Part 2 — Solutions for Lecture 4

TECH2: Introduction to Programming, Data, and Information Technology

Richard Foltyn
Norwegian School of Economics (NHH)

October 23, 2024

See GitHub repository for notebooks and data:
https://github.com/richardfoltyn/TECH2-H24

Contents

1 Concatenating and merging data

1.1 Concatenation

Your turn.

1. Create a new Series with observations [‘C1’, ‘C2’].
2. Using the previously created Series a and b, concatenate all three objects along the row axis and create a new (unique) index.
3. Repeat the previous step, but now concatenate along the column axis. Assign the column names ‘Column1’, ‘Column2’, and ‘Column3’.

Solution.

[2]:
```
import pandas as pd

# Recreate Series a, b:
# Create first series of 3 observations
a = pd.Series(['A1', 'A2', 'A3'])
# Create second series with 5 observations
b = pd.Series([f'B{i}' for i in range(5)])
```

[3]:
```
# Create Series c
c = pd.Series(['C1', 'C2'])
```

[4]:
```
# Concatenate Series a, b, c and reset the index
s = pd.concat((a, b, c)).reset_index(drop=True)
s
```

[4]:
```
0    A1
1    A2
2    A3
3    B0
4    B1
5    B2
6    B3
7    B4
8    C1
9    C2
dtype: object
```

[5]:
```python
s = pd.concat((a, b, c), axis=1, keys=['Column1', 'Column2', 'Column3'])
s
```

[5]:
```
  Column1 Column2 Column3
0      A1      B0      C1
1      A2      B1      C2
2      A3      B2     NaN
3     NaN      B3     NaN
4     NaN      B4     NaN
```

Your turn. Use the data files located in the folder ../data/FRED to perform the following tasks:

1. Load the data in FRED_monthly_1950.csv and FRED_monthly_1960.csv into two different DataFrames. The files contain monthly macroeconomic time series for the 1950s and 1960s, respectively.

   Hint: Use pd.read_csv(. . . , parse_dates=['DATE']) to automatically parse strings stored in the DATE column as dates.

2. Concatenate these DataFrames along the row dimension to get a total of 240 observations.
3. Set the column DATE as index for the newly created DataFrame.

Solution.

Part (1)

[83]:
```python
# Path to data folder
DATA_PATH = '/home/richard/repos/teaching/TECH2-H24/data/FRED'
```

[84]:
```python
import pandas as pd

# Load data from the 1950s
df1 = pd.read_csv(f'{DATA_PATH}/FRED_monthly_1950.csv', parse_dates=['DATE'])
df1.head(5)
```

[84]:
```
        DATE   CPI  UNRATE  FEDFUNDS  REALRATE  LFPART
0 1950-01-01  23.5     6.5       NaN       NaN    58.9
1 1950-02-01  23.6     6.4       NaN       NaN    58.9
2 1950-03-01  23.6     6.3       NaN       NaN    58.8
3 1950-04-01  23.6     5.8       NaN       NaN    59.2
4 1950-05-01  23.8     5.5       NaN       NaN    59.1
```

[85]:
```python
# Load data from the 1960s
df2 = pd.read_csv(f'{DATA_PATH}/FRED_monthly_1960.csv', parse_dates=['DATE'])
df2.head(5)
```

```
[85]:          DATE   CPI  UNRATE  FEDFUNDS  REALRATE  LFPART
      0 1960-01-01  29.4     5.2       4.0       NaN    59.1
      1 1960-02-01  29.4     4.8       4.0       NaN    59.1
      2 1960-03-01  29.4     5.4       3.8       NaN    58.5
      3 1960-04-01  29.5     5.2       3.9       NaN    59.5
      4 1960-05-01  29.6     5.1       3.8       NaN    59.5
```

Part (2)

```
[86]: # Concatenate data sets along the first dimension (rows)
      df = pd.concat((df1, df2), axis=0)
```

```
[87]: # First half contains data from the 1950s
      df.head(5)
```

```
[87]:          DATE   CPI  UNRATE  FEDFUNDS  REALRATE  LFPART
      0 1950-01-01  23.5     6.5       NaN       NaN    58.9
      1 1950-02-01  23.6     6.4       NaN       NaN    58.9
      2 1950-03-01  23.6     6.3       NaN       NaN    58.8
      3 1950-04-01  23.6     5.8       NaN       NaN    59.2
      4 1950-05-01  23.8     5.5       NaN       NaN    59.1
```

```
[88]: # Second half contains data from the 1960s
      df.tail(5)
```

```
[88]:            DATE   CPI  UNRATE  FEDFUNDS  REALRATE  LFPART
      115 1969-08-01  36.9     3.5       9.2       NaN    60.3
      116 1969-09-01  37.1     3.7       9.2       NaN    60.3
      117 1969-10-01  37.3     3.7       9.0       NaN    60.4
      118 1969-11-01  37.5     3.5       8.8       NaN    60.2
      119 1969-12-01  37.7     3.5       9.0       NaN    60.2
```

Part (3)

Note that the index of the newly created `DataFrame` is not unique:

```
[89]: # Select rows at index 0: returns 2 (!) different rows
      df.loc[0]
```

```
[89]:          DATE   CPI  UNRATE  FEDFUNDS  REALRATE  LFPART
      0 1950-01-01  23.5     6.5       NaN       NaN    58.9
      0 1960-01-01  29.4     5.2       4.0       NaN    59.1
```

```
[90]: # Set Date as new (unique!) index
      df = df.set_index('DATE')
      df.head(10)
```

```
[90]:              CPI  UNRATE  FEDFUNDS  REALRATE  LFPART
      DATE
      1950-01-01  23.5     6.5       NaN       NaN    58.9
      1950-02-01  23.6     6.4       NaN       NaN    58.9
      1950-03-01  23.6     6.3       NaN       NaN    58.8
      1950-04-01  23.6     5.8       NaN       NaN    59.2
      1950-05-01  23.8     5.5       NaN       NaN    59.1
      1950-06-01  23.9     5.4       NaN       NaN    59.4
      1950-07-01  24.1     5.0       NaN       NaN    59.1
      1950-08-01  24.2     4.5       NaN       NaN    59.5
      1950-09-01  24.3     4.4       NaN       NaN    59.2
      1950-10-01  24.5     4.2       NaN       NaN    59.4
```

1.2 Merging and joining data sets

Your turn. Use the data files located in the folder ../data/FRED to perform the following tasks:

1. Load the data in CPI.csv and GDP.csv into two different DataFrames. The files contain monthly data for the Consumer Price Index (CPI) and quarterly data for GDP, respectively.

   Hint: Use pd.read_csv(..., parse_dates=['DATE']) to automatically parse strings stored in the DATE column as dates.

2. Merge the CPI with the GDP time series with merge() using a left join (how='left'). How many observations does the resulting DataFrame have?

3. Merge the CPI with the GDP time series with merge() using an inner join (how='inner'). How many observations does the resulting DataFrame have, and why is this different from the previous case?

Solution.

Part (1)

[91]:
```
# Path to data folder
DATA_PATH = '/home/richard/repos/teaching/TECH2-H24/data/FRED'
```

[92]:
```
import pandas as pd

cpi = pd.read_csv(f'{DATA_PATH}/CPI.csv', parse_dates=['DATE'])
cpi.head(5)
```

[92]:
```
        DATE   CPI
0 1947-01-01  21.5
1 1947-02-01  21.6
2 1947-03-01  22.0
3 1947-04-01  22.0
4 1947-05-01  22.0
```

[93]:
```
gdp = pd.read_csv(f'{DATA_PATH}/GDP.csv', parse_dates=['DATE'])
gdp.head(5)
```

[93]:
```
        DATE     GDP
0 1947-01-01  2182.7
1 1947-04-01  2176.9
2 1947-07-01  2172.4
3 1947-10-01  2206.5
4 1948-01-01  2239.7
```

Part (2)

[94]:
```
# Merge so that left DataFrame determines resulting index
df = pd.merge(cpi, gdp, on='DATE', how='left')
df.head(12)
```

[94]:
```
         DATE   CPI     GDP
0  1947-01-01  21.5  2182.7
```

```
1   1947-02-01  21.6     NaN
2   1947-03-01  22.0     NaN
3   1947-04-01  22.0  2176.9
4   1947-05-01  22.0     NaN
5   1947-06-01  22.1     NaN
6   1947-07-01  22.2  2172.4
7   1947-08-01  22.4     NaN
8   1947-09-01  22.8     NaN
9   1947-10-01  22.9  2206.5
10  1947-11-01  23.1     NaN
11  1947-12-01  23.4     NaN
```

[95]:
```
# Number of observations
N = len(df)
print(f'Number of observations with left join: {N:,d}')
```

```
Number of observations with left join: 932
```

Part (3)

[96]:
```
# Drop columns with missing observations in GDP
df = pd.merge(cpi, gdp, on='DATE', how='inner')
df.head(12)
```

[96]:
```
          DATE   CPI     GDP
0   1947-01-01  21.5  2182.7
1   1947-04-01  22.0  2176.9
2   1947-07-01  22.2  2172.4
3   1947-10-01  22.9  2206.5
4   1948-01-01  23.7  2239.7
5   1948-04-01  23.8  2276.7
6   1948-07-01  24.4  2289.8
7   1948-10-01  24.3  2292.4
8   1949-01-01  24.0  2260.8
9   1949-04-01  23.9  2253.1
10  1949-07-01  23.7  2276.4
11  1949-10-01  23.7  2257.4
```

[97]:
```
# Number of observations
N = len(df)
print(f'Number of observations with inner join: {N:,d}')
```

```
Number of observations with inner join: 310
```

The inner join drops all dates from `cpi` which are not present in the `gdp` DataFrame, hence the number of rows in the merged DataFrame is only a third of the original data (since the GDP data is quarterly).

Your turn. Use the data files located in the folder ../data/FRED to perform the following tasks:

1. Load the data in CPI.csv and GDP.csv into two different DataFrames. The files contain monthly data for the Consumer Price Index (CPI) and quarterly data for GDP, respectively.

   Hint: Use pd.read_csv(. . . , parse_dates=['DATE']) to automatically parse strings stored in the DATE column as dates.

2. Set the DATE column as the index for each of the two DataFrames.
3. Merge the CPI with the GDP time series with join(). Do this with both a left and an inner join.

Solution.

Part (1)

```
[98]: # Path to data folder
      DATA_PATH = '/home/richard/repos/teaching/TECH2-H24/data/FRED'
```

```
[99]: import pandas as pd

      cpi = pd.read_csv(f'{DATA_PATH}/CPI.csv', parse_dates=['DATE'])
      # Alternatively, we can set the index directly when loading the data
      # cpi = pd.read_csv(f'{DATA_PATH}/CPI.csv', parse_dates=['DATE'], index_col='DATE')
      cpi.head(5)
```

```
[99]:         DATE   CPI
      0 1947-01-01  21.5
      1 1947-02-01  21.6
      2 1947-03-01  22.0
      3 1947-04-01  22.0
      4 1947-05-01  22.0
```

```
[100]: gdp = pd.read_csv(f'{DATA_PATH}/GDP.csv', parse_dates=['DATE'])
       # Alternatively, we can set the index directly when loading the data
       # gdp = pd.read_csv(f'{DATA_PATH}/GDP.csv', parse_dates=['DATE'], index_col='DATE')
       gdp.head(5)
```

```
[100]:         DATE     GDP
       0 1947-01-01  2182.7
       1 1947-04-01  2176.9
       2 1947-07-01  2172.4
       3 1947-10-01  2206.5
       4 1948-01-01  2239.7
```

Part (2)

If we didn't specify the index columns using `index_col` as an argument to `pd.read_csv()`, we can set the index after loading the data.

```
[101]: # Set DATE column as index
       cpi = cpi.set_index('DATE')
       gdp = gdp.set_index('DATE')
```

Part (3)

```
[102]: # Perform left join (the default)
       df = cpi.join(gdp)
       df.head(10)
```

```
[102]:              CPI     GDP
       DATE
       1947-01-01  21.5  2182.7
       1947-02-01  21.6     NaN
       1947-03-01  22.0     NaN
       1947-04-01  22.0  2176.9
       1947-05-01  22.0     NaN
       1947-06-01  22.1     NaN
       1947-07-01  22.2  2172.4
```

```
1947-08-01  22.4     NaN
1947-09-01  22.8     NaN
1947-10-01  22.9  2206.5
```

[103]:
```python
# Perform inner join
df = cpi.join(gdp, how='inner')
df.head(10)
```

[103]:
```
             CPI     GDP
DATE
1947-01-01  21.5  2182.7
1947-04-01  22.0  2176.9
1947-07-01  22.2  2172.4
1947-10-01  22.9  2206.5
1948-01-01  23.7  2239.7
1948-04-01  23.8  2276.7
1948-07-01  24.4  2289.8
1948-10-01  24.3  2292.4
1949-01-01  24.0  2260.8
1949-04-01  23.9  2253.1
```

2 Dealing with missing values

Your turn. Use the data files located in the folder ../data/FRED to perform the following tasks:

1. Load the data in CPI.csv and GDP.csv into two different DataFrames. The files contain monthly data for the Consumer Price Index (CPI) and quarterly data for GDP, respectively.

   Hint: Use pd.read_csv(..., parse_dates=['DATE']) to automatically parse strings stored in the DATE column as dates.

2. Merge the CPI with the GDP time series with merge() using a left join. This creates missing values in the GDP column.

3. Impute the missing GDP values using interpolate() and replace the missing values in column GDP.

Solution.

Part (1)

[2]:
```python
# Path to data folder
DATA_PATH = '/home/richard/repos/teaching/TECH2-H24/data/FRED'
```

[3]:
```python
import pandas as pd

# Load CPI data
cpi = pd.read_csv(f'{DATA_PATH}/CPI.csv', parse_dates=['DATE'])

# Load GDP data
gdp = pd.read_csv(f'{DATA_PATH}/GDP.csv', parse_dates=['DATE'])
```

Part (2)

[7]:
```
# Merge CPI and GDP into a single DataFrame, use keys from CPI
df = pd.merge(cpi, gdp, how='left')

# Print first 12 months
df.head(12)
```

[7]:
```
         DATE   CPI     GDP
0  1947-01-01  21.5  2182.7
1  1947-02-01  21.6     NaN
2  1947-03-01  22.0     NaN
3  1947-04-01  22.0  2176.9
4  1947-05-01  22.0     NaN
5  1947-06-01  22.1     NaN
6  1947-07-01  22.2  2172.4
7  1947-08-01  22.4     NaN
8  1947-09-01  22.8     NaN
9  1947-10-01  22.9  2206.5
10 1947-11-01  23.1     NaN
11 1947-12-01  23.4     NaN
```

Since GDP data is available on quarterly frequency, only every third month contains non-missing values.

Part (3)

[12]:
```
# Linearly interpolate missing value
df['GDP'] = df['GDP'].interpolate(method='linear')

# Print first 12 months to confirm that missing values are gone
df.head(12)
```

[12]:
```
         DATE   CPI          GDP
0  1947-01-01  21.5  2182.700000
1  1947-02-01  21.6  2180.766667
2  1947-03-01  22.0  2178.833333
3  1947-04-01  22.0  2176.900000
4  1947-05-01  22.0  2175.400000
5  1947-06-01  22.1  2173.900000
6  1947-07-01  22.2  2172.400000
7  1947-08-01  22.4  2183.766667
8  1947-09-01  22.8  2195.133333
9  1947-10-01  22.9  2206.500000
10 1947-11-01  23.1  2217.566667
11 1947-12-01  23.4  2228.633333
```