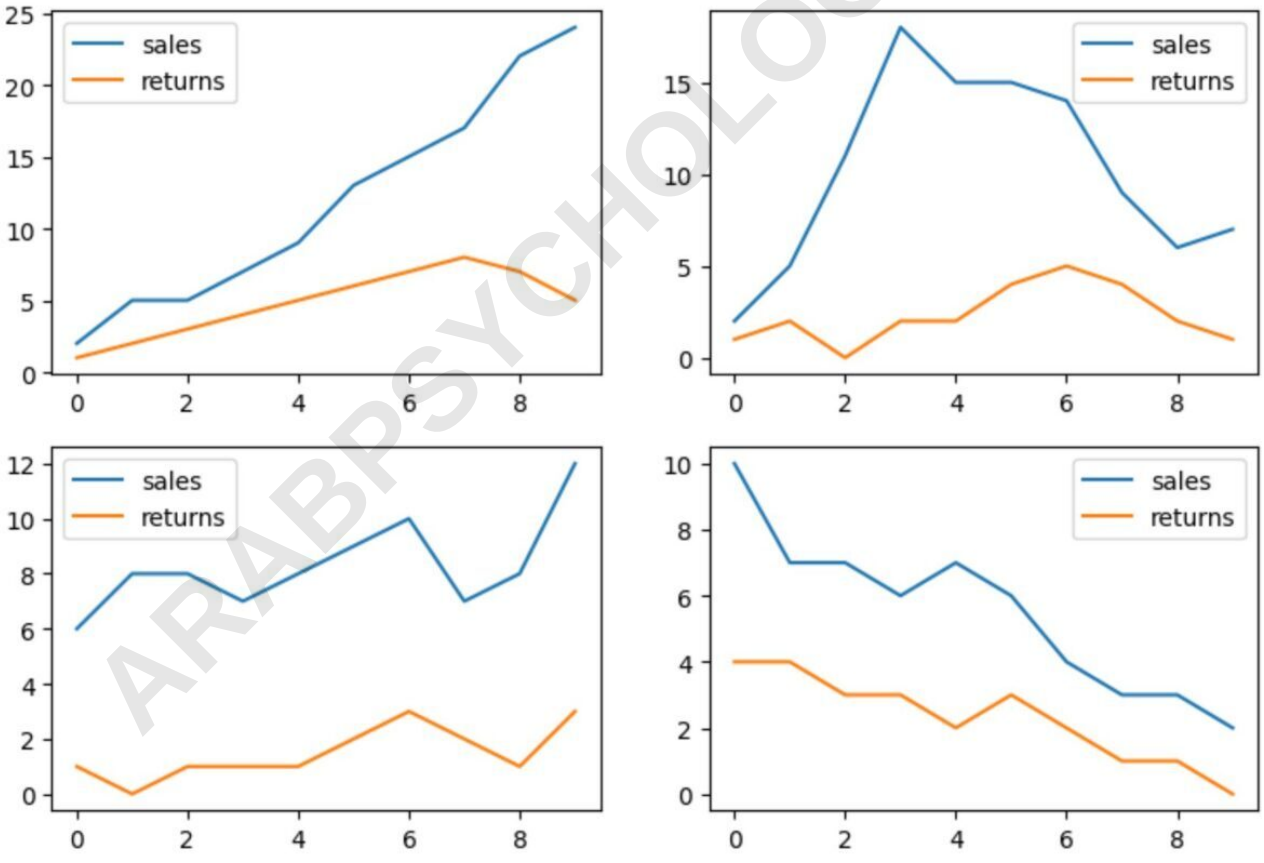
Suppose we have four pandas DataFrames that contain information on sales and returns at four different retail stores:

```
import pandas as pd

#create four DataFrames
df1 = pd.DataFrame({'sales': ,
'returns': })

df2 = pd.DataFrame({'sales': ,
'returns': })

df3 = pd.DataFrame({'sales': ,
'returns': })

df4 = pd.DataFrame({'sales': ,
'returns': })
```

We can use the following syntax to plot each of these DataFrames in a subplot that has a layout of 2 rows and 2 columns:

```
import matplotlib.pyplotas plt
```

```
#define subplot layout
fig, axes = plt.subplots(nrows=2, ncols=2)

#add DataFrames to subplots
df1.plot(ax=axes)
df2.plot(ax=axes)
df3.plot(ax=axes)
df4.plot(ax=axes)
```



Each of the four DataFrames is displayed in a subplot.

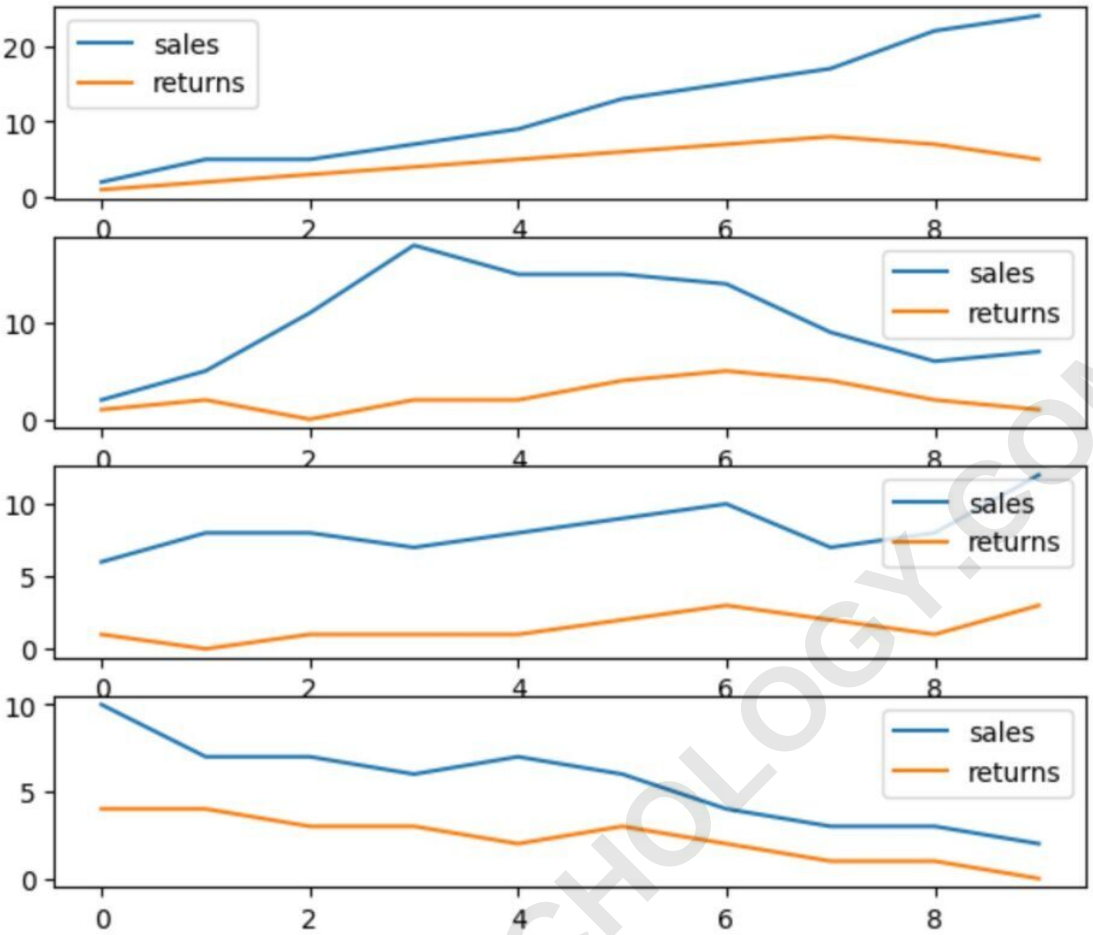
Note that we used the axes argument to specify where each DataFrame should be placed.

For example, the DataFrame called df1 was placed in the position with a row index value of 0 and a column index value of 0 (e.g. the subplot in the upper left corner).

Also note that you can change the layout of the subplots by using the nrows and ncols arguments.

For example, the following code shows how to arrange the subplots in four rows and one column:

```
import matplotlib.pyplotas plt

#define subplot layout
fig, axes = plt.subplots(nrows=4, ncols=1)

#add DataFrames to subplots
df1.plot(ax=axes)
df2.plot(ax=axes)
df3.plot(ax=axes)
df4.plot(ax=axes)
```



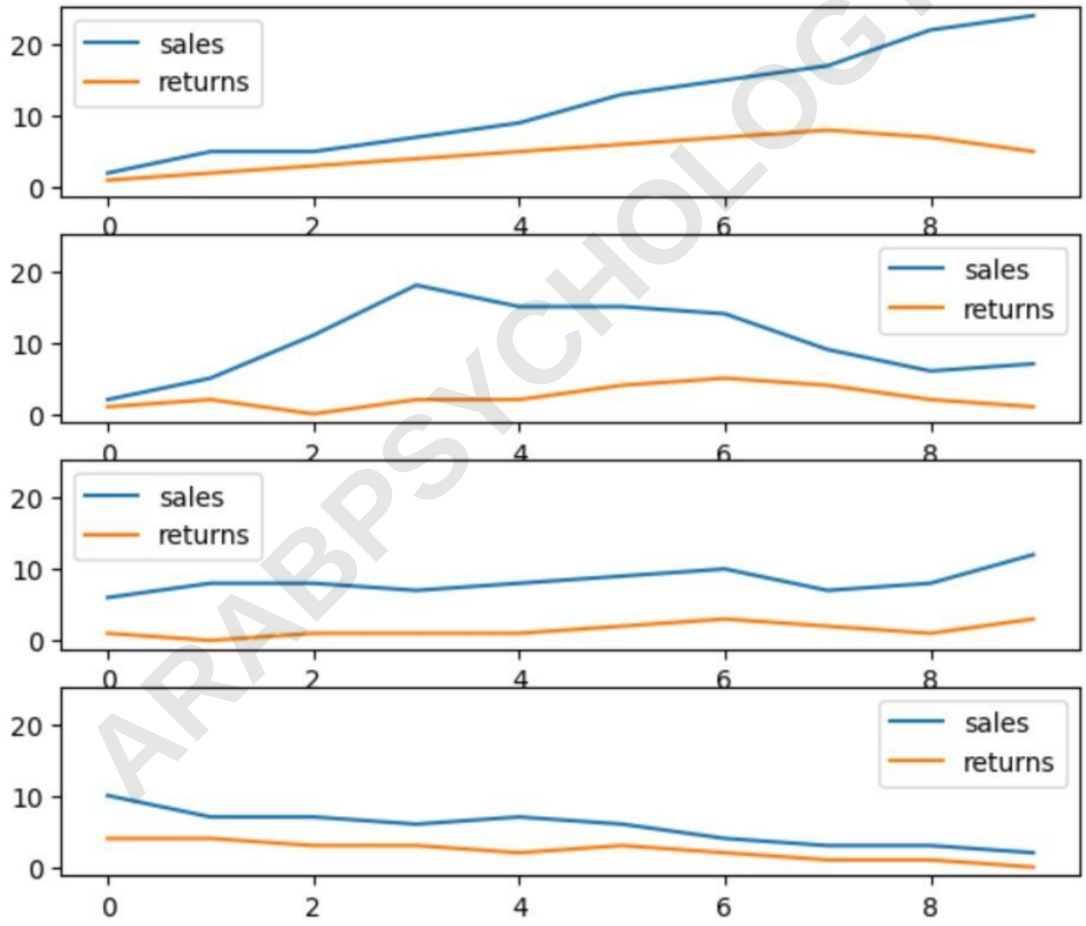
The subplots are now arranged in a layout with four rows and one column.

For example, the following code shows how to use the sharey argument to force all of the subplots to have the same y-axis scale:

import matplotlib.pyplotas plt

#define subplot layout, force subplots to have same y-axis scale

```
fig, axes = plt.subplots(nrows=4, ncols=1, sharey=True)

#add DataFrames to subplots
df1.plot(ax=axes)
df2.plot(ax=axes)
df3.plot(ax=axes)
df4.plot(ax=axes)
```



Notice that the y-axis for each subplot now ranges from 0 to 20.

The following tutorials explain how to perform other common operations in pandas: