FINANCIAL ANALYTICS

Contents

1. Data sources

Before we proceed with the course, let's perform a quick analysis of what it's like to work with financial data. To begin, there are many online sources that provide such information.

1.1. Monetary perspective

These can be split into two categories: paid and free.

Online sources:

paid	vs	free
well-structured data ✓		good enough data quality ✓
missing values rarely		not as precise or as rich compared to the data provided by paid websites ✗

Logically, data from paid online financial data sources is well-structured and rarely contains missing values. Alternatively, free data sources do not necessarily contain incorrect information. They can provide good enough data quality to help you manage your personal finances, among other things. However, it might not be as precise or as rich compared to the data provided by paid websites.

That's why we have based our course data on free online sources. It suits the goal of our course perfectly and comes at no cost to the user.

Nevertheless, we insist that the important skills and tools you will acquire—programming in Python, thinking analytically, and solving finance exercises using Python—are universal. Thus, data comes as a secondary ingredient.

Having said that, let's move on to the next video, where we will shed some light on the current online financial data sources you can use when taking the course. See you there.

So, let's recap what Python for Finance, Investment Fundamentals, and Data Analytics is about. Up to this point, the course has been focusing on teaching you the programming skills in Python. For the rest of the course, we will also look at the fundamentals of finance and investment. We will go through various analytics topics and show you how they can be applied to real-life data using Python.

It's important to emphasize that these three tools—Python, finance fundamentals, and data analytics—are the focus of this course. They are the ones that will give you an edge in the market.

Another crucial ingredient for making good analyses is the data you'll be using. Dealing with high-quality information will have a significant positive impact on the outcome of your work. However, the amount of data available is constantly changing, and you won't make exactly the same conclusions if you perform the same analysis today versus a year from now. Moreover, since this course uses data retrieved from free online sources, it's understandable that you might encounter slightly erroneous numbers or stumble upon missing values.

1.2. Technical perspective

Let's look at using financial data from a technical perspective.

First, the data you'll be using for financial, or any other type of analysis comes from one of two sources: a web server or your computer.

In practice, to access data stored on a web server, you'll need to connect to its API (Application Programming Interface).

In our case, we will need financial data APIs. We can also call them online financial data sources.

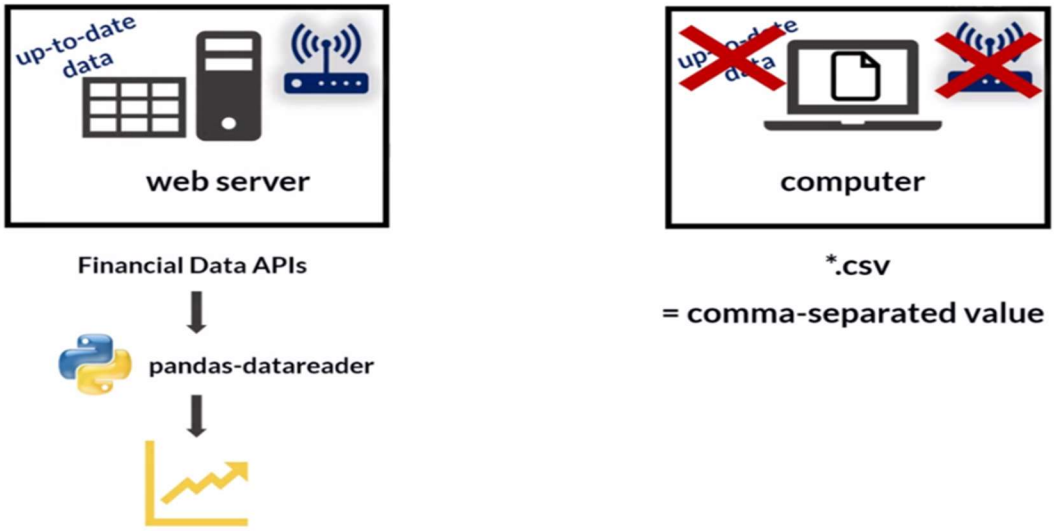
Examples are the IPX, Morningstar, Alpha Vantage, and the Crandall APIs, amongst many others.

As a quick note, we must mention that a characteristic of these servers is that they provide up-to-date data. That's why you'll need an internet connection to access these APIs.

Furthermore, from the point of view of Python, the PANDAS data reader is an example of a module that will help you retrieve data from the financial data sources and prepare it for analysis.

Apart from extracting online data, you may use information stored as certain types of files on your computer.

A file format that every programmer or analyst must know how to work with is the comma-separated value format, frequently abbreviated CSV.

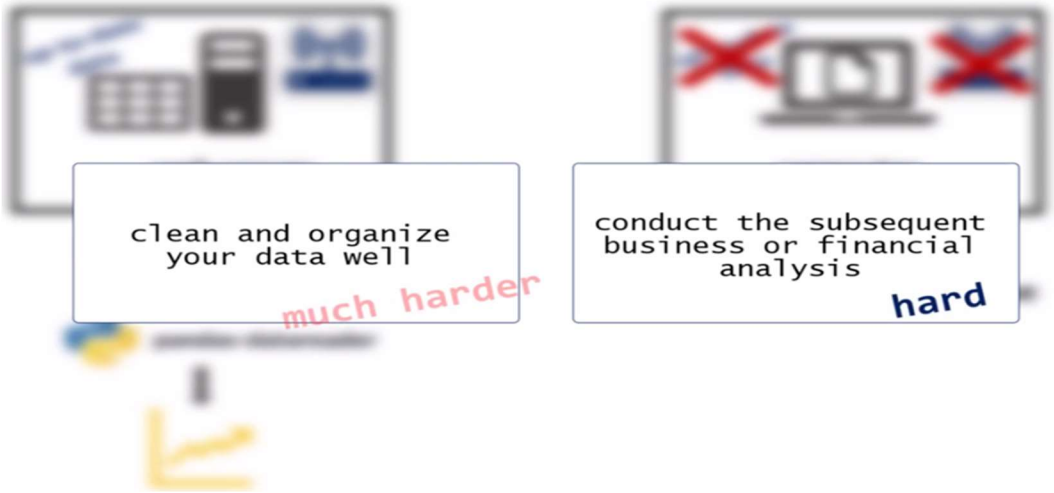


Logically, you won't need an internet connection to use data stored in CSV. However, it will typically not contain up-to-date information.

