Week 7 Python

March 14, 2024

1 Week 7 Python

1.1 Filtering Data

[1]:
```
import pandas as pd
```

We will import the empoyee data set we used before

[2]:
```
df = pd.read_csv('updated_employee_dataset.csv')
```

We can look at the description of the dataframe by doing the following.

[3]:
```
df.describe()
```

[3]:

	Employee_ID	Years_of_Experience	Salary
count	50.00000	50.000000	50.000000
mean	25.50000	10.017600	75044.000000
std	14.57738	5.462507	13656.266372
min	1.00000	1.120000	52800.000000
25%	13.25000	5.277500	63193.750000
50%	25.50000	10.210000	75525.000000
75%	37.75000	14.607500	86518.750000
max	50.00000	19.420000	98550.000000

The `.describe()` method in pandas generates summary statistics of numerical columns in a DataFrame, such as count, mean, standard deviation, min, max, and quartiles (25%, 50%, 75%). It provides a quick overview of the data's distribution and central tendency.

Next let's look at the first few rows of the dataframe

[4]:
```
df.head(10)
```

[4]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	\
0	1	Finance	1.82	False	Good	
1	2	IT	2.09	False	Poor	
2	3	IT	13.24	False	Excellent	
3	4	Finance	1.55	False	Excellent	
4	5	Marketing	4.16	True	Average	
5	6	Finance	7.42	True	Excellent	

```
6            7  Marketing                10.42       True              Good
7            8  Marketing                12.90       True              Good
8            9  Marketing                10.00       True              Poor
9           10         HR                18.74      False           Average
```

```
    Salary First_Name Last_Name
0  54550.0    Michael     Davis
1  55225.0      Karen     Brown
2  83100.0     Joseph   Johnson
3  53875.0      David    Garcia
4  60400.0      Linda  Martinez
5  68550.0    Michael     Brown
6  76050.0    Charles     Moore
7  82250.0      David     Lopez
8  75000.0      David   Johnson
9  96850.0   Patricia  Martinez
```

Let's say you want to look at people working in Finance department. This means looking at all the rows where

[5]:
```
filter1=( df['Department']=='Finance' )
```

This code creates a filter named `filter1` that can be used to select rows from a DataFrame `df` where the 'Department' column has values equal to 'Finance'. Essentially, it checks each row in the 'Department' column to see if it matches 'Finance' and returns a boolean `Series` (`True`for rows that match, `False` for those that don't), which can then be used to filter the DataFrame.

[6]:
```
df[filter1]
```

[6]:
```
    Employee_ID Department  Years_of_Experience  Full_Time Performance_Score  \
0             1    Finance                 1.82      False              Good
3             4    Finance                 1.55      False         Excellent
5             6    Finance                 7.42       True         Excellent
10           11    Finance                 6.17      False              Poor
14           15    Finance                14.47       True              Good
23           24    Finance                18.83       True           Average
25           26    Finance                 8.55       True         Excellent
28           29    Finance                 8.09      False              Good
29           30    Finance                 8.82       True         Excellent
31           32    Finance                 2.08      False              Poor
34           35    Finance                15.13       True              Poor
38           39    Finance                16.00      False              Good
40           41    Finance                 1.12       True           Average
44           45    Finance                11.35       True         Excellent
45           46    Finance                14.65       True              Good
```

```
    Salary First_Name  Last_Name
0  54550.0    Michael      Davis
```

```
3   53875.0      David     Garcia
5   68550.0    Michael      Brown
10  65425.0      Linda      Brown
14  86175.0   Jennifer      Davis
23  97075.0     Joseph   Anderson
25  71375.0    Barbara      Smith
28  70225.0     Robert      Smith
29  72050.0    Charles      Lopez
31  55200.0    William  Hernandez
34  87825.0   Patricia      Davis
38  90000.0       Mary      Jones
40  52800.0     Joseph     Miller
44  78375.0     Joseph      Davis
45  86625.0       John      Lopez
```

The code `df[filter1]` applies the previously defined filter `filter1` to the DataFrame `df`. It selects and returns only those rows where the 'Department' column is equal to 'Finance', as determined by the condition in `filter1`. This effectively filters the DataFrame to only include records related to the Finance department.

