Applying an IF condition in Pandas DataFrame

Let's now review the following 5 cases:

(1) IF condition – Set of numbers

Suppose that you created a DataFrame in Python that has 10 numbers (from 1 to 10). You then want to apply the following IF conditions:

- If the number is *equal or lower* than 4, then assign the value of 'True'
- Otherwise, if the number is *greater* than 4, then assign the value of 'False'

This is the general structure that you may use to create the IF condition:

```
df.loc[df.column name condition, 'new column name'] = 'value if condition is
met'
```

For our example, the Python code would look like this:

```
from pandas import DataFrame



Numbers = {'set_of_numbers': [1,2,3,4,5,6,7,8,9,10]}

df = DataFrame(Numbers,columns=['set_of_numbers'])



df.loc[df.set_of_numbers <= 4, 'equal_or_lower_than_4?'] = 'True'

df.loc[df.set_of_numbers > 4, 'equal_or_lower_than_4?'] = 'False'



print (df)
```

Here is the result that you'll get in Python:

```
   set_of_numbers equal_or_lower_than_4?
0               1                   True
1               2                   True
2               3                   True
3               4                   True
4               5                  False
5               6                  False
6               7                  False
7               8                  False
8               9                  False
9              10                  False
```

(2) IF condition – set of numbers and *lambda*

You'll now see how to get the same results as in case 1 by using *lambada,* where the conditions are:

- If the number is *equal or lower* than 4, then assign the value of 'True'
- Otherwise, if the number is *greater* than 4, then assign the value of 'False'

Here is the generic structure that you may apply in Python:

```
df['new column name'] = df['df column name'].apply(lambda x: 'value if
condition is met' if x condition else 'value if condition is not met')
```

And for our example:

```
from pandas import DataFrame



Numbers = {'set_of_numbers': [1,2,3,4,5,6,7,8,9,10]}

df = DataFrame(Numbers,columns=['set_of_numbers'])



df['equal or lower than 4?'] = df['set of numbers'].apply(lambda x: 'True' if
x <= 4 else 'False')



print (df)
```

This is the result that you'll get, which matches with case 1:

	set_of_numbers	equal_or_lower_than_4?
0	1	True
1	2	True
2	3	True
3	4	True
4	5	False
5	6	False
6	7	False
7	8	False
8	9	False
9	10	False

(3) IF condition – strings

Now, let's create a DataFrame that contains only strings/text with 4 *names*: Jon, Bill, Maria, Emma.

The conditions are:

- If the name is *equal to* 'Bill,' then assign the value of 'Match'
- Otherwise, if the name is *not* 'Bill,' then assign the value of 'Mis-Match'

```
from pandas import DataFrame



Names1 = {'First_name': ['Jon','Bill','Maria','Emma']}

df = DataFrame(Names1,columns=['First_name'])



df.loc[df.First_name == 'Bill', 'name_match'] = 'Match'

df.loc[df.First_name != 'Bill', 'name_match'] = 'Mis-Match'



print (df)
```

Once you run the above Python code, you'll see:

```
  First_name name_match
0        Jon  Mis-Match
1       Bill      Match
2      Maria  Mis-Match
3       Emma  Mis-Match
```

(4) IF condition – strings and *lambada*

You'll get the same results as in case 3 by using *lambada:*

```
from pandas import DataFrame



Names1 = {'First_name': ['Jon','Bill','Maria','Emma']}

df = DataFrame(Names1,columns=['First_name'])



df['name match'] = df['First name'].apply(lambda x: 'Match' if x == 'Bill'
else 'Mis-Match')



print (df)
```

And here is the output from Python:

```
  First_name name_match
0        Jon  Mis-Match
1       Bill      Match
2      Maria  Mis-Match
3       Emma  Mis-Match
```

(5) IF condition with OR

In the final case, let's apply these conditions:

- If the name is 'Bill' **or** 'Emma,' then assign the value of 'Match'
- Otherwise, if the name is neither 'Bill' nor 'Emma,' then assign the value of 'Mis-Match'

```
from pandas import DataFrame
```

```
Names1 = {'First_name': ['Jon','Bill','Maria','Emma']}

df = DataFrame(Names1,columns=['First_name'])



df.loc[(df.First name == 'Bill') | (df.First name == 'Emma'), 'name match'] =
'Match'

df.loc[(df.First name != 'Bill') & (df.First name != 'Emma'), 'name match'] =
'Mis-Match'



print (df)
```

Run the Python code, and you'll get the following result:

```
  First_name name_match
0        Jon  Mis-Match
1       Bill      Match
2      Maria  Mis-Match
3       Emma      Match
```