VISUALIZING TIME SERIES DATA IN PYTHON

Working with more than one time series

Thomas Vincent
Head of Data Science, Getty Images

Working with multiple time series

An isolated time series

```
date,ts1
1949-01,112
1949-02,118
1949-03,132
```

A file with multiple time series

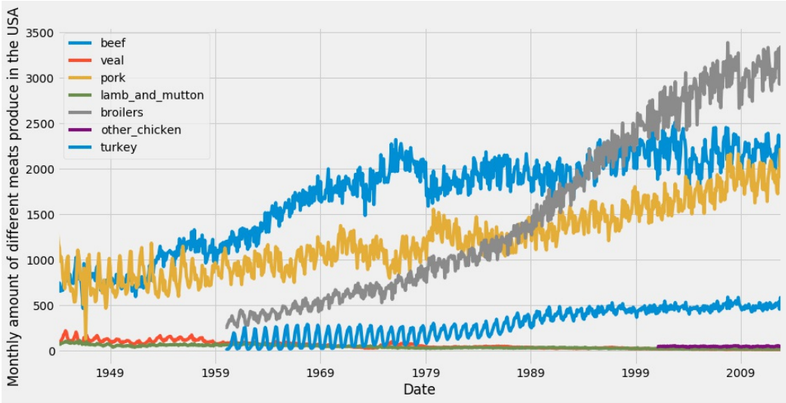
```
date,ts1,ts2,ts3,ts4,ts5,ts6,ts7
2012-01-01,2113.8,10.4,1987.0,12.1,3091.8,43.2,476.7
2012-02-01,2009.0,9.8,1882.9,12.3,2954.0,38.8,466.8
2012-03-01,2159.8,10.0,1987.9,14.2,3043.7,40.1,502.1
```

The Meat production dataset

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

In [2]: meat = pd.read_csv("meat.csv")

In [3]: print(meat.head(5))

         date   beef   veal    pork  lamb_and_mutton  broilers
0  1944-01-01  751.0   85.0  1280.0             89.0       NaN
1  1944-02-01  713.0   77.0  1169.0             72.0       NaN
2  1944-03-01  741.0   90.0  1128.0             75.0       NaN
3  1944-04-01  650.0   89.0   978.0             66.0       NaN
4  1944-05-01  681.0  106.0  1029.0             78.0       NaN

   other_chicken  turkey
0            NaN     NaN
1            NaN     NaN
2            NaN     NaN
3            NaN     NaN
4            NaN     NaN
```

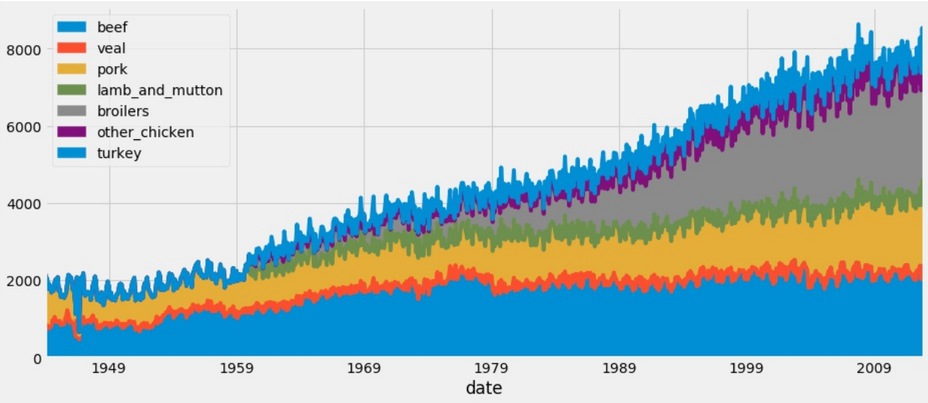
Summarizing and plotting multiple time series

```
In [1]: import matplotlib.pyplot as plt

In [2]: plt.style.use('fivethirtyeight')

In [3]: ax = df.plot(figsize=(12, 4),
                     fontsize=14)

In [4]: plt.show()
```



Area charts

```
In [1]: import matplotlib.pyplot as plt

In [2]: plt.style.use('fivethirtyeight')

In [3]: ax = df.plot.area(figsize=(12, 4), fontsize=14)

In [4]: plt.show()
```



VISUALIZING TIME SERIES DATA IN PYTHON

Let's practice!

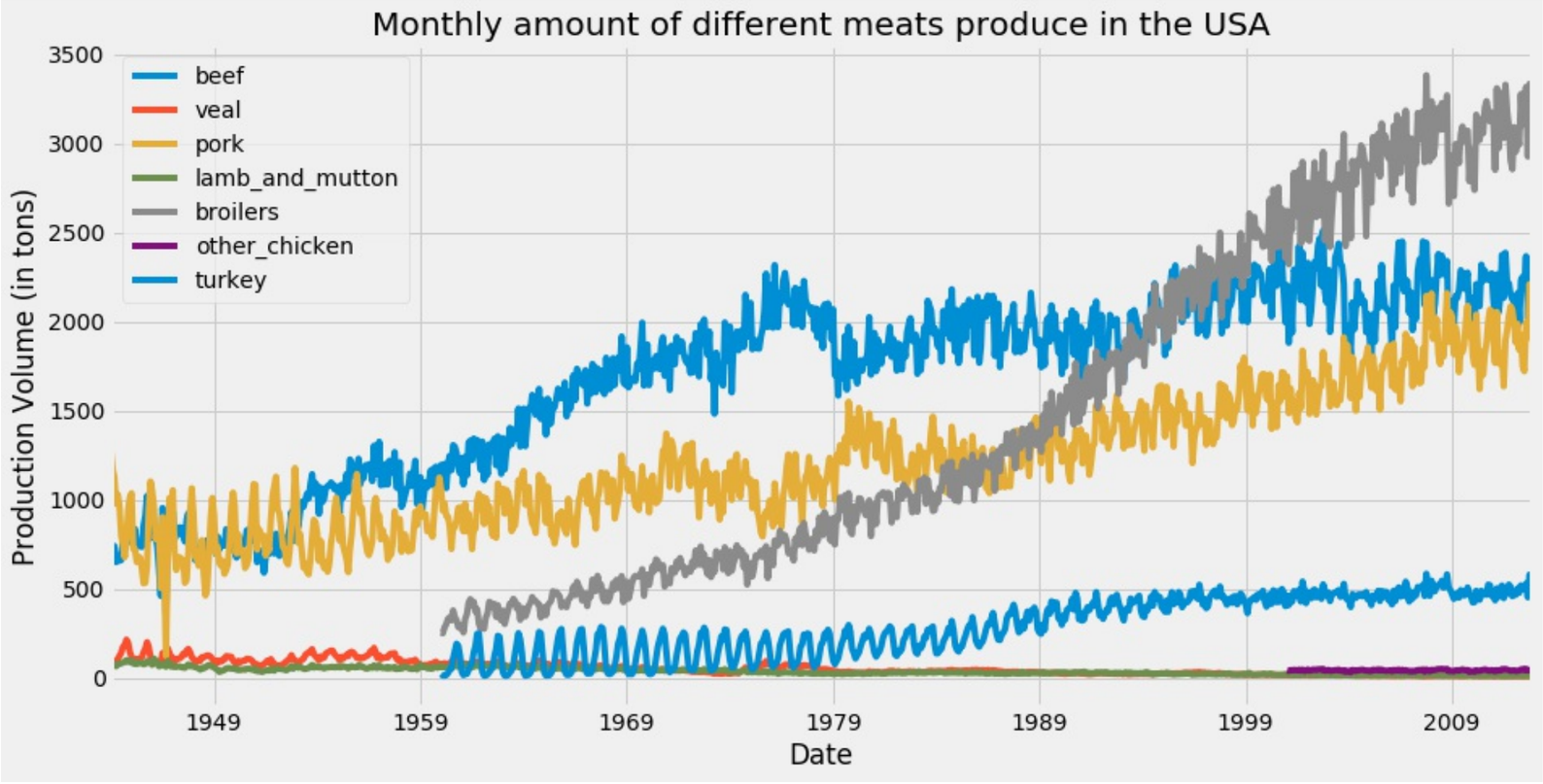
VISUALIZING TIME SERIES DATA IN PYTHON

Plot multiple time series

Thomas Vincent
Head of Data Science, Getty Images

Clarity is key

In this plot, the default matplotlib color scheme assigns the same color to the beef and turkey time series.

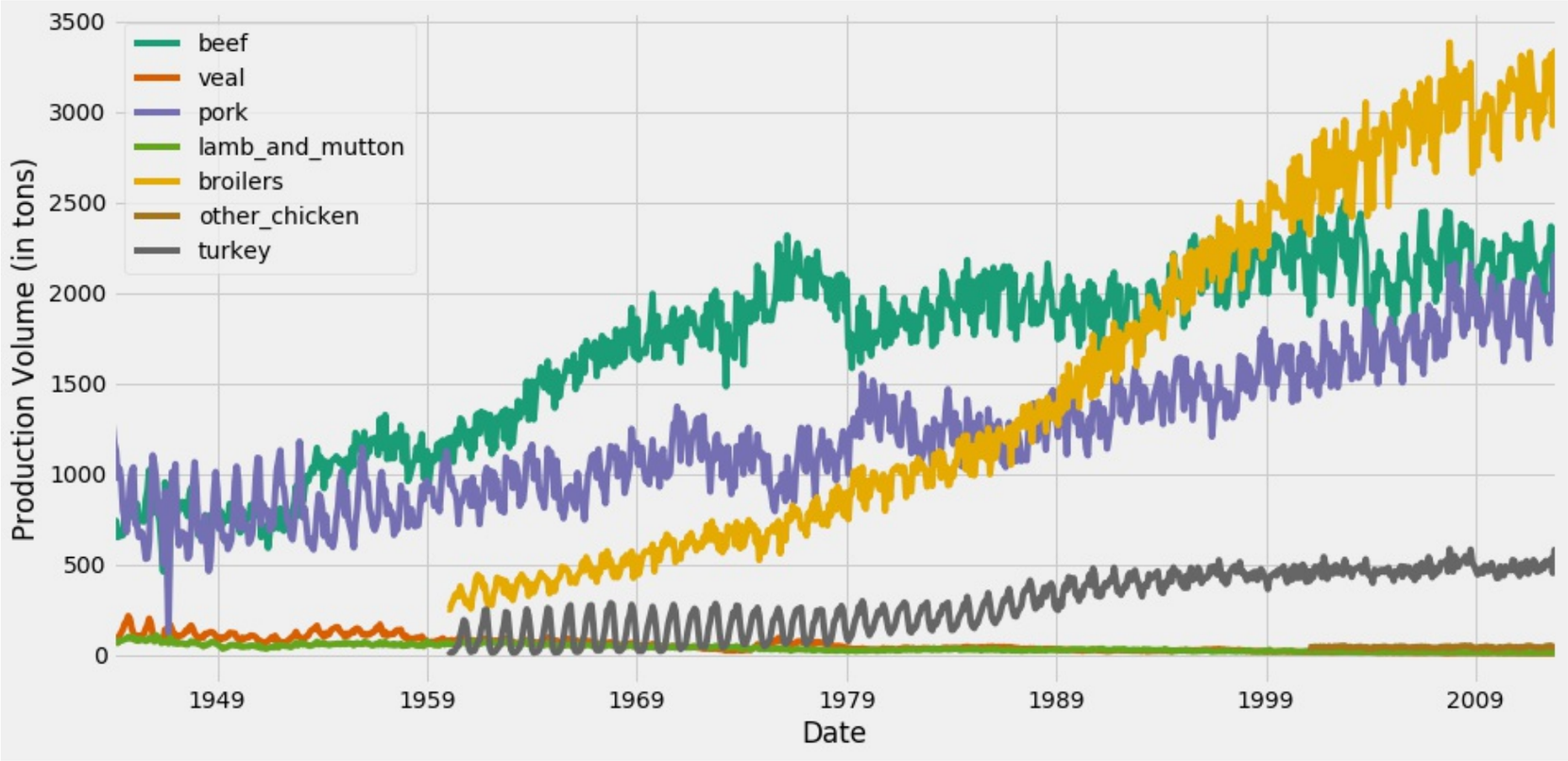


The colormap argument

```
In [1]: ax = df.plot(colormap='Dark2', figsize=(14, 7))

In [2]: ax.set_xlabel('Date')

In [3]: ax.set_ylabel('Production Volume (in tons)')

In [4]: plt.show()
```

For the full set of available colormaps, click here.

Changing line colors with the colormap argument

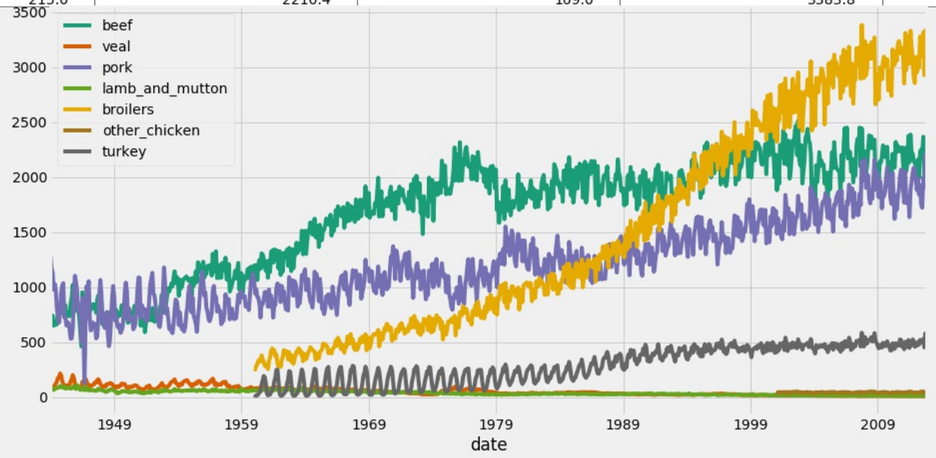


Enhancing your plot with information

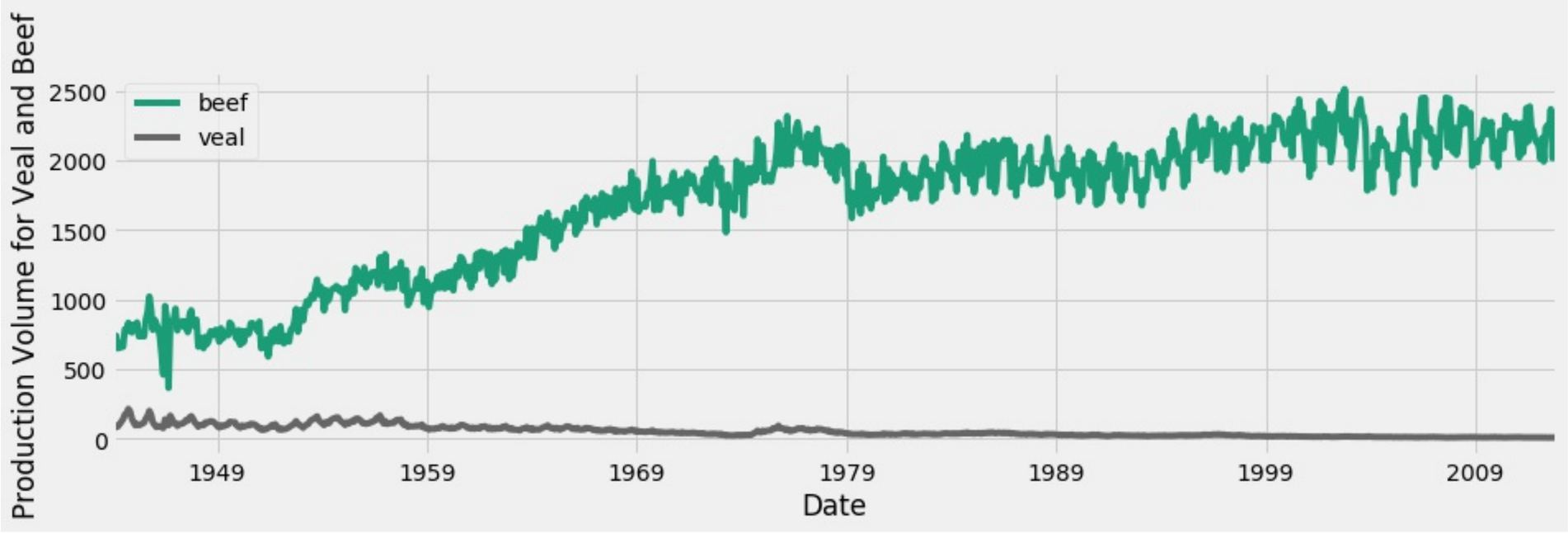
```
In [1]: ax = df.plot(colormap='Dark2', figsize=(14, 7))

In [2]: df_summary = df.describe()

# Specify values of cells in the table
In [3]: ax.table(cellText=df_summary.values,
           # Specify width of the table
           colWidths=[0.3]*len(df.columns),
           # Specify row labels
           rowLabels=df_summary.index,
           # Specify column labels
           colLabels=df_summary.columns,
           # Specify location of the table
           loc='top')

In [4]: plt.show()
```

Adding Statistical summaries to your plots

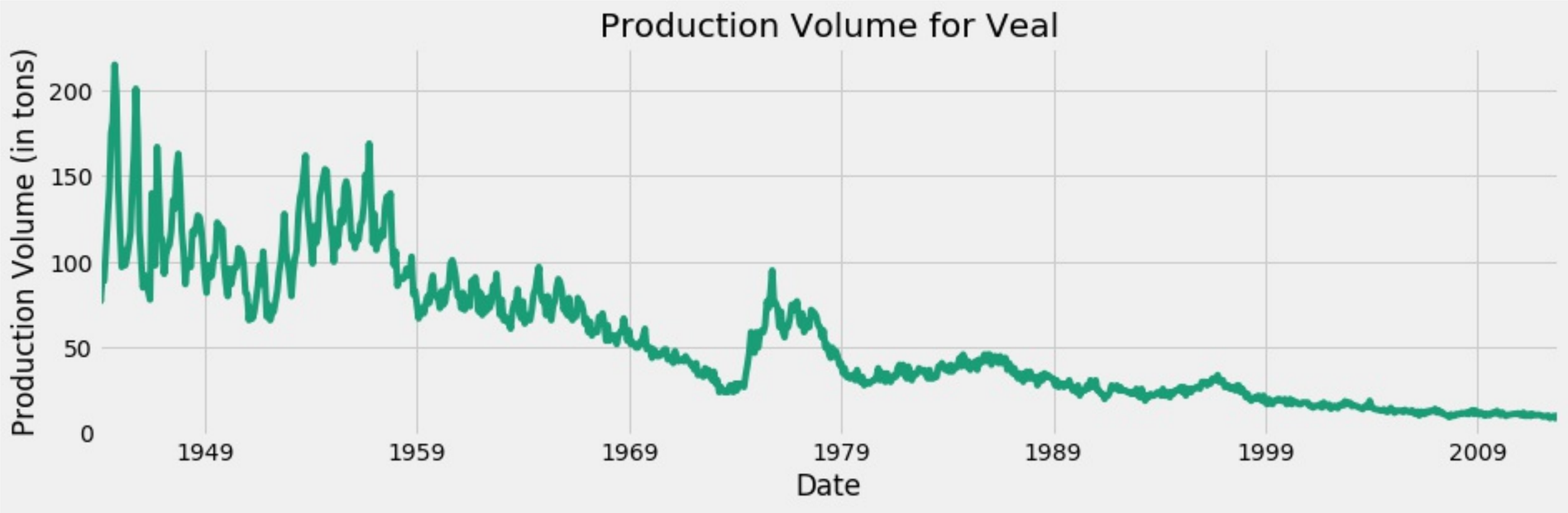
	beef	veal	pork	lamb_and_mutton	broilers	other_chicken	turkey
count	827.0	827.0	827.0	827.0	635.0	143.0	635.0
mean	1683.46336155	54.1985489722	1211.68379686	38.3607013301	1516.58251969	43.0335664336	292.814645669
std	501.698479574	39.0628037539	371.311802273	19.6243398061	963.012101054	3.86714117313	162.482637819
min	366.0	8.8	124.0	10.9	250.9	32.3	12.4
25%	1231.5	24.0	934.5	23.0	636.35	40.2	154.15
50%	1853.0	40.0	1156.0	31.0	1211.3	43.4	278.3
75%	2070.0	79.0	1466.0	55.0	2426.65	45.65	449.15
max	2512.0	215.0	2210.4	109.0	3383.8	51.1	585.1



Dealing with different scales

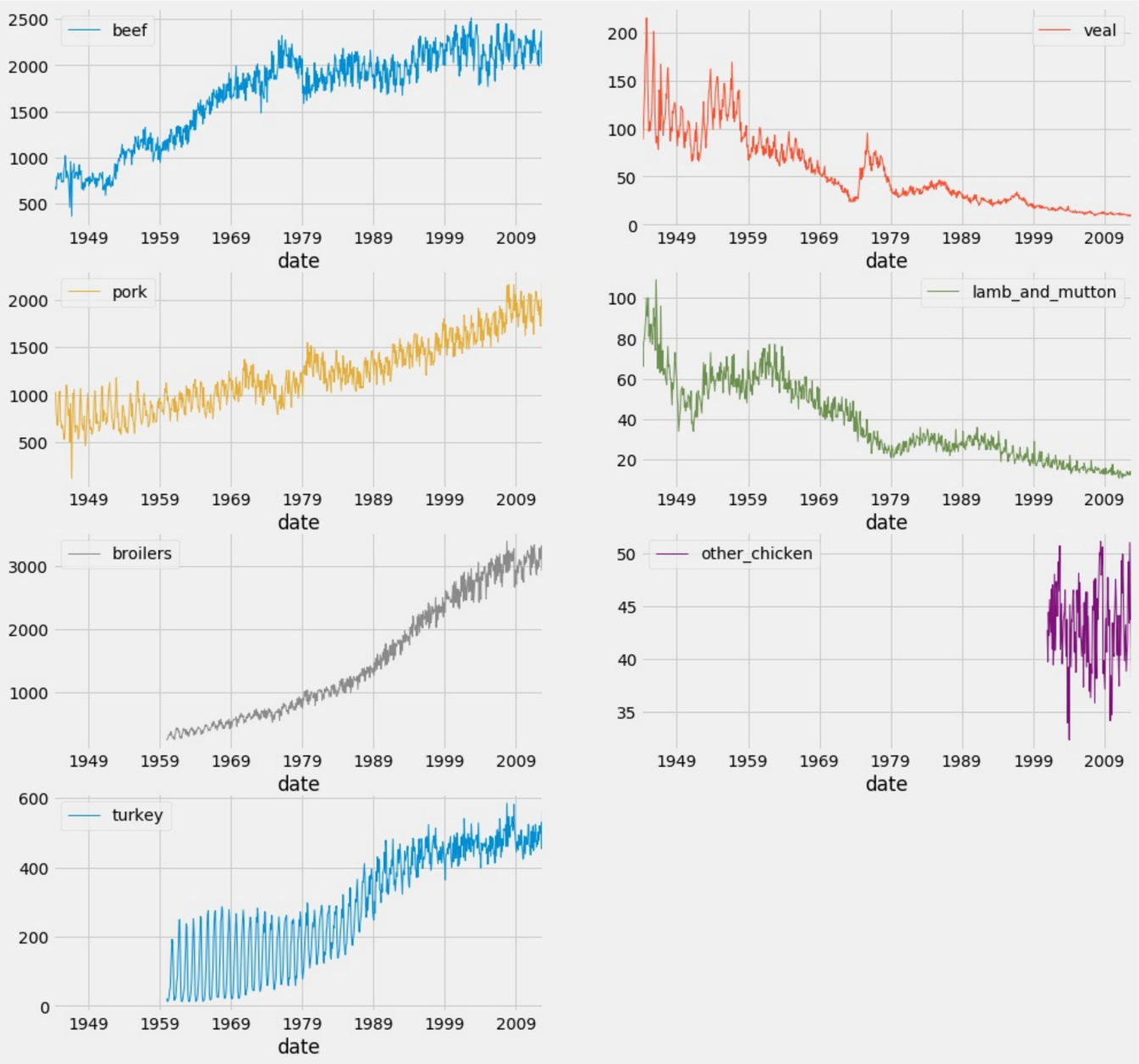


Only veal


Production Volume for Veal
Production Volume (in tons)
200
150
100
50
0
1949
1959
1969
1979
1989
1999
2009
Date

Facet plots

```
In [1]: df.plot(subplots=True,
                linewidth=0.5,
                layout=(2, 4),
                figsize=(16, 10),
                sharex=False,
                sharey=False)

In [2]: plt.show()
```


beef
2500
2000
1500
1000
500
1949
1959
1969
1979
1989
1999
2009
date
veal
200
150
100
50
0
pork
lamb_and_mutton
100
80
60
40
20
broilers
3000
other_chicken
50
45
40
35
turkey
600
400
200
0

VISUALIZING TIME SERIES DATA IN PYTHON

Time for some action!

VISUALIZING TIME SERIES DATA IN PYTHON

Find relationships between multiple time series

Thomas Vincent
Head of Data Science, Getty Images

Correlations between two variables

- In the field of Statistics, the correlation coefficient is a measure used to determine the strength or lack of relationship between two variables:
  - Pearson's coefficient can be used to compute the correlation coefficient between variables for which the relationship is thought to be linear
  - Kendall Tau or Spearman rank can be used to compute the correlation coefficient between variables for which the relationship is thought to be non-linear

Compute correlations

```
In [1]: from scipy.stats.stats import pearsonr

In [2]: from scipy.stats.stats import spearmanr

In [3]: from scipy.stats.stats import kendalltau

In [4]: x = [1, 2, 4, 7]

In [5]: y = [1, 3, 4, 8]

In [6]: pearsonr(x, y)
SpearmanrResult(correlation=0.9843, pvalue=0.01569)

In [7]: spearmanr(x, y)
SpearmanrResult(correlation=1.0, pvalue=0.0)