[7]:
```
df[df['Department']=='Finance' ]
```

[7]:
```
    Employee_ID Department  Years_of_Experience  Full_Time Performance_Score  \
0             1    Finance                 1.82      False              Good
3             4    Finance                 1.55      False         Excellent
5             6    Finance                 7.42       True         Excellent
10           11    Finance                 6.17      False              Poor
14           15    Finance                14.47       True              Good
23           24    Finance                18.83       True           Average
25           26    Finance                 8.55       True         Excellent
28           29    Finance                 8.09      False              Good
29           30    Finance                 8.82       True         Excellent
31           32    Finance                 2.08      False              Poor
34           35    Finance                15.13       True              Poor
38           39    Finance                16.00      False              Good
40           41    Finance                 1.12       True           Average
44           45    Finance                11.35       True         Excellent
45           46    Finance                14.65       True              Good

     Salary First_Name  Last_Name
0   54550.0    Michael      Davis
3   53875.0      David     Garcia
5   68550.0    Michael      Brown
10  65425.0      Linda      Brown
14  86175.0   Jennifer      Davis
23  97075.0     Joseph   Anderson
25  71375.0    Barbara      Smith
```

```
28  70225.0     Robert      Smith
29  72050.0    Charles      Lopez
31  55200.0    William  Hernandez
34  87825.0   Patricia      Davis
38  90000.0       Mary      Jones
40  52800.0     Joseph     Miller
44  78375.0     Joseph      Davis
45  86625.0       John      Lopez
```

The code `df[df['Department'] == 'Finance']` directly filters the DataFrame `df` to include only rows where the 'Department' column equals 'Finance'. This line of code combines the creation of the filter and its application into a single step. It checks each row in the 'Department' column for the condition (`'Department' == 'Finance'`) and returns a new DataFrame containing only those rows that meet the criteria. This is a common pattern in pandas for filtering data based on a specific condition.

We have not saved the filtered data anywhere. Lets do that by creating a new dataframe called `df_filter1`

[8]:
```
df_filter1=df[df['Department']=='Finance' ]
```

[9]:
```
df_filter1
```

[9]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	Salary	First_Name	Last_Name
0	1	Finance	1.82	False	Good	54550.0	Michael	Davis
3	4	Finance	1.55	False	Excellent	53875.0	David	Garcia
5	6	Finance	7.42	True	Excellent	68550.0	Michael	Brown
10	11	Finance	6.17	False	Poor	65425.0	Linda	Brown
14	15	Finance	14.47	True	Good	86175.0	Jennifer	Davis
23	24	Finance	18.83	True	Average	97075.0	Joseph	Anderson
25	26	Finance	8.55	True	Excellent	71375.0	Barbara	Smith
28	29	Finance	8.09	False	Good			
29	30	Finance	8.82	True	Excellent			
31	32	Finance	2.08	False	Poor			
34	35	Finance	15.13	True	Poor			
38	39	Finance	16.00	False	Good			
40	41	Finance	1.12	True	Average			
44	45	Finance	11.35	True	Excellent			
45	46	Finance	14.65	True	Good			

```
28  70225.0     Robert      Smith
29  72050.0    Charles      Lopez
31  55200.0    William  Hernandez
34  87825.0   Patricia      Davis
38  90000.0       Mary      Jones
40  52800.0     Joseph     Miller
44  78375.0     Joseph      Davis
45  86625.0       John      Lopez
```

What if you wanted to look employees in Finance who scored Excellent!

[10]:
```
filter2=( df['Department']=='Finance' ) & (df['Performance_Score']=='Excellent')
```

The code `filter2 = (df['Department'] == 'Finance') & (df['Performance_Score'] == 'Excellent')` creates a more specific filter named `filter2` that combines two conditions with an AND operator (`&`):

1. It checks if the 'Department' column is equal to 'Finance'.
2. It also checks if the 'Performance_Score' column is equal to 'Excellent'.