It is important to know that neither of the two approaches is straightforward. However, in the next few lectures, we will clarify how each of them works and introduce you to the relevant programming tools.

This will be a valuable addition to your skill set, as experts often say it is usually much harder to clean and organize your data well than to conduct the subsequent business or financial analysis.

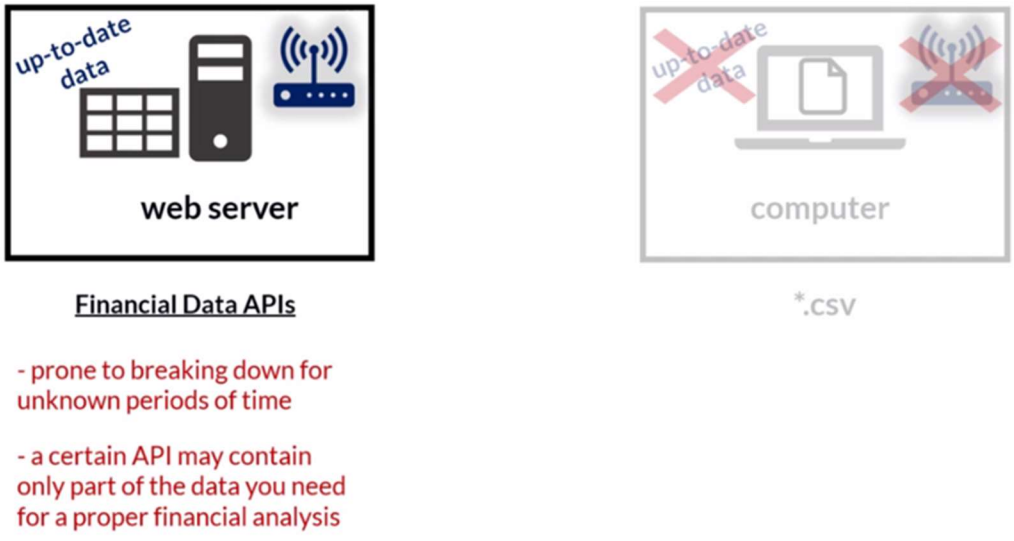
At first glance, it seems from this lecture that you should lean toward APIs, doesn't it? They provide the newest data, and all you need is an internet connection. So why bother using CSV files at all?



Here's why.

First, web services are prone to breaking down for unknown periods of time.

Second, it is possible that a certain API may contain only part of the data you need for a proper financial analysis. Currently, some APIs offer just single or multiple stock data, while others may provide foreign stock data or market indices data. Note that an analyst may often need to work with all of these.



Furthermore, you can only connect to some APIs if you use Python 3 and not when using Python

2. Meanwhile, others are okay if you are using either of the two versions.

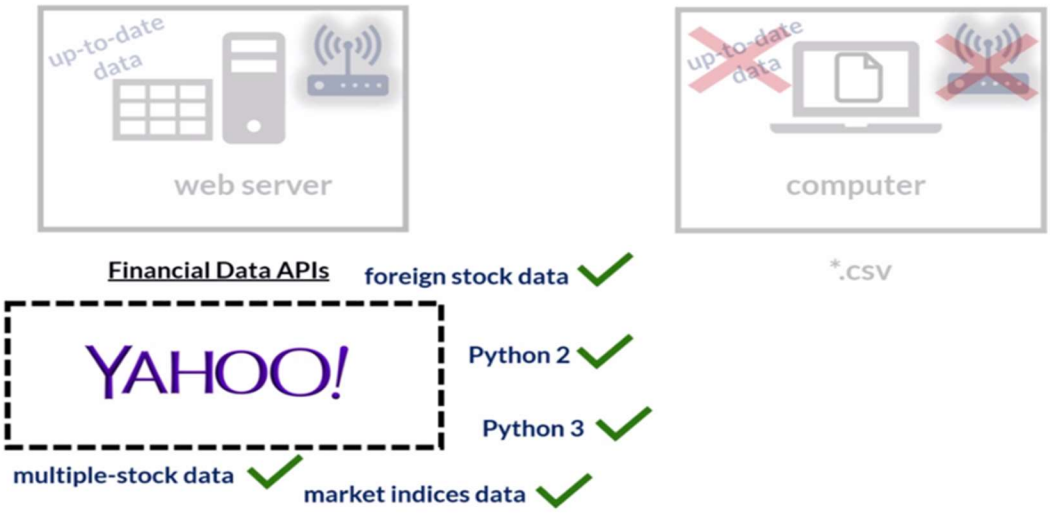
There is a free API, the Yahoo Finance API, which provides all the above-mentioned types of data, and it does that for any version of Python, being perfect for the objectives. It is the API on which we have based our entire course.

That is why, as you proceed, you will see we have used Yahoo! as a data source in our code.

However, online data from Yahoo! may not always be available. Once, on a very gloomy day, Yahoo's web services were interrupted. So, people who do financial analysis had to look for a substitute.

The Google Finance API was around and could substitute the functionality provided by Yahoo! to a great extent. However, its services were discontinued too.

Why internet behemoths like Google and Yahoo may discontinue such services is beyond us. But it is what it is, and we must adapt.



So, focusing on our course rather than the constant appearance of new APIs, the statement we want to make here is the following:

To be sure that the entire financial part of the course flows smoothly from A to Z, use the CSV files attached to the videos. Although the information contained there is not up to date, it will be most appropriate as it is the richest and easiest to-handle data we can provide.

It will allow you to focus on coding, financial theory, and the relation between the two. As we already mentioned, these are the most important skills you acquire on this course anyway.

If you wonder how the CSV files were created, we must say they are based on data retrieved from the Yahoo Finance API at a moment when it was functioning.

However, if this sounds complicated, just remember the CSV files can be found in the resources section of each financial lecture. They are the ones that fit the course goals best.

To be sure that the entire financial part of the course flows smoothly from A to Z,

Use the *.csv files!

Now, if you insist on working with up-to-date financial data that is free and the Yahoo API is not available, we suggest you use the Morningstar or the Alpha Vantage API, regardless of if you are using Python 2 or Python 3. Only use the Alpha Vantage API if you are on Python 3.

However, please remember, we cannot guarantee you will always have access to such services. They can be interrupted, which means downloading data from them may become temporarily or permanently impossible.

We will post messages in the announcement section of the course if there are major changes in the status quo that must be shared with you.

Nevertheless, if you are not able to pull the exact information needed for a specific financial lecture, be prepared to have to change the API or to swiftly switch to CSV files.

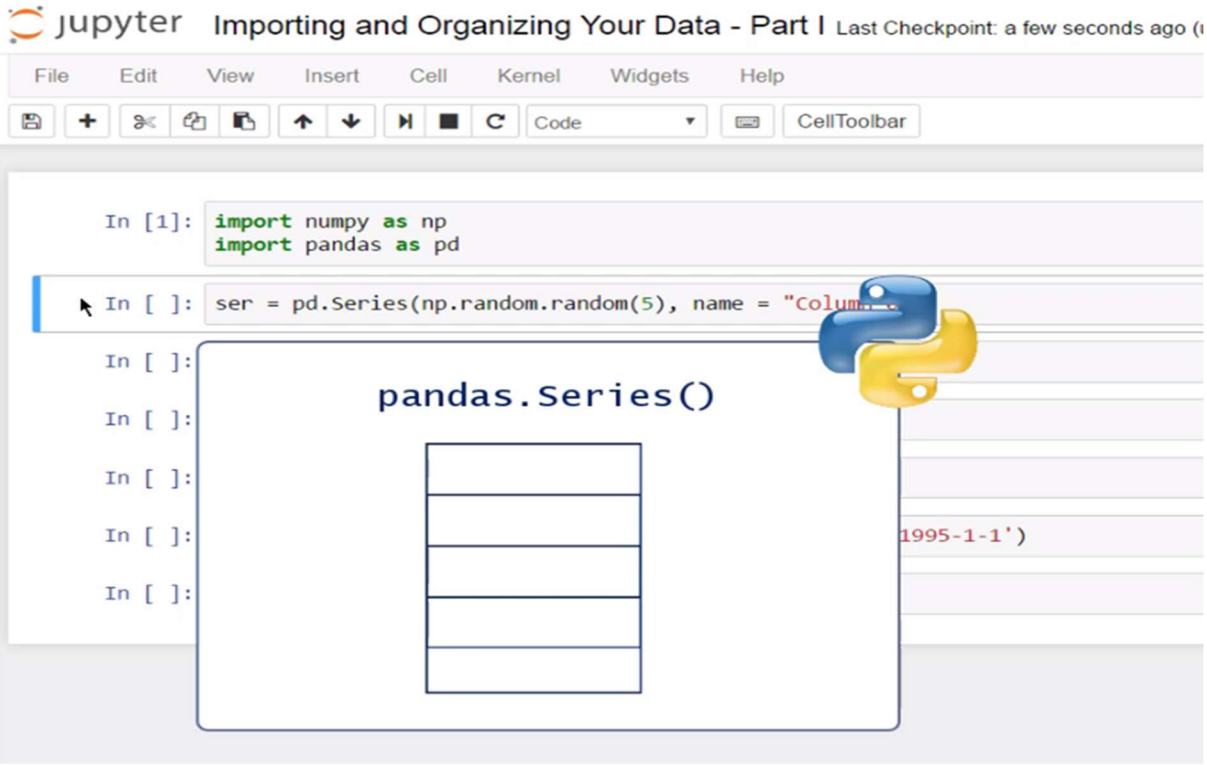
We know this is a long and slightly confusing lecture, but we had no choice. We had to prepare a video introducing you to the options we have when using financial data in Python.

2. Improving and Organizing Data in Python

2.1. Series and Data frame

In this lesson, we will continue to explore fundamental techniques that will help us improve the organization of data in Python. Logically, the focus will be on the PANDAS package, and NumPy will help us illustrate the subject better.

One of the two main data types defined in PANDAS is the Series. You can think of it as single-column data, a set of observations related to a single variable.



Let us create the variable that will carry the information—it will be of the Series type, and I would like it to contain five random numbers. Hence, I can use NumPy's random function to create the five random numbers.

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

In [2]: ser = pd.Series(np.random.random(5), name = "Column 01")

In [3]: ser

Out[3]: 0    0.658426
        1    0.640298
        2    0.773439
        3    0.928862
        4    0.002408
        Name: Column 01, dtype: float64
```

What is left is to assign a name to the column. "Column one" sounds okay, doesn't it?

The indices on the left side of the screen show rows indexed starting from zero and the five randomly generated numbers, just as we expected.

You can think of a Series object as a dictionary whose items are of the same data type. Here, we have float numbers, right? Therefore, you can access one of the elements through its index, just as you would with a dictionary. See [2] in brackets will return the third value from our column.

```
import numpy as np
import pandas as pd
```

```
ser = pd.Series(np.random.random(5), name = "Column 01")
```

```
ser
```

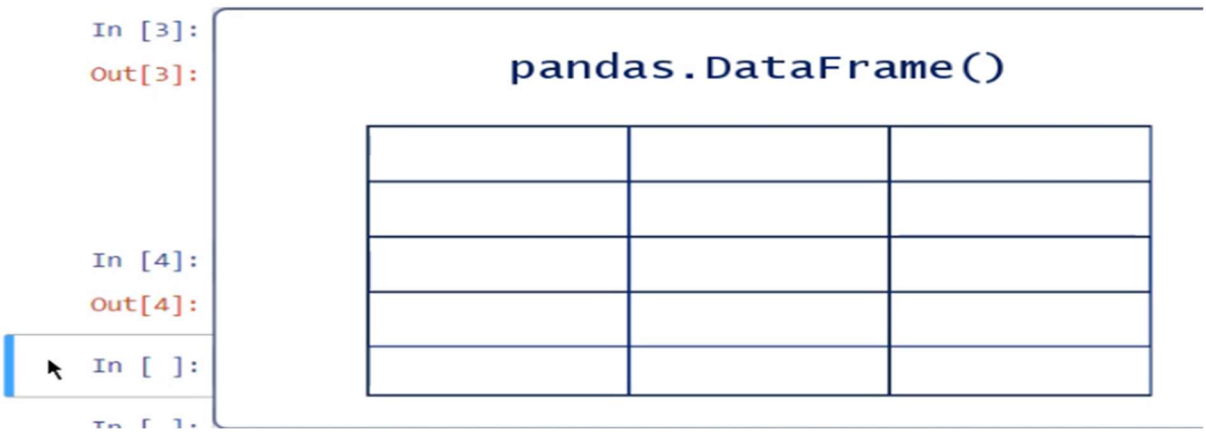
```
0    0.658426
1    0.640298
2    0.773439
3    0.928862
4    0.002408
Name: Column 01, dtype: float64
```

```
ser[2]
```

```
0.77343901787731273
```

```
from pandas_datareader import data as wb
```

The other data type is called a Data Frame. It is like the Series data type, but with several columns.



Before dealing with DataFrame, we should get acquainted with the following.

2.2. Retrieving Single Stock

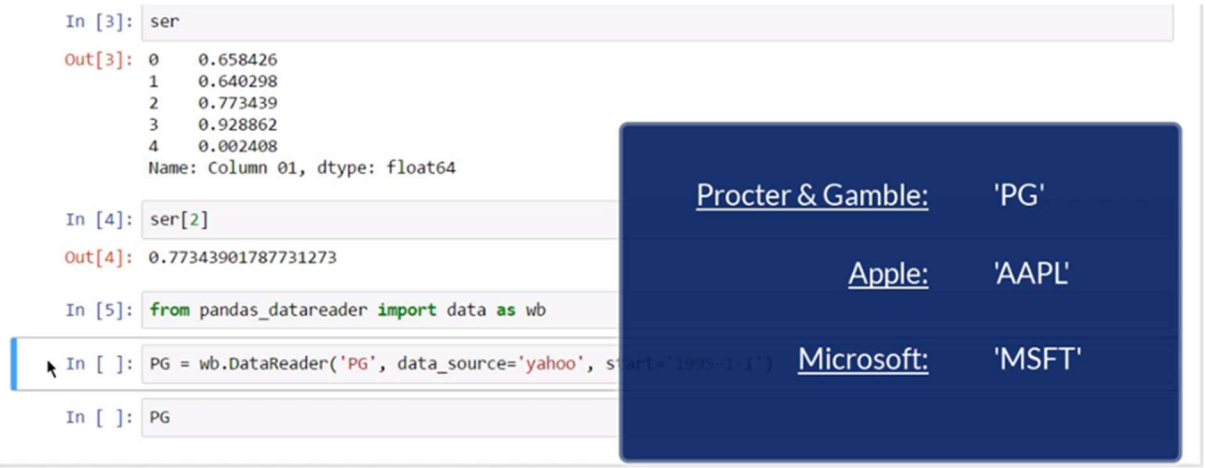
Let's import data from the PANDAS data reader package we've PIP installed. This will help us avoid confusion with the data sets we will later create in our notebook files.

```
In [5]: from pandas_datareader import data as wb

In [ ]: PG = wb.DataReader('PG', data_source='yahoo', start='1995-1-1')

In [ ]: PG
```

Let us rename this item using the following convention: "WB," standing for "web."

Our goal is to extract data from Yahoo! Finance about Procter and Gamble, starting from the 1st of January 1995. The data_reader function will allow us to do this in one row.

Let us call the data set PG, as in Procter and Gamble. Set it equal to the data_reader function from the web package. Use this data reader function and specify three important parameters:



1. The first one is the Procter and Gamble ticker. A ticker is the symbol with which a company's shares are quoted on the stock exchange. For example, Procter and Gamble's ticker symbol is PG, Apple's is AAPL, and Microsoft's is MSFT.

2. After PG, we should determine the data source. It will be Yahoo!

3. Finally, let us specify which data we want extracted. The start date will be the 1st of January 1995, written as 1995-01-01, all hyphenated. We should not forget to place the last two items within quotes.

```
In [5]: from pandas_datareader import data as wb
In [6]: PG = wb.DataReader('PG', data_source='yahoo', start='1995-1-1')
In [7]: PG
```

Out[7]:

	Open	High	Low	Close	Volume	Adj Close
Date						
1995-01-03	61.875000	62.500000	61.750000	62.375000	3318400	9.168104
1995-01-04	62.125000	62.625000	61.250000	61.875000	2218800	9.094613
1995-01-05	61.500000	61.750000	60.875000	61.000000	2319600	8.966002
1995-01-06	60.625000	61.625000	60.625000	61.125000	3438000	8.984375
1995-01-09	61.375000	61.625000	60.750000	60.875000	1795200	8.947629
1995-01-10	61.125000	61.750000	60.750000	61.625000	4364000	9.057867
1995-01-11	62.375000	62.375000	61.500000	61.500000	3738400	9.039494
1995-01-12	61.500000	62.125000	61.250000	62.125000	3307600	9.131359

This is how we can extract real-life historical data. Exciting, right?

In our next lecture, you will learn additional tools that will be helpful for data analytics.

2.3. Retrieving Multiple Stock Data

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

In [2]: from pandas_datareader import data as wb

In [ ]: PG = wb.DataReader('PG', data_source = 'morningstar', start = '1995-01-01')

In [ ]: PG

In [ ]: tickers = ['PG', 'MSFT', 'T', 'F', 'GE']
        new_data = pd.DataFrame()
        for t in tickers:
            new_data[t] = wb.DataReader(t, data_source='iex', start='2015-1-1')['close']

In [ ]: new_data.tail()
```

First, retrieve information about a single stock from IEX. It will suffice to substitute Morningstar with IEX and 1995 with 2015 in the code. Then you can execute the respective code cell.

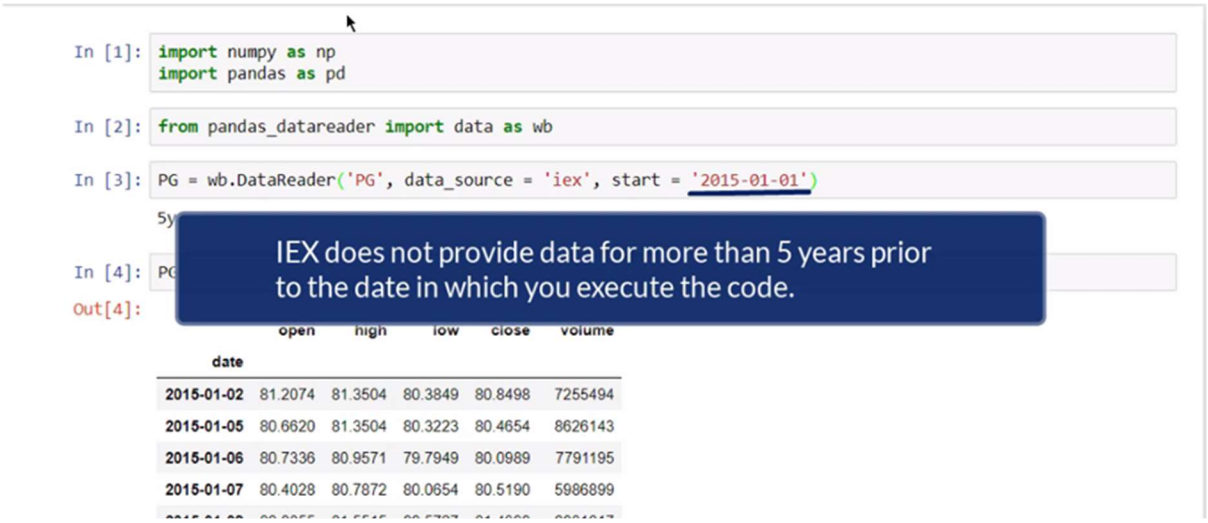
See, we obtained an analogous output with stock prices starting from January 1st, 2015. This is because IEX does not provide data for more than five years prior to the date on which you execute the code.

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

In [2]: from pandas_datareader import data as wb

In [3]: PG = wb.DataReader('PG', data_source = 'iex', start = '2015-01-01')
        5y

In [4]: PG
Out[4]:
```

date	open	high	low	close	volume
2015-01-02	81.2074	81.3504	80.3849	80.8498	7255494
2015-01-05	80.6620	81.3504	80.3223	80.4654	8626143
2015-01-06	80.7336	80.9571	79.7949	80.0989	7791195
2015-01-07	80.4028	80.7872	80.0654	80.5190	5986899
2015-01-08	80.8855	81.5515	80.5727	81.4398	6831617

Now, let us explore how to extract a multiple stock list from IEX. To begin, we need to create a list with the ticker symbols of the companies mentioned. These are PPG, MSFT, T, F, and G.

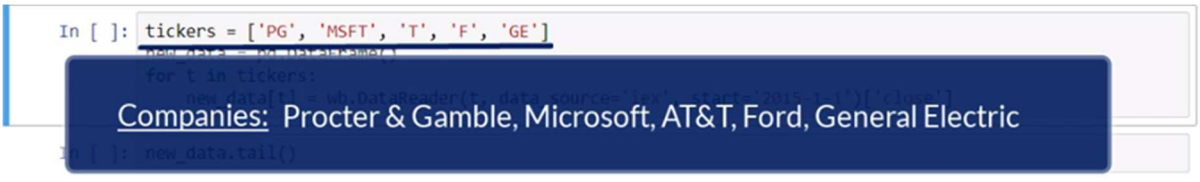
Do you remember how to create a dictionary by typing its name and assigning it to an empty value within braces, and then filling it with key-value pairs? The logic here is similar. I will create a new Data Frame object from pandas and call it new_data.



How will I fill it in with information? Row by row for the past 22 years? No. As mentioned earlier, one of the challenges of working with IEX is that it only provides data for the past five years. However, while this API provides a limited amount of data, it is perfectly suitable for this course, so we can still use it for this exercise.

Instead of filling the Data Frame object with information row by row, I will use iteration in Python. I will write a loop: for every ticker in tickers, provide me with the closing price for each ticker for each instance, that is, for each day since January 1st, 2015.

The correct syntax is to assign the outcome of the data_reader function for each ticker from IEX, starting from January 1st, 2015, to new_data with an index of t. This line is similar to the one used for downloading the data of PAG. The difference is that here we had to index t in the new_data object at the beginning. Importantly, since we are only interested in the closing price column, we should indicate the exact column name in brackets at the end of the line.

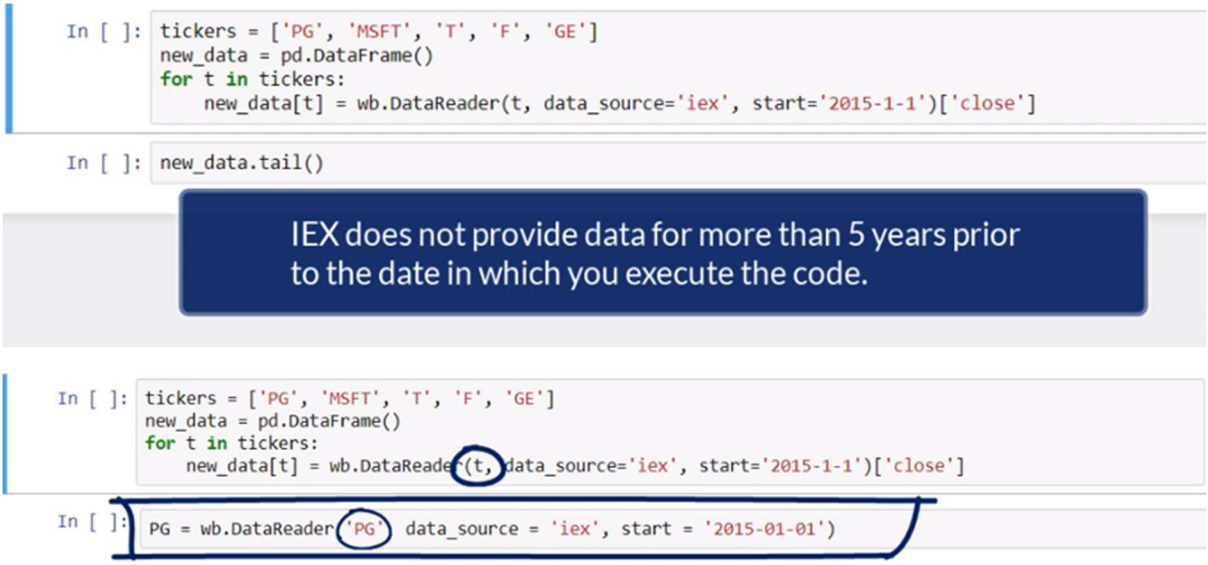


Remember that this API uses lowercase letters for its column names, so we must type "close" in lowercase letters only.

Good. This approach will only work if the information for all five companies is organized in the same way with the same column names. Let's execute and check the tail of the new data set.

