Basics

March 2, 2025

1 Guided Project: Storytelling Data Visualization on Exchange Rates

The dataset we'll use describes Euro daily exchange rates between 1999 and 2021. The euro (symbolized with €) is the official currency in most of the countries of the European Union.

If the exchange rate of the euro to the US dollar is 1.5, you get 1.5 US dollars if you pay 1.0 euro (one euro has more value than one US dollar at this exchange rate).

[3]:
```
import pandas as pd

exchange_rates = pd.read_csv('euro-daily-hist_1999_2020.csv')
exchange_rates.head()
```

[3]:
```
  Period\Unit: [Australian dollar ] [Bulgarian lev ] [Brazilian real ]  \
0   2021-01-08               1.5758           1.9558            6.5748
1   2021-01-07               1.5836           1.9558            6.5172
2   2021-01-06               1.5824           1.9558            6.5119
3   2021-01-05               1.5927           1.9558            6.5517
4   2021-01-04               1.5928           1.9558            6.3241

  [Canadian dollar ] [Swiss franc ] [Chinese yuan renminbi ] [Cypriot pound ]  \
0             1.5543         1.0827                   7.9184              NaN
1             1.5601         1.0833                   7.9392              NaN
2             1.5640         1.0821                   7.9653              NaN
3             1.5651         1.0803                   7.9315              NaN
4             1.5621         1.0811                   7.9484              NaN

  [Czech koruna ] [Danish krone ]  … [Romanian leu ] [Russian rouble ]  \
0          26.163          7.4369  …          4.8708           90.8000
1          26.147          7.4392  …          4.8712           91.2000
2          26.145          7.4393  …          4.8720           90.8175
3          26.227          7.4387  …          4.8721           91.6715
4          26.141          7.4379  …          4.8713           90.3420

  [Swedish krona ] [Singapore dollar ] [Slovenian tolar ] [Slovak koruna ]  \
0          10.0510              1.6228                NaN              NaN
1          10.0575              1.6253                NaN              NaN
```

```
2          10.0653              1.6246                NaN              NaN
3          10.0570              1.6180                NaN              NaN
4          10.0895              1.6198                NaN              NaN

  [Thai baht ] [Turkish lira ] [US dollar ]  [South African rand ]
0      36.8480          9.0146       1.2250                18.7212
1      36.8590          8.9987       1.2276                18.7919
2      36.9210          9.0554       1.2338                18.5123
3      36.7760          9.0694       1.2271                18.4194
4      36.7280          9.0579       1.2296                17.9214

[5 rows x 41 columns]
```

[5]:
```
exchange_rates.tail()
```

[5]:
```
     Period\Unit: [Australian dollar ] [Bulgarian lev ] [Brazilian real ]  \
5694   1999-01-08               1.8406              NaN               NaN
5695   1999-01-07               1.8474              NaN               NaN
5696   1999-01-06               1.8820              NaN               NaN
5697   1999-01-05               1.8944              NaN               NaN
5698   1999-01-04               1.9100              NaN               NaN

     [Canadian dollar ] [Swiss franc ] [Chinese yuan renminbi ]  \
5694             1.7643         1.6138                      NaN
5695             1.7602         1.6165                      NaN
5696             1.7711         1.6116                      NaN
5697             1.7965         1.6123                      NaN
5698             1.8004         1.6168                      NaN

     [Cypriot pound ] [Czech koruna ] [Danish krone ]  … [Romanian leu ]  \
5694          0.58187          34.938          7.4433  …          1.3143
5695          0.58187          34.886          7.4431  …          1.3092
5696          0.58200          34.850          7.4452  …          1.3168
5697          0.58230          34.917          7.4495  …          1.3168
5698          0.58231          35.107          7.4501  …          1.3111

     [Russian rouble ] [Swedish krona ] [Singapore dollar ]  \
5694           27.2075           9.1650              1.9537
5695           26.9876           9.1800              1.9436
5696           27.4315           9.3050              1.9699
5697           26.5876           9.4025              1.9655
5698           25.2875           9.4696              1.9554

     [Slovenian tolar ] [Slovak koruna ] [Thai baht ] [Turkish lira ]  \
5694           188.8400           42.560      42.5590          0.3718
5695           188.8000           42.765      42.1678          0.3701
5696           188.7000           42.778      42.6949          0.3722
```

```
5697           188.7750           42.848      42.5048          0.3728
5698           189.0450           42.991      42.6799          0.3723

     [US dollar ]  [South African rand ]
5694       1.1659                6.7855
5695       1.1632                6.8283
5696       1.1743                6.7307
5697       1.1790                6.7975
5698       1.1789                6.9358

[5 rows x 41 columns]
```

[6]:
```
exchange_rates.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5699 entries, 0 to 5698
Data columns (total 41 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Period\Unit:               5699 non-null   object
 1   [Australian dollar ]       5699 non-null   object
 2   [Bulgarian lev ]           5297 non-null   object
 3   [Brazilian real ]          5431 non-null   object
 4   [Canadian dollar ]         5699 non-null   object
 5   [Swiss franc ]             5699 non-null   object
 6   [Chinese yuan renminbi ]   5431 non-null   object
 7   [Cypriot pound ]           2346 non-null   object
 8   [Czech koruna ]            5699 non-null   object
 9   [Danish krone ]            5699 non-null   object
 10  [Estonian kroon ]          3130 non-null   object
 11  [UK pound sterling ]       5699 non-null   object
 12  [Greek drachma ]           520 non-null    object
 13  [Hong Kong dollar ]        5699 non-null   object
 14  [Croatian kuna ]           5431 non-null   object
 15  [Hungarian forint ]        5699 non-null   object
 16  [Indonesian rupiah ]       5699 non-null   object
 17  [Israeli shekel ]          5431 non-null   object
 18  [Indian rupee ]            5431 non-null   object
 19  [Iceland krona ]           3292 non-null   float64
 20  [Japanese yen ]            5699 non-null   object
 21  [Korean won ]              5699 non-null   object
 22  [Lithuanian litas ]        4159 non-null   object
 23  [Latvian lats ]            3904 non-null   object
 24  [Maltese lira ]            2346 non-null   object
 25  [Mexican peso ]            5699 non-null   object
 26  [Malaysian ringgit ]       5699 non-null   object
 27  [Norwegian krone ]         5699 non-null   object
```

```
 28  [New Zealand dollar ]     5699 non-null   object
 29  [Philippine peso ]        5699 non-null   object
 30  [Polish zloty ]           5699 non-null   object
 31  [Romanian leu ]           5637 non-null   float64
 32  [Russian rouble ]         5699 non-null   object
 33  [Swedish krona ]          5699 non-null   object
 34  [Singapore dollar ]       5699 non-null   object
 35  [Slovenian tolar ]        2085 non-null   object
 36  [Slovak koruna ]          2608 non-null   object
 37  [Thai baht ]              5699 non-null   object
 38  [Turkish lira ]           5637 non-null   float64
 39  [US dollar ]              5699 non-null   object
 40  [South African rand ]     5699 non-null   object
dtypes: float64(3), object(38)
memory usage: 1.8+ MB
```

[7]:
```
# Clean data
exchange_rates.rename(columns={'[US dollar ]': 'US_dollar',
                               'Period\\Unit:': 'Time'},
                      inplace=True)
exchange_rates['Time'] = pd.to_datetime(exchange_rates['Time'])
exchange_rates.sort_values('Time', inplace=True)
exchange_rates.reset_index(drop=True, inplace=True)
exchange_rates.head()
```

[7]:

	Time	[Australian dollar]	[Bulgarian lev]	[Brazilian real]	\
0	1999-01-04	1.9100	NaN	NaN	
1	1999-01-05	1.8944	NaN	NaN	
2	1999-01-06	1.8820	NaN	NaN	
3	1999-01-07	1.8474	NaN	NaN	
4	1999-01-08	1.8406	NaN	NaN	

	[Canadian dollar]	[Swiss franc]	[Chinese yuan renminbi]	[Cypriot pound]	\
0	1.8004	1.6168	NaN	0.58231	
1	1.7965	1.6123	NaN	0.58230	
2	1.7711	1.6116	NaN	0.58200	
3	1.7602	1.6165	NaN	0.58187	
4	1.7643	1.6138	NaN	0.58187	

	[Czech koruna]	[Danish krone]	…	[Romanian leu]	[Russian rouble]	\
0	35.107	7.4501	…	1.3111	25.2875	
1	34.917	7.4495	…	1.3168	26.5876	
2	34.850	7.4452	…	1.3168	27.4315	
3	34.886	7.4431	…	1.3092	26.9876	
4	34.938	7.4433	…	1.3143	27.2075	

	[Swedish krona]	[Singapore dollar]	[Slovenian tolar]	[Slovak koruna]	\

```
0          9.4696              1.9554          189.0450          42.991
1          9.4025              1.9655          188.7750          42.848
2          9.3050              1.9699          188.7000          42.778
3          9.1800              1.9436          188.8000          42.765
4          9.1650              1.9537          188.8400          42.560

  [Thai baht ] [Turkish lira ] US_dollar  [South African rand ]
0      42.6799          0.3723    1.1789                6.9358
1      42.5048          0.3728    1.1790                6.7975
2      42.6949          0.3722    1.1743                6.7307
3      42.1678          0.3701    1.1632                6.8283
4      42.5590          0.3718    1.1659                6.7855

[5 rows x 41 columns]
```

[9]:
```
euro_to_dollar = exchange_rates[['Time','US_dollar']].copy()
euro_to_dollar['US_dollar'].value_counts()
```

[9]:
```
US_dollar
-         62
1.2276     9
1.1215     8
1.1305     7
1.0867     6
          ..
1.1869     1
1.1752     1
1.1770     1
1.1750     1
1.1821     1
Name: count, Length: 3528, dtype: int64
```

[13]:
```
euro_to_dollar = euro_to_dollar[euro_to_dollar['US_dollar'] != '-']
euro_to_dollar['US_dollar'] = euro_to_dollar['US_dollar'].astype(float)
euro_to_dollar.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5637 entries, 0 to 5698
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Time       5637 non-null   datetime64[ns]
 1   US_dollar  5637 non-null   float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 132.1 KB
```

[20]:
```
import matplotlib.pyplot as plt
euro_to_dollar['rolling_mean']= euro_to_dollar['US_dollar'].rolling(30).mean()
euro_to_dollar
```

[20]:

	Time	US_dollar	rolling_mean
0	1999-01-04	1.1789	NaN
1	1999-01-05	1.1790	NaN
2	1999-01-06	1.1743	NaN
3	1999-01-07	1.1632	NaN
4	1999-01-08	1.1659	NaN
…	…	…	…
5694	2021-01-04	1.2296	1.211170
5695	2021-01-05	1.2271	1.212530
5696	2021-01-06	1.2338	1.213987
5697	2021-01-07	1.2276	1.215357
5698	2021-01-08	1.2250	1.216557

[5637 rows x 3 columns]

[23]:
```
bush_obama_trump = euro_to_dollar.copy()[(euro_to_dollar['Time'].dt.year>=2001)␣
 ↪& (euro_to_dollar['Time'].dt.year<2021)]
bush = bush_obama_trump[bush_obama_trump['Time'].dt.year<2009]
obama = bush_obama_trump[(bush_obama_trump['Time'].dt.year>=2009) &␣
 ↪(bush_obama_trump['Time'].dt.year<2017)]
trump =  bush_obama_trump[bush_obama_trump['Time'].dt.year>=2017]
```

2 The Euro changes rates through out 3 US presidents

[155]:
```
import matplotlib.pyplot as plt
import matplotlib.style as style

style.use('fivethirtyeight')

plt.figure(figsize=(15, 6), dpi=100)

ax1 = plt.subplot(2, 3, 1)
ax2 = plt.subplot(2, 3, 2)
ax3 = plt.subplot(2, 3, 3)
ax4 = plt.subplot(2, 1, 2)

axes = [ax1, ax2, ax3, ax4]
presidents = [bush, obama, trump]
national_color = ['green', 'orange', 'blue']

# Design basic for each graph
for a, pre, c in zip(axes, presidents, national_color):
```

```
    a.set_ylim(0.8, 1.7)
    a.set_yticks([1.0, 1.2, 1.4, 1.6])
    a.set_yticklabels(['1.0', '1.2', '1.4', '1.6'], alpha=0.3)

    # Plot data
    a.plot(pre['Time'], pre['rolling_mean'], color =c)
    a.grid(alpha=0.5)

    # Pick min/max year
    min_year = pre['Time'].dt.year.min()
    max_year = pre['Time'].dt.year.max()

    # Convert to datetime
    year_labels = list(range(min_year, max_year + 2, 1))  # Năm hiển thị
    year_ticks = pd.to_datetime([f"{y}-01-01" for y in year_labels])  # Chuyển␣
 ↪thành datetime

    # Cập nhật trục X
    a.set_xlim(pre['Time'].min(), pre['Time'].max())  # Đặt giới hạn trục X
    a.set_xticks(year_ticks)
    a.set_xticklabels(year_labels, alpha=0.5)

# Chỉnh các bảng
#Ax1
ax1.set_xticklabels(['2001','', '2003', '', '2005','', '2007','', '2009']) # Do␣
 ↪set_xticks và set_xticklabels phải cùng số lượng giá trị
ax1.text(12450,2.0,'BUSH', weight ='bold', size =20, color='green')
ax1.text(12250,1.9,'(2001-2009)', weight ='bold', size =15, color='green')
#Ax2
ax2.set_xticklabels(['2009','', '2011', '', '2013','', '2015','', '2017'])
ax2.text(15300,2.0,'OBAMA', weight ='bold', size =20, color='orange')
ax2.text(15200,1.9,'(2009-2017)', weight ='bold', size =15, color='orange')

#Ax3
ax3.text(17670,2.0,'TRUMP', weight ='bold', size =20, color='blue')
ax3.text(17600,1.9,'(2017-2021)', weight ='bold', size =15, color='blue')

#Ax4
ax4.plot(bush['Time'],bush['rolling_mean'], color ='green')
ax4.plot(obama['Time'],obama['rolling_mean'], color ='orange')
ax4.plot(trump['Time'],trump['rolling_mean'], color ='blue')
ax4.set_xticklabels([])
#Add the graph title and subtitle
ax1.text(11050, 2.5, 'EURO-USD rate averaged 1.22, under the last three US␣
 ↪presidents', weight='bold', fontsize=25)
ax2.text(10500, 2.22, '''EURO-USD exchange rates under George W. Bush␣
 ↪(2001-2009), Barack Obama (2009-2017),
```

```
and Donald Trump (2017-2021)''', fontsize=20)

#Add footage
ax4.text(10500, 0.6, 'DATAQUEST' + ' '*110 + 'Source: European Central Bank',
      color='#f0f0f0', backgroundcolor='#4d4d4d',
      fontsize=18)
plt.show()
```