This filter will return `True` for rows where both conditions are met simultaneously, meaning it identifies employees who are in the Finance department and have an 'Excellent' performance score. This allows for more targeted data selection from the DataFrame `df`.

The `&` operator is used in pandas for element-wise logical AND operations between conditions, requiring parentheses around each condition. The `and` keyword is used for logical AND operations in Python's control structures, evaluating the overall truthiness of each side. Use `&` for combining conditions in pandas filtering, and `and` for combining boolean expressions in if-statements and other logical controls.

Let's apply the filter!

[11]:
```
df[filter2]
```

[11]:
```
    Employee_ID Department  Years_of_Experience  Full_Time Performance_Score  \
3             4    Finance                 1.55      False         Excellent
5             6    Finance                 7.42       True         Excellent
25           26    Finance                 8.55       True         Excellent
29           30    Finance                 8.82       True         Excellent
44           45    Finance                11.35       True         Excellent

     Salary First_Name Last_Name
3   53875.0      David    Garcia
5   68550.0    Michael     Brown
25  71375.0    Barbara     Smith
29  72050.0    Charles     Lopez
44  78375.0     Joseph     Davis
```

[12]:
```
filter3=df['Years_of_Experience']>=10
df[filter3]
```

[12]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	Salary	First_Name	Last_Name
2	3	IT	13.24	False	Excellent	83100.0	Joseph	Johnson
6	7	Marketing	10.42	True	Good	76050.0	Charles	Moore
7	8	Marketing	12.90	True	Good	82250.0	David	Lopez
8	9	Marketing	10.00	True	Poor	75000.0	David	Johnson
9	10	HR	18.74	False	Average	96850.0	Patricia	Martinez
11	12	IT	11.26	True	Average	78150.0	John	Wilson
12	13	HR	19.10	True	Good	97750.0	Mary	Thomas
14	15	Finance	14.47	True	Good	86175.0	Jennifer	Davis
17	18	HR	12.48	True	Poor	81200.0	Barbara	Thomas
20	21	IT	15.91	False	Average	89775.0	Elizabeth	Anderson
21	22	IT	15.97	True	Poor	89925.0	Jessica	Gonzalez
23	24	Finance	18.83	True	Average	97075.0	Joseph	Anderson
24	25	Marketing	11.52	True	Excellent	78800.0	Charles	Gonzalez
26	27	HR	10.74	True	Good	76850.0	Karen	Miller
30	31	IT	19.08	True	Good	97700.0	Michael	Davis
33	34	HR	15.79	True	Poor	89475.0	Sarah	Taylor
34	35	Finance	15.13	True	Poor	87825.0	Patricia	Davis
36	37	IT	13.26	False	Poor	83150.0	Sarah	Williams
37	38	IT	19.42	False	Good			
38	39	Finance	16.00	False	Good			
39	40	HR	11.64	True	Excellent			
41	42	Marketing	14.48	False	Good			
42	43	Marketing	15.61	True	Poor			
44	45	Finance	11.35	True	Excellent			
45	46	Finance	14.65	True	Good			
47	48	IT	16.16	True	Excellent			

```
37  98550.0    William      Smith
38  90000.0       Mary      Jones
39  79100.0      Karen   Martinez
41  86200.0    Michael  Rodriguez
42  89025.0    Barbara     Miller
44  78375.0     Joseph      Davis
45  86625.0       John      Lopez
47  90400.0     Thomas      Smith
```

The code creates a filter `filter3` that selects rows from the DataFrame `df` where the 'Years_of_Experience' column has values greater than or equal to 10. This filter identifies employees with 10 or more years of experience. `df[filter3]` then applies this filter to `df`, returning a new DataFrame containing only the rows that meet this condition, effectively filtering the dataset to include only those employees with a significant amount of experience.

[13]:
```
df['Company']= 'XYZ _ltd'
```

The code `df['Company'] = 'XYZ _ltd'` assigns the string 'XYZ _ltd' to a new or existing column named 'Company' in the DataFrame `df`. This operation sets every row in the 'Company' column to have the value 'XYZ _ltd', effectively creating or updating the 'Company' column to uniformly represent that all entries belong to 'XYZ _ltd'. If the 'Company' column didn't previously exist, it will be created with this assignment.

[14]:
```
df.head(10)
```

[14]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	Salary	First_Name	Last_Name	Company
0	1	Finance	1.82	False	Good	54550.0	Michael	Davis	XYZ _ltd
1	2	IT	2.09	False	Poor	55225.0	Karen	Brown	XYZ _ltd
2	3	IT	13.24	False	Excellent	83100.0	Joseph	Johnson	XYZ _ltd
3	4	Finance	1.55	False	Excellent	53875.0	David	Garcia	XYZ _ltd
4	5	Marketing	4.16	True	Average	60400.0	Linda	Martinez	XYZ _ltd
5	6	Finance	7.42	True	Excellent	68550.0	Michael	Brown	XYZ _ltd
6	7	Marketing	10.42	True	Good	76050.0	Charles	Moore	XYZ _ltd
7	8	Marketing	12.90	True	Good	82250.0	David	Lopez	XYZ _ltd
8	9	Marketing	10.00	True	Poor	75000.0	David	Johnson	XYZ _ltd
9	10	HR	18.74	False	Average	96850.0	Patricia	Martinez	XYZ _ltd

[15]:
```
df['Senior']= 0
```

The code `df['Senior'] = 0` creates or updates a column named 'Senior' in the DataFrame `df`, setting its value to 0 for all rows. This effectively adds a new feature or attribute to the dataset, where every entry is initially marked as not senior (assuming 0 is used to indicate a false or negative condition for being a senior employee). If the 'Senior' column didn't exist before, it will be created with this operation.

[16]:
```
df.head(10)
```

[16]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	\
0	1	Finance	1.82	False	Good	
1	2	IT	2.09	False	Poor	
2	3	IT	13.24	False	Excellent	
3	4	Finance	1.55	False	Excellent	
4	5	Marketing	4.16	True	Average	
5	6	Finance	7.42	True	Excellent	
6	7	Marketing	10.42	True	Good	
7	8	Marketing	12.90	True	Good	
8	9	Marketing	10.00	True	Poor	
9	10	HR	18.74	False	Average	

	Salary	First_Name	Last_Name	Company	Senior
0	54550.0	Michael	Davis	XYZ _ltd	0
1	55225.0	Karen	Brown	XYZ _ltd	0
2	83100.0	Joseph	Johnson	XYZ _ltd	0
3	53875.0	David	Garcia	XYZ _ltd	0
4	60400.0	Linda	Martinez	XYZ _ltd	0
5	68550.0	Michael	Brown	XYZ _ltd	0
6	76050.0	Charles	Moore	XYZ _ltd	0
7	82250.0	David	Lopez	XYZ _ltd	0
8	75000.0	David	Johnson	XYZ _ltd	0
9	96850.0	Patricia	Martinez	XYZ _ltd	0

[17]:
```
df['Senior'][df['Years_of_Experience']>=10]=1
```

```
/var/folders/01/jybr74m10l5d2ldzyht8kj5h0000gn/T/ipykernel_1236/2092124849.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Senior'][df['Years_of_Experience']>=10]=1
```

The code `df['Senior'][df['Years_of_Experience'] >= 10] = 1` is used to update the 'Senior' column in the DataFrame `df`, setting its value to 1 for all rows where the 'Years_of_Experience' is greater than or equal to 10. This effectively marks employees with 10 or more years of experience as senior (assuming 1 indicates a true or positive condition for being a senior employee).

However, it's important to note that this direct indexing method to set values based on a condition (also known as chained assignment) can sometimes lead to unpredictable results or a `SettingWithCopyWarning` in pandas. A more recommended approach for such operations would be to use the `.loc` method to ensure the operation is performed correctly and to avoid potential issues:

```
df.loc[df['Years_of_Experience'] >= 10, 'Senior'] = 1
```

This method clearly specifies the rows to be updated and the column on which the operation should be performed, reducing the risk of unintended side effects.

[18]:
```
df.head(10)
```

[18]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	\
0	1	Finance	1.82	False	Good	
1	2	IT	2.09	False	Poor	
2	3	IT	13.24	False	Excellent	
3	4	Finance	1.55	False	Excellent	
4	5	Marketing	4.16	True	Average	
5	6	Finance	7.42	True	Excellent	
6	7	Marketing	10.42	True	Good	
7	8	Marketing	12.90	True	Good	
8	9	Marketing	10.00	True	Poor	
9	10	HR	18.74	False	Average	

	Salary	First_Name	Last_Name	Company	Senior
0	54550.0	Michael	Davis	XYZ _ltd	0
1	55225.0	Karen	Brown	XYZ _ltd	0
2	83100.0	Joseph	Johnson	XYZ _ltd	1
3	53875.0	David	Garcia	XYZ _ltd	0
4	60400.0	Linda	Martinez	XYZ _ltd	0
5	68550.0	Michael	Brown	XYZ _ltd	0
6	76050.0	Charles	Moore	XYZ _ltd	1
7	82250.0	David	Lopez	XYZ _ltd	1
8	75000.0	David	Johnson	XYZ _ltd	1
9	96850.0	Patricia	Martinez	XYZ _ltd	1

[19]:
```
df['Email1']=df['First_Name']+"."+df['Last_Name']+'@xyz.com'
```

[20]:
```
df.head(10)
```

[20]:

	Employee_ID	Department	Years_of_Experience	Full_Time	Performance_Score	\
0	1	Finance	1.82	False	Good	
1	2	IT	2.09	False	Poor	
2	3	IT	13.24	False	Excellent	
3	4	Finance	1.55	False	Excellent	
4	5	Marketing	4.16	True	Average	
5	6	Finance	7.42	True	Excellent	

```
6            7  Marketing                10.42       True              Good
7            8  Marketing                12.90       True              Good
8            9  Marketing                10.00       True              Poor
9           10         HR                18.74      False           Average
```

```
    Salary First_Name Last_Name   Company  Senior                  Email1
0  54550.0    Michael     Davis  XYZ _ltd       0      Michael.Davis@xyz.com
1  55225.0      Karen     Brown  XYZ _ltd       0        Karen.Brown@xyz.com
2  83100.0     Joseph   Johnson  XYZ _ltd       1     Joseph.Johnson@xyz.com
3  53875.0      David    Garcia  XYZ _ltd       0       David.Garcia@xyz.com
4  60400.0      Linda  Martinez  XYZ _ltd       0     Linda.Martinez@xyz.com
5  68550.0    Michael     Brown  XYZ _ltd       0      Michael.Brown@xyz.com
6  76050.0    Charles     Moore  XYZ _ltd       1      Charles.Moore@xyz.com
7  82250.0      David     Lopez  XYZ _ltd       1        David.Lopez@xyz.com
8  75000.0      David   Johnson  XYZ _ltd       1      David.Johnson@xyz.com
9  96850.0   Patricia  Martinez  XYZ _ltd       1  Patricia.Martinez@xyz.com
```

The code `df['Email'] = df['First_Name'] + "." + df['Last_Name'] + '@xyz.com'` constructs an email address for each row in the DataFrame `df` by concatenating the 'First_Name' and 'Last_Name' columns with a period between them, and then appending '@xyz.com' at the end. This operation creates or updates the 'Email' column, where each row's value is now a string representing the employee's email address in the format of "first_name.last_name@xyz.com". This is a common pattern for automatically generating email addresses in datasets where employees are given standardized email formats based on their names.

[ ]: