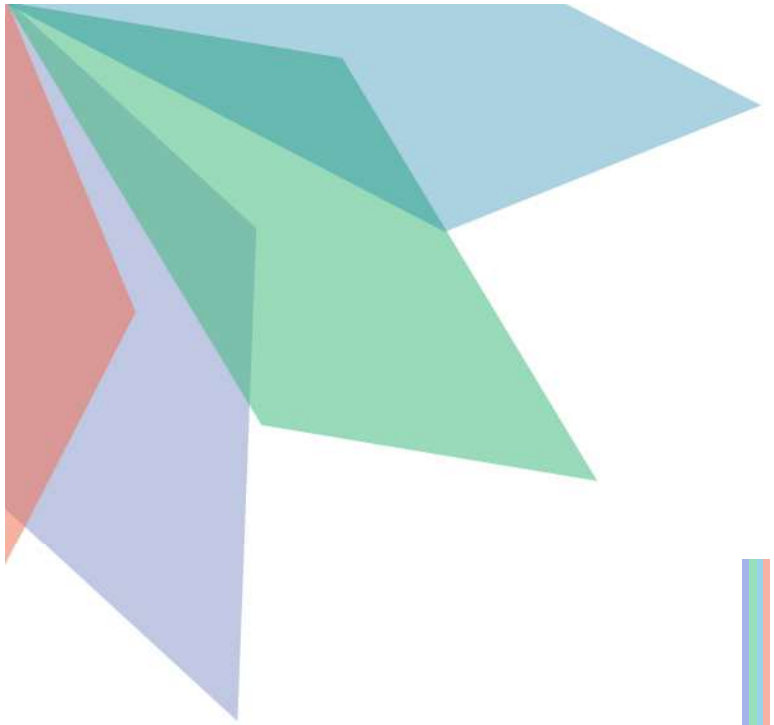


Python for Finance: Introduction

Mamoru Nagano
Seikei University

§ Python Libraries in Session 1

pandas_datareader

https://pandas-datareader.readthedocs.io/en/latest/

datetime

https://docs.python.org/ja/3/library/datetime.html

pandas

https://pandas.pydata.org/docs/

NumPy

https://numpy.org/doc/stable/

1. How to Get Market Data : pandas_datareader

Below, this seminar first introduces how to obtain daily stock price data, interest rate data, and macroeconomic data from databases of financial information agencies and international organizations using APIs. These data acquisition mainly uses pandas_datareader. No package installation is required when using Google Colab.

```
import pandas_datareader as data
```

First of all, we need to describe the source code above to import the library. Here, **data** is a nickname (something like) when using this library and you can choose any nicknames here.

https://pandas-datareader.readthedocs.io/en/latest/

If you don't use Google Colab, but choose to install it on your device, you should type:

```
pip install pandas-datareader
```

The steps to use pandas_datareader are as follows:

```
dataframe name＝data.DataReader(‘ticker code‘,‘data source‘,
sample period start date, last day of sample period)
```

#example :

```
import pandas_datareader as data
df=data.DataReader(‘^KLCI','stooq',start=‘2000-01-01’,end=‘2024-
05-30’)
```

2. Time Series Management : datetime

When acquiring time series data, importing the library datetime in advance will simplify the use of pandas_datareader.

```
import datetime as dt
```

The source codes to import these two libraries and retrieve the daily Kuala Lumpur Composite Stock Index data is as follows.

```
#ex :
import pandas_datareader as data
import datetime as dt
```

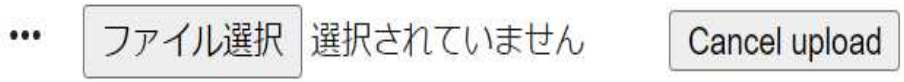
If you fix the data start date as January 1, 2010 and the data end date as May 31, 2024 using datetime, the source code of pandas_datareader will be as follows.

```
start=dt.date(2010,1,1)  #data start date
end=dt.date(2024,5,31)  #data end date
df=data.DataReader('^KLCI','stooq',start=start,end=end)
```

3. Getting Data from a CSV File

If you want to retrieve data from a CSV file, type the following codes and click a "file selection" icon appears on the screen below the cell. Then, select a CSV file from the data folder to retrieve the data.

```
import pandas as pd
from google.colab import files
uploaded = files.upload()
```



After importing data from a CSV file, it is required to store it in a pandas data frame

```
dataframe① = pd.read_csv((‘file_name.csv'),encoding="Shift-JIS",index_col=0)
```

encoding is a command that specifies the character code of the file to be read as Shift-JIS or UTF-8 (for Japanese). Index_col=0 specifies that the leftmost column of data is used as the index.

4. Getting Data from URL

Until now, most of the empirical data has been stored in files on users' PCs using spreadsheet software such as MS-Excel. However, in the future, data storage in data science will tend to be uploaded to online storage such as Google Drive, GitHub, One Drive, etc., retrieved from a URL, and shared among developers rather than on individual personal PCs.

```
dataframe① = "https://github.com/user_name/repository_name/blob/main/file_name?raw=true"

dataframe② = pd.read_csv(dataframe①)
```

ex : Source code to read data from URL and how to store it in a da ta frame

```
url = "https://github.com/nagamamo/data/blob/main/3_single_factor_data.csv?raw=true"

data = pd.read_csv(url)
```

4. Fundamentals of Data Science : Pandas & NumPy

Pandas and NumPy are python libraries that are frequently used in the field of data science. Although their purposes are similar, pandas is more user-friendly, and NumPy is often used for advanced and fast calculations.

In particular, storing data in pandas dataframes improves the efficiency of data analysis. For example, just as you can calculate a new variable from two variables stored in MS Excel and calculate a third variable, you can calculate new variable d ata from two variables stored in a pandas dataframe.。

On the other hand, NumPy is also useful when repeating simulations tens of thousands of times or more when using finance theory for empirical analysis. The reason for this is that NumPy brings high-speed calculations to Python. Since the library of pandas was originally created from NumPy, it is easy to convert data between these two dataframes. Therefore, it is preferable to first store market data in pandas dataframes and then take advantage of the strengths of each to achieve your programming goals.

5. Pandas : Data Management

Pandas can perform not only numerical calculations but also various data management. It is especially useful for data preprocessing such as removing missing values, merging two data frames, and input/output of CSV data. It is also useful for retrieving data with pandas_datareader and then visualizing the data using descriptive statistics with describe() or converting the data to percentage change with pct_change.

```
import pandas as pd

df=data.DataReader('^KLCI','stooq',start='2000-01-01',end='2024-05-30').sort_values(by='Date',ascending=True)
```

6. Pandas ： Average Rate of Returns

If you can store asset price data in a pandas dataframe, you can easily calculate returns and risks. To calculate the return and risk of an asset, if you instruct the data in a pandas dataframe to perform the following calculation, the percentage change from the previous period will be calculated regardless of the frequency of the data, such as daily or weekly.

dataframe② = dataframe①.pct_change()

Additionally, missing values are removed by dropna().

dataframe③= dataframe①.column_name.dropna()

【Note】 dropna() has various missing value processing patterns. If it cannot fix the data by default, please refer to, https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html and configure settings according to individual circumstances.

In the case of time-series data, when converting original asset price data into a rate of change, the first row data is always a missing value.
Therefore, once you get used to typing python source code, it is better to calculate the rate of change and give instructions for missing values at the same time, as shown below.

dataframe③= dataframe①.column_ name.pct_change().dropna()

7. Pandas : Average Rate of Return and Risk

In finance theory, the average rate of return compared to the previous period is the average rate of return, and its standard deviation is the risk. Here's how to calculate average returns, standard deviations, and other descriptive statistics.

dataframe④= dataframe③.mean()
dataframe⑤= dataframe③.median()
dataframe⑥= dataframe③.std()
dataframe⑦= dataframe③.var()
dataframe⑧= dataframe③.max()
dataframe⑨= dataframe③.min()
dataframe⑩= dataframe③.describe()

8. NumPy: from pandas dataframe to ndarray

NumPy is useful for high-speed calculations. Calculations that would take tens of minutes with pandas can be completed in a few seconds using NumPy.

Additionally, there are several libraries that assume NumPy arrays for data input a nd output, such as the machine learning library scikit-learn.

You can store your data directly as a NumPy array (ndarray), but you can convert i t from a pandas dataframe to the ndarray by adding ".values" to the pandas dataframe.

```
import numpy as np
ndarray❶ = dataframe①[['Column_A']].values
```

9. NumPy : Average Rate of Return and Risk

Here's how to use NumPy to convert a pandas dataframe to ndarray and calculate the average rate of return.

ndarray❸= np.log(ndarray ❶).diff(1)

ndarray❹ = np. mean(ndarray❸)
ndarray❺ = np. median(ndarray❸)
ndarray❻ = np.std(ndarray❸)
ndarray❼ = np. var(ndarray❸)
ndarray❽ = np. max(ndarray❸)
ndarray❾ = np. min(ndarray❸)

11. Probability Distribution Depiction

Once the average rate of return and the standard deviation are calculated, a probability distribution can be depicted. Below is the most standard normal distribution used in modern finance theory using its mean and standard deviation.

If the rate of return of a stochastic variable that follows a normal distribution (a variable that fluctuates with a certain probability) denotes x, its mean value is μ, and standard deviation is σ, then the probability density function $f(x)$ can be expressed as:

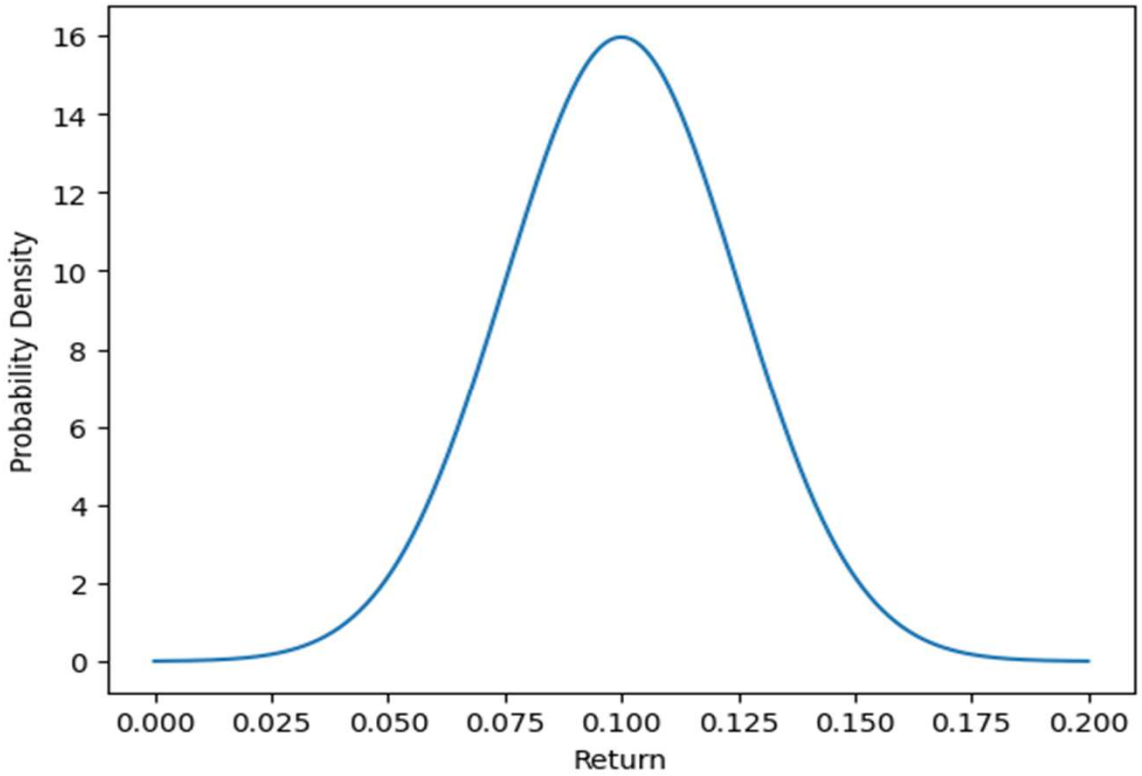
$$f(x) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \qquad (-\infty < x < \infty)$$

[1]Library Imports

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
```

[2]Depiction of Normal Distribution

```
X = np.arange(0, 0.2, 0.0001)
Y = norm.pdf(X, 0.1, 0.025)
plt.xlabel("Return")
plt.ylabel("Probability Density")
plt.plot(X, Y)
plt.show()
```


Probability Density
16
14
12
10
8
6
4
2
0
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
Return

§ Manuals of Python Major Libraries

pandas_datareader
https://pandas-datareader.readthedocs.io/en/latest/

datetime
https://docs.python.org/ja/3/library/datetime.html

pandas
https://pandas.pydata.org/docs/

NumPy
https://numpy.org/doc/stable/

SciPy
https://docs.scipy.org/doc/scipy/

Contact:

Mamoru Nagano,
Professor of Finance,
Seikei University, Japan
mnagano@econ.seikei.ac.jp