How do I apply conditional formatting to cells in Pandas?

Authored by

stats writer

June 26, 2024

Conditional formatting is a feature in Pandas, a popular data analysis and manipulation tool, which allows users to visually highlight cells in a DataFrame based on certain conditions. This can be achieved by setting specific rules or criteria, such as numerical values, text strings, or date ranges, and applying formatting styles such as colors, font styles, or data bars to the cells that meet those conditions. This feature is useful for identifying patterns or outliers in large datasets and can be applied through the use of built-in functions or custom code. By utilizing conditional formatting in Pandas, users can efficiently analyze and interpret their data with a more visual approach.

Pandas: Apply Conditional Formatting to Cells

You can use the df.style.applymap() function to apply conditional formatting to cells in a pandas DataFrame.

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

Example: Apply Conditional Formatting to Cells in Pandas

Suppose we have the following pandas DataFrame that contains information about various basketball players:

```
import pandas as pd

#create DataFrame
df = pd.DataFrame({'points': ,
'assists': ,
'rebounds': })

#view DataFrame
```

```
print(df)

points assists rebounds
0 18 4 3
1 22 5 9
2 19 5 12
3 14 4 4
4 14 9 4
5 11 12 9
6 20 11 8
7 28 8 2
```

We can use the following code to apply a light green background to each cell in the DataFrame that has a value less than 10:

```
#define function for conditional formatting
defcond_formatting(x):
if x < 10:
return 'background-color: lightgreen'
else:
returnNone#display DataFrame with conditional formatting applied
df.style.applymap(cond_formatting)
```

	points	assists	rebounds
0	18	4	3
1	22	5	9
2	19	5	12
3	14	4	4
4	14	9	4
5	11	12	9
6	20	11	8
7	28	8	2

Notice that each cell in the DataFrame that has a value less than 10 now has a light green background.

Note: If the conditional formatting is not working in a Jupyter notebook, be sure to run the command %pip install Jinja2 first.

We can also use the color and font-weight arguments to apply more complex conditional formatting.

The following example shows how to do so:

```
#define function for conditional formatting
defcond_formatting(x):
if x < 10:
return 'background-color: lightgreen; color: red; font-
```

weight: bold'

elif x < 15:

return 'background-color: yellow'

else:

return None#display DataFrame with conditional formatting applied

df.style.applymap(cond_formatting)

	points	assists	rebounds
0	18	4	3
1	22	5	9
2	19	5	12
3	14	4	4
4	14	9	4
5	11	12	9
6	20	11	8
7	28	8	2

Here is how the conditional formatting function worked in this example:

For values less than 10, use a light green background with bold red fontFor values ≥ 10 but less than 15, use a yellow backgroundFor values greater than 15, use no conditional formatting

Feel free to use as many if, elif, and else functions as you'd like to apply as many conditional formatting rules to the cells in the DataFrame as you'd like.

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