3.0 Creating & Modifying a Dataframe/Series

May 19, 2019 5:18 PM

0.0 Showing Dataframe

Some of the concepts will be covered later but stated here for organization purposes

dataframe.iloc [rows, columns]
- Pretty prints with index labels and column labels

dataframe.head ()
- Pretty prints with index labels and column labesl

dataframe.values.tolist ()
- Converts the dataframe to a list of list. Does not include the column labels (and of course not the index labels)

When writing lines that displays a dataframe, it needs to be at the end of that code block or it needs to be in a code block of its own to display the dataframe when the code block is ran

Example:

```
android.head()
print ("# of rows: ", android.shape[0])
print ("# of columns: ",android.shape[1])
```

Did not display the top 5 rows as per head command

So separated out android.head () in a separate box and it displayed correctly

1.0 Creating a Dataframe

Pandas is created from a dictionary (unlike NumPy which is created from a list (1D array) or list of list (2D array)

1.1.1 Using a CSV File

Dataframe is usually created by reading a CSV file

pd.read_csv (arguments)

Commonly used arguments:

```
f500 = pd.read_csv('f500.csv',index_col=0)
```

Screen clipping taken: 2019-05-18 6:30 PM

File path

"f500.csv" is file path. Full file path needs to be specified if the data file is not in the same folder as python code file

Index_Col

Index_col = 0 ==> use the first column as row labels

With index_col = 0

	rank	revenues	revenue_change
Walmart	1	485873	0.8
State Grid	2	315199	-4.4
Sinopec Group	3	267518	-9.1

Screen clipping taken: 2019-05-18 6:35 PM

Columns labels are: rank, revenues, revenue_change
Row labels are: Walmart, State Grid, Sinopec Group

Without index_col = 0 (by default it is set to "none")

	company	rank	revenues	revenue_change
0	Walmart	1	485873	0.8
1	State Grid	2	315199	-4.4
2	Sinopec Group	3	267518	-9.1
3	China National Petroleum	4	262573	-12.3

Screen clipping taken: 2019-05-18 6:36 PM

Column labels stays the same except that we have one new column label "company" which contains the company's names

Row labels are numeric and starts at 0 (example: 0,1,2,3…). We still need to wrap the numeric labels in quotation mark when selecting by labels!

Using index_col = 0 is more commonly used approach

Encoding

Encoding is a way to translate character (any language) to computer's binary representation

Most common encodings are:

- UTF-8 (the default for Python)
- Latin-1 (also known as ISO-8895-1)
- Windows-1251

By default, encoding argument is set to "UTF-8-encoding"

If reading a csv file into a dataframe with default encoding argument does not work then specify any of the other two encoding

The argument is: encoding = "UTF-8-encoding" for example

Full documentation is here: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

Delimiter

Can set the delimiter to something else

Full documentation: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

na_values

String/list; used to specify which values in a column we should also read as nan

1.1.2 Using csv file Link Directly

Example:

f500 = pd.read_csv (direct_link, low_memory = 0)

Where direct_link contains the url to the csv file
low_memory = 0 (or False) to silence dtypes warning

See Guided Project: Finding the Best Markets to Advertise In for real example

1.2 Creating Dataframe From Scratch + Adding to Dataframe

To create a dataframe from scratch, first create a dictionary

Key: Value

"Key" becomes the column labels and "value" becomes the cell values

Then use pd.DataFrame (dictionary) to create the dataframe

Other Ways of Creating Dataframe + Adding Columns

Bunch of ways to create dataframe as well as adding columns is here:

https://www.geeksforgeeks.org/adding-new-column-to-existing-dataframe-in-pandas/

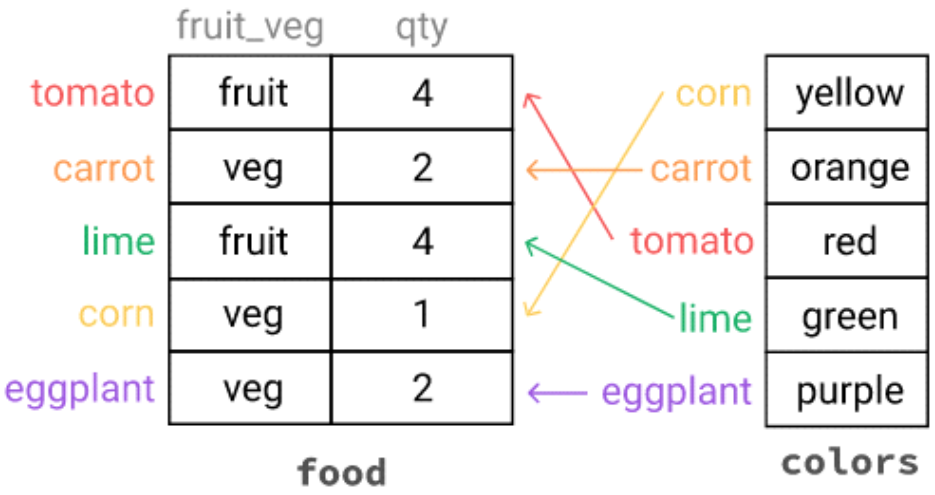
Adding Rows

Adding rows to dataframe involves using Dataframe.append () method. Instruction on how to use this method is here:

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.append.html

Adding Series to a Dataframe

When adding a series to a dataframe,
panda adds the rows in series to the dataframe by matching the index labels regardless of the order of data



When adding a series to a dataframe, if panda does not find the matching index labels, it assigns Nan to the new column in the dataframe

This works whether the index labels are strings or integers as long as you have not made the modifications to the index labels, you can use index alignment to your advantage

When we try to assign value to a column that does not exist, Panda just creates a new column for the dataframe

1.3 Assigning a Value

General syntax:

dataframe.loc [row_label, column_label] = value
dataframe.iloc {row_index, column_index] = value

To assign null values, type in np.nan (np stands for numpy)

Can use boolean filtering and assigning value in one line

1.4 Dataframe Index

We can assign axis name (column axis or row axis)

To set the index to none, use

Dataframe.index.name = None

Renaming index axis name:

```
# reset the index name
df.rename_axis('Index_name', axis = 'rows')
```

Screen clipping taken: 2019-05-19 7:38 PM

Renaming column axis name:

```
# set the name of column axes
df.rename_axis('Column_Index_name', axis = 'columns')
```

Screen clipping taken: 2019-05-19 7:39 PM

More details can be found here:

https://www.geeksforgeeks.org/setting-the-name-of-the-axes-in-pandas-dataframe/

1.5 Labels Retention and Row/Column Labels vs. Row/Column Index

As discussed below and somewhat above, when we extract a series or a dataframe from a dataframe, the subset data maintains the labels from the original dataframe. Thus, we could still use the labels from the original dataframe to select data.

If our dataframe contains numeric row labels then some of the row labels could be destroyed in one of the following ways:

- Selecting a subset of the data
    - The index labels are retained from the original data. So for example, if we extracted rows with labels 300, 418, 481, the row labels are still 300, 418, 481
    - However, using integer positions, first row is index 0, second row is index 1, third row is index 3
- (the subset contains the index labels from original data; it does not restart at index label = 0]
- Removing certain rows (e.g when we had to remove rows that had null values)
- Randomizing the order of the rows in our dataframe (common occurrence in machine learning)
- Sorting the rows

However, no matter what, the index position adapts to the new dataset. So the first row/column is at index 0, second row/column is at index 1, etc.

When we use an index that is not present in our dataset, we get this error: the label [x] is not in in the [index]

1.6 Reordering Columns

Create a list with the columns in specific order

Then new_df = old_df [column_order]

Where column_order is a list like shown below

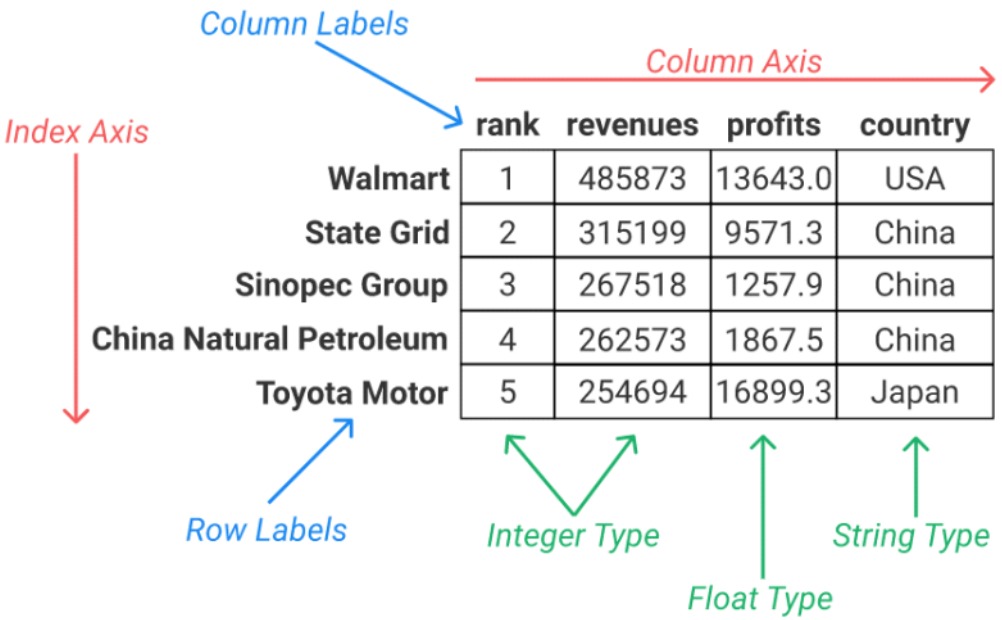
```
specific_order = ['manufacturer', 'model_name', 'category', 'screen_size_inches',
    'screen', 'cpu', 'cpu_manufacturer',  'cpu_speed', 'ram_gb',
    'storage_1_type', 'storage_1_capacity_gb', 'storage_2_type',
    'storage_2_capacity_gb', 'gpu', 'gpu_manufacturer', 'os',
    'os_version', 'weight_kg', 'price_euros']
```

2.0 Creating a Series

2.1 From a Dataframe

Series is typically created from a dataframe. Note that when we create a series from a dataframe, it retains the labels from the dataframe.

For example, from the dataframe below



Let's create a series containing the first row of data

test_series = f500.loc ["Walmart"]

Here is the test_series variable

	Walmart
rank	1
revenues	485873
revenue_change	0.8
profits	13643
assets	198825
profit_change	-7.2
ceo	C. Douglas McMillon
industry	General Merchandisers
sector	Retailing
previous_rank	1
country	USA
hq_location	Bentonville, AR
website	http://www.walmart.com

Notice, how the column labels from the dataframe "f500" became the row labels for the "test_series"

2.2 Creating a series from Scratch

This involves passing a list to the function pd.series()

https://www.geeksforgeeks.org/creating-a-pandas-series-from-lists/

Could also pass a numpy 1D array. Example of this is in 1.0 Sampling in Statistics Page but repeated here:

```
clusters = pd.Series(wnba['Team'].unique()).sample(4, random_state = 0)

sample = pd.DataFrame()

for cluster in clusters:
    data_collected = wnba[wnba['Team'] == cluster]
    sample = sample.append(data_collected)

sampling_error_height = wnba['Height'].mean() - sample['Height'].mean()
sampling_error_age = wnba['Age'].mean() - sample['Age'].mean()
sampling_error_BMI = wnba['BMI'].mean() - sample['BMI'].mean()
sampling_error_points = wnba['PTS'].mean() - sample['PTS'].mean()
```

First line

If we just do: wnba ['Team'].unique () then we get a numpy ndarray

We do pd.series (numpy 1d array) to create a series

Then we randomly select 4 of them

2.3 Mathematical Operations

We can use these operations on a series

- Adding a constant
- Addition, subtraction, multiplication, division