In [8]: kendalltau(x, y)
KendalltauResult(correlation=1.0, pvalue=0.0415)
```

What is a correlation matrix?

- When computing the correlation coefficient between more than two variables, you obtain a correlation matrix
  - Range: [-1, 1]
  - 0: no relationship
  - 1: strong positive relationship
  - -1: strong negative relationship

What is a correlation matrix?

- A correlation matrix is always "symmetric"
- The diagonal values will always be equal to 1

```
     x     y     z
x  1.00 -0.46  0.49
y -0.46  1.00 -0.61
z  0.49 -0.61  1.00
```

Computing Correlation Matrices with Pandas

```
In [1]: corr_p = meat[['beef', 'veal', 'turkey']].corr(method='pearson')

In [2]: print(corr_p)

           beef     veal     turkey
beef      1.000   -0.829    0.738
veal     -0.829    1.000   -0.768
turkey    0.738   -0.768    1.000

In [3]: corr_s = meat[['beef', 'veal', 'turkey']].corr(method='spearman')

In [4]: print(corr_s)

           beef     veal     turkey
beef      1.000   -0.812    0.778
veal     -0.812    1.000   -0.829
turkey    0.778   -0.829    1.000
```

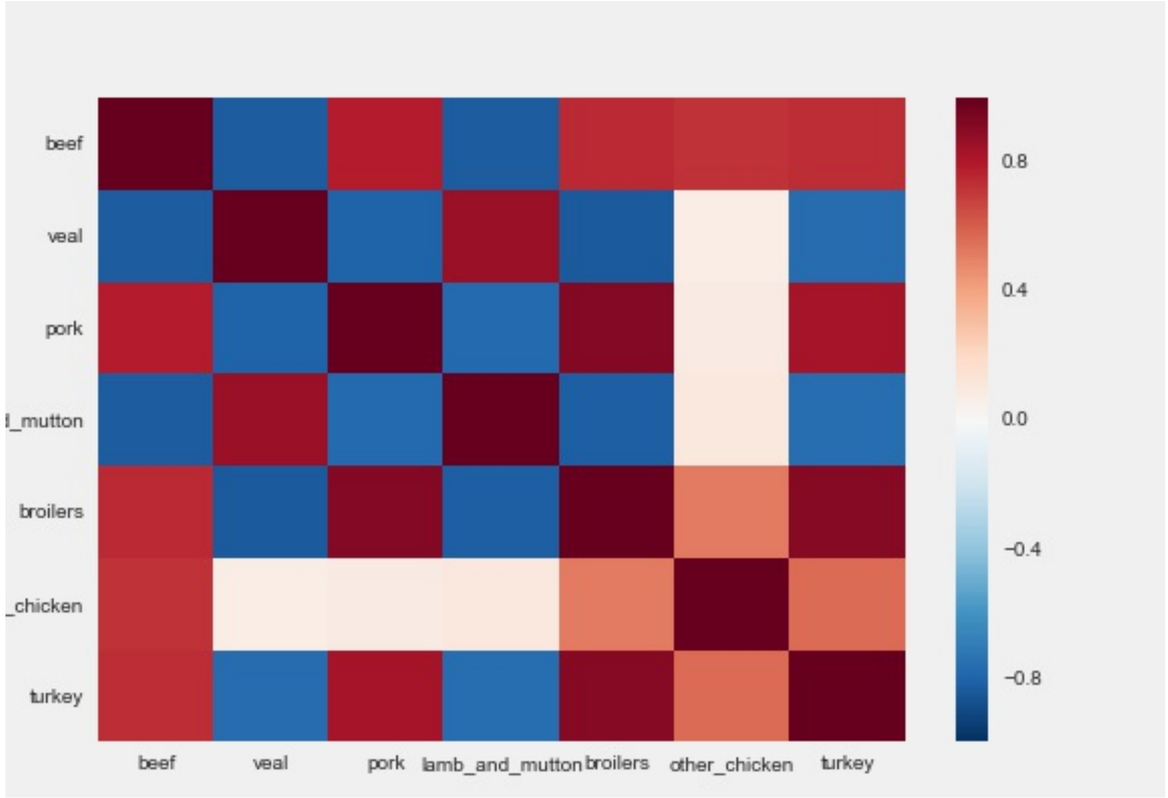
Computing Correlation Matrices with Pandas

```
In [1]: corr_mat = meat.corr(method='pearson')
```

Heatmap

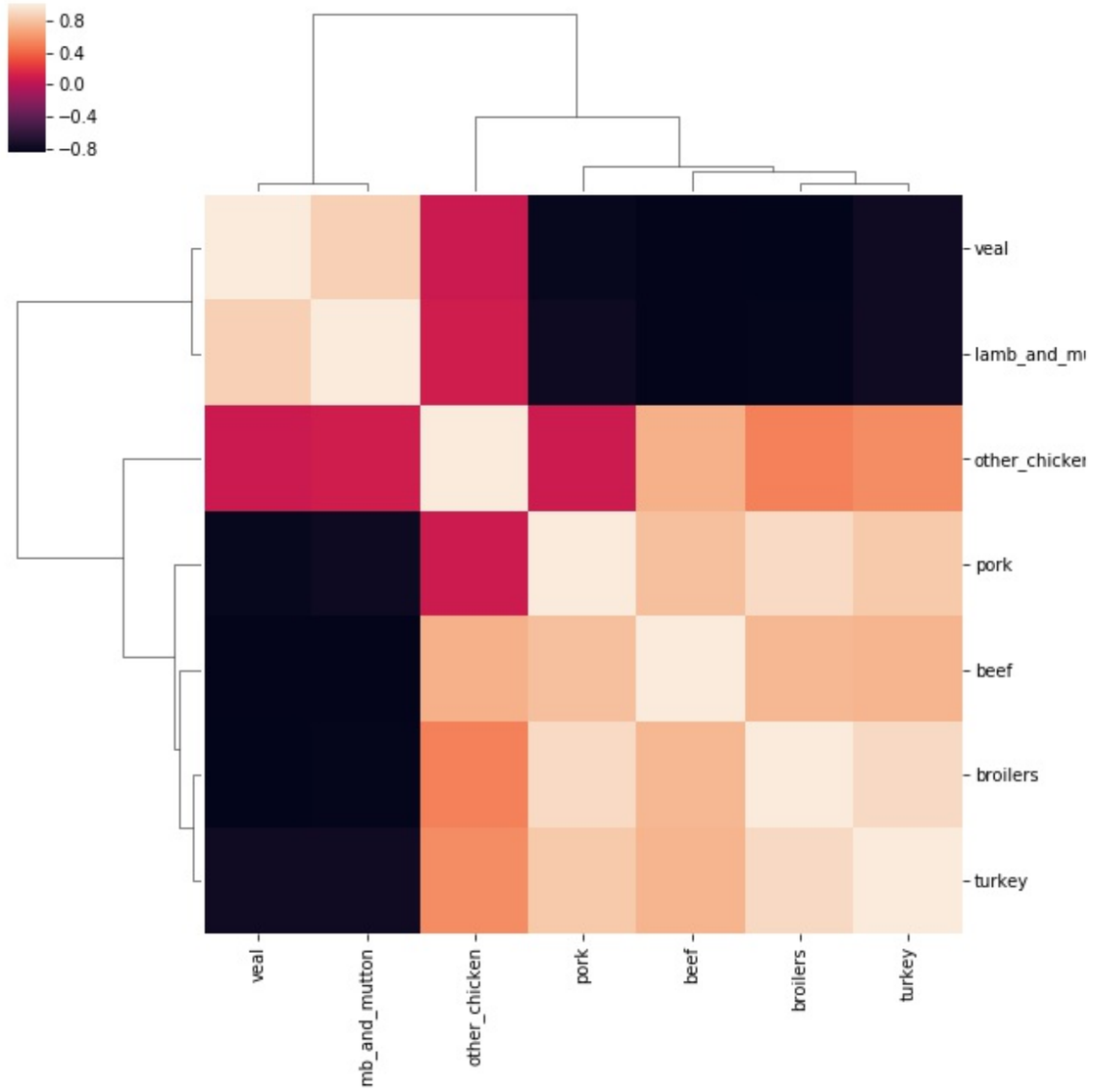
```
In [2]: import seaborn as sns

In [3]: sns.heatmap(corr_mat)
```

Heatmap


beef
veal
pork
l_mutton
broilers
_chicken
turkey
beef
veal
pork
lamb_and_mutton
broilers
other_chicken
turkey
0.8
0.4
0.0
−0.4
−0.8

Clustermap

```
In [4]: sns.clustermap(corr_mat)
```


0.8
0.4
0.0
−0.4
−0.8
veal
lamb_and_mu
other_chicken
pork
beef
broilers
turkey
veal
mb_and_mutton
other_chicken
pork
beef
broilers
turkey

VISUALIZING TIME SERIES DATA IN PYTHON

Let's practice!