```
In [ ]: tickers = ['PG', 'MSFT', 'T', 'F', 'GE']
        new_data = pd.DataFrame()
        for t in tickers:
            new_data[t] = wb.DataReader(t, data_source='iex', start='2015-1-1')['close']

In [ ]: new_data.tail()
```

Looks good, right? As you can see, online data sources only provide a limited part of the information we need for our analysis. In addition, in the notebook files attached to these videos, we will also explain how to use another API, Alpha Vantage. However, this one is not the easiest option from a programmatic point of view. The benefit, though, is that it contains a large amount of rich data.



In the meantime, do not forget that by pressing the link provided in the resources section of the videos, you can access files that contain the code shown in the videos. We encourage you to thoroughly check out the notebook files, as they often contain additional and useful information.

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

In [2]: from pandas_datareader import data as wb

In [3]: PG = wb.DataReader('PG', data_source = 'iex', start = '2015-01-01')

        5y

In [4]: PG

Out[4]:
```

date	open	high	low	close	volume
2015-01-02	81.2074	81.3504	80.3849	80.8498	7255494
2015-01-05	80.6620	81.3504	80.3223	80.4654	8626143
2015-01-06	80.7336	80.9571	79.7949	80.0989	7791195
2015-01-07	80.4028	80.7872	80.0654	80.5190	5986899
2015-01-08	80.8855	81.5515	80.5727	81.4398	6831617


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

In [2]: from pandas_datareader import data as wb

In [3]: PG = wb.DataReader('PG', data_source = 'iex', start = '2015-01-01')

        5y

In [4]: PG

Out[4]:
```


IEX does not provide data for more than 5 years prior to the date in which you execute the code.

date	open	high	low	close	volume
2015-01-02	81.2074	81.3504	80.3849	80.8498	7255494
2015-01-05	80.6620	81.3504	80.3223	80.4654	8626143
2015-01-06	80.7336	80.9571	79.7949	80.0989	7791195
2015-01-07	80.4028	80.7872	80.0654	80.5190	5986899


```
In [ ]: tickers = ['PG', 'MSFT', 'T', 'F', 'GE']
        for t in tickers:

In [ ]: new_data.tail()
```

Companies: Procter & Gamble, Microsoft, AT&T, Ford, General Electric


```
In [ ]: tickers = ['PG', 'MSFT', 'T', 'F', 'GE']
        new_data = pd.DataFrame()
        for t in tickers:
            new_data[t] = wb.DataReader(t, data_source='iex', start='2015-1-1')['close']

In [ ]: new_data.tail()
```

In [ ]:
```
tickers = ['PG', 'MSFT', 'T', 'F', 'GE']
new_data = pd.DataFrame()
for t in tickers:
    new_data[t] = wb.DataReader(t, data_source='iex', start='2015-1-1')['close']
```

In [ ]:
```
new_data.tail()
```

IEX does not provide data for more than 5 years prior to the date in which you execute the code.

In [ ]:
```
tickers = ['PG', 'MSFT', 'T', 'F', 'GE']
new_data = pd.DataFrame()
for t in tickers:
    new_data[t] = wb.DataReader(t, data_source='iex', start='2015-1-1')['close']
```

In [ ]:
```
PG = wb.DataReader('PG', data_source = 'iex', start = '2015-01-01')
```

3. Understanding Investment: Upside and Downside

3.1. Risk-Return Tradeoff

When we think of an investment, we must remember two things: its upside and its downside.

In other words, we should consider:

The profit that will be made if everything goes well.

The risk of losses if the investment is unsuccessful.

This principle applies to every venture, whether you are:

Buying shares in a company.

Investing in corporate or government debt.

Purchasing real estate properties.

Acquiring gold.

Investing in mutual or pension funds.

You must always weigh these two parameters: the profit you can expect and the risk of losing money.

Why Do Different Investments Carry Different Levels of Risk and Profitability?

A comparison between bonds and stocks can help highlight this difference.

Bonds

Government bonds offer an average rate of return of 3%. Historically, there have been very few instances of governments going bankrupt and not repaying what is owed to investors.

While some risk is associated with government bonds, it is minimal and well-contained. An investor buying government debt can be confident that, a year later, they will receive their initial investment plus the expected interest.

Stocks

Equity shares have a higher rate of return, approximately 6%. However, they come with much more frequent fluctuations and price changes, as several factors influence a company's share price, such as:

Operations

Competition

Regulatory environment

Strategic decisions

Industry trends

A lot can go right, but there's also a lot that can go wrong. An investor buying equity shares should understand that the higher return they expect comes at the cost of increased uncertainty and risk.

Conclusion: Balancing Risk and Return

The art of finance is not about maximizing an investor's returns in a single year. It is about making informed decisions that consider both dimensions—risk and return—and optimizing the risk-return combination in an investment portfolio.

Introduction to Risk and Return Calculations

Now that we understand the importance of risk and return for financial securities like stocks and bonds, we are ready to learn how to calculate these two parameters.

In this part of the course, we will use a structured approach:

Concepts will first be explained in theory to help you understand their meaning and purpose.

Then, we'll teach you how to perform the actual calculations in practice using Python.

We are aiming for a trifecta: theory, real-world application, and Python. Awesome!

3.2. Rates of Return

In the first few lessons, we will focus on rates of return.

Investor's Goal: Earning a Good Rate of Return

Every investor's main objective is to earn a good rate of return on their investment. But what does that mean? Let us break it down with a practical example.

Imagine you bought one share of Apple at the beginning of the year when the stock was trading at $105. By the end of the year, the price increased to $116. Apart from any transaction costs or

dividends paid during the year, if you sell your share today, you will receive $116, meaning you made a profit of $11.

Evaluating Your Investment

Is that a good return? Should you be satisfied with your investment? To answer that, you would need to compare your investment in Apple shares with other potential investments. However, since different stocks have different prices, you cannot just compare dollar amounts. Instead, you need a measure that ensures comparability between investments with different prices—this is where the rate of return comes in.

Simple Rate of Return

The rate of return allows us to measure performance consistently across various investments. To calculate the simple rate of return, use the following formula:

Rate of Return=Ending Price−Beginning PriceBeginning Price\text {Rate of Return} = \frac{\text{Ending Price} - \text{Beginning Price}}{\text{Beginning Price}}Rate of Return=Beginning PriceEnding Price−Beginning Price

In our case, it would be:

116−105/105=10.5%\frac{116 - 105}{105} = 10.5\%

So, the simple rate of return for your Apple investment is 10.5%. This allows you to compare it with other investments more easily.

Adjusting for Dividends

If Apple paid a $2 dividend at the end of the year, the rate of return calculation would include the dividend:

116+2−105/105=12.4%\frac{116 + 2 - 105}{105} = 12.4\%

Logarithmic Rate of Return

You can also calculate the logarithmic return, which is given by:

Log Return=log (Ending PriceBeginning Price)=log (116)−log (105)\text{Log Return} = \log\left(\frac{\text{Ending Price}}{\text{Beginning Price}}\right) = \log(116) - \log(105)Log Return=log(Beginning PriceEnding Price)=log(116)−log(105)

The log return in this case is approximately 10%, slightly different from the simple return. The choice between using simple or log returns depends on consistency:

Use simple returns when dealing with multiple assets over the same time frame.

Use log returns when calculating returns for a single asset over time.

Time Frames and Comparisons

It's crucial to remember the time frame for which you've calculated a rate of return. In our example, the rate of return is based on a yearly period (from January 1st to December 31st). You can calculate rates of return for periods other than a year, such as daily, monthly, or quarterly, but be careful not to compare investments with different holding periods. For consistent comparisons, investors typically annualize these returns.

Historical and Expected Rates of Return

Investors are always interested in future rates of return, but since predicting the future is impossible, we rely on historical performance to form expectations. The past performance of a security helps us estimate a reasonable expected rate of return.

In the next lesson, we'll practice calculating a security's historical rate of return. See you there!

3.3. Risk

Investors love earning a great rate of return, and that's clear. They don't like risk or uncertainty, the chance of losing a large portion of their money.

Risk and Return in Investment Decision-Making

We said risk and return are the two most important dimensions in investment decision-making. Therefore, it is easy to understand why we must spend a good portion of time learning how to measure and forecast a security's risk.

Defining Risk for an Investor

How can we define risk for an investor? If you invest $1,000 of your money in a stock that's trading on the stock exchange, and you know that, on average, this stock earned 15%, you would want to know how the average was made, right?

Whether the stock earned 14% in one year, 16% the next year, 13% the year after that, and 17% in the last year of historical observations, or whether it earned +50%, -20%, -20% again, and then +50% in the final year. There is a significant difference between the two sets of data.

Understanding Variability

In the first case, you can be certain your money will earn an amount that is in line with what you expect. Things can be slightly better or slightly worse, but the rate of return is always 13% to 17%.

Things change dramatically with the second set of data. There is a huge variability from one year to the next. The average return is the same. However, you can't have an idea about what comes next. If you are an investor who holds the stock for two years, you can lose 40% of its value if the years when you hold the stock are years two and three.

The Role of Variability in Finance

So, variability plays a significant role in the world of finance. It is the best measure of risk we have. A volatile stock market is much more likely to deviate from its historical returns and surprise investors negatively. Yes, it can surprise investors positively, too. However, investors don't like surprises and are much more sensitive to the possibility of losing their initial investment.

Risk-Averse Investors

Most people prefer to have a good idea about the rate of return they can expect from a security or a portfolio of securities and are doing their best to reduce the risk they are exposed to. We need to remember that investors are risk averse. They don't like risk for the sake of risk, and their main goal is to measure the risk they are facing and reduce it as much as possible.

Measuring Risk: Variance and Standard Deviation

Commonly used statistical measures such as variance and standard deviation can help us a great deal when we try to quantify risk associated with the dispersion in the likely outcome. Such dispersion is measured by a security's variance and standard deviation.

Calculating Variance

To be more precise, the variance of a security measures the dispersion of a set of data points around their mean value. It is equal to the following algebra equation: variance (σ^2) is equal to the sum of the squares of the difference between a data point (X) and the mean, divided by N - 1.

For those of you who haven't seen it, here is an example that should make things easier. The mean of the four data points is 15%. Let's calculate the dispersion of each of the four points from the mean and elevate them to the second degree:

$14\% - 15\%^2$

$16\% - 15\%^2$

$13\% - 15\%^2$

$17\% - 15\%^2$

These are the four dispersions from the mean elevated to the second degree. Let's calculate their sum. Once we have the sum, we have to divide it by the number of observations we've had, minus one. Here we had four observations, so we'll divide it by three. The variance is equal to 0.00033.

Standard Deviation

If we take the square root of the variance, we obtain the standard deviation of this sample of observations, which is equal to 1.8%. Let's now calculate the variance of the second set of numbers. We use the same formula for the four observations we have here:

$50\% - 15\%^2$

$-20\% - 15\%^2$

$-20\% - 15\%^2$

$50\% - 15\%^2$

Let's sum the four components, divide by three, and obtain the variance. As you can see, it is much larger in this case—0.16—and the standard deviation is even higher, 40%. But we kind of expected it, didn't we? In this example, we knew the lowest and the highest returns were 35% away from the average value of 15%. So, we obtained a close approximation once more. The second set of returns had a much larger dispersion and was a lot riskier.

Conclusion

This is how we can calculate a stock's variance and standard deviation. We are doing well. In our next lesson, we'll learn how to do this in Python. Thanks for reading.

3.4. Calculating Security Risk

The focus of this lecture will be on calculating a security's risk.

Setting Up Data and Libraries

We start with the familiar steps of importing the necessary libraries and downloading data from Yahoo Finance, which we store in a DataFrame object called security_data.

Selecting the Companies and Time Period

It is essential to ensure that we extract data for the correct companies and within the correct timespan. Here, the two companies chosen are Procter & Gamble (P&G) and its German peer Beiersdorf, with the tickers PG and B, respectively. We will examine their stock behavior over the past ten years, starting from January 1, 2007.

Defining Risk and Volatility

The standard deviation of a company's returns can also be referred to as its risk or volatility. A stock with returns that show a large deviation from its mean is considered more volatile. In this lecture, we will analyze which of the two companies' stocks are riskier or more volatile.

Logarithmic Returns and Security Returns Table

To begin, we calculate the logarithmic returns for each company over the selected timeframe. These log returns give us a clearer picture of each stock's behavior, which we store in a new table called Security Returns. This table will have two columns, one for P&G and one for Beiersdorf, containing their respective log returns.

Calculating Mean and Standard Deviation

Next, we calculate the mean and standard deviation for the log returns in the Security Returns DataFrame. First, we obtain the daily average return using the mean, which results in a small number. To annualize this value, we multiply it by the number of trading days in a year (250).

This gives us an annual rate of return slightly above 6%. The same Python logic is applied to calculate the stock's volatility, which involves using the method std() for standard deviation. To annualize the volatility, we not only multiply it by 250 but also raise it to the power of 0.5 (taking the square root of 250), as the standard deviation is itself the square root of the variance.

Analyzing Both Companies' Risk and Returns

We then repeat this process for Beiersdorf, finding a higher mean but also a higher volatility percentage. To make the comparison easier, we print the annualized means and standard deviations of both companies' side by side.

Using NumPy for More Advanced Calculations

An alternative method to obtain both means and risk values is by leveraging NumPy. However, this requires creating two-dimensional arrays for the mean and standard deviation values, which allows for more complex operations, such as matrix multiplication, in advanced calculations.

Conclusion

The lesson reinforces the key principle from the previous lecture: stocks with higher expected returns often come with greater risk. Beiersdorf's returns are slightly higher than P&G's, but this advantage comes at the cost of greater volatility.

4. Introduction to the Relationship Between Financial Securities

In this lesson, we will talk about one of the most important concepts in finance: the relationship between financial securities. It is reasonable to expect that the prices of shares in a stock exchange are influenced by the same factors. The most obvious example is the development of the economy in general.

Favorable macroeconomic conditions facilitate the business of all companies. When people have jobs and money in their pockets, they spend more, and companies benefit as their revenues increase. In a tough economy, consumers reduce their spending and buy mainly products of primary importance. Since stock prices are determined by profits, whenever the economy is doing well, stock prices are higher, and investors are willing to pay a high price for profitable companies. Conversely, during a recession, companies' profits are lower, and share prices fall significantly.

Industry-Specific Impacts

So, one important relationship is that company shares are influenced by the state of the economy. However, different industries are influenced in different ways. Who do you think will suffer more during a crisis—carmakers or supermarket chains?

Of course, people can't stop buying food and groceries, but they can easily postpone buying a new car and continue driving an old vehicle. Therefore, the state of the economy impacts different industries differently.

Importance of Diversification in Investment Portfolios

How important is this to an investor building a portfolio of stocks? It is very important. Imagine the following scenario:

> You own stock in a tech company, say Facebook, and have savings to buy shares in a second company. Your two options are LinkedIn and Walmart. Let's assume you expect the same rate of return from the two stocks (though unrealistic, this will serve our example).

All else being equal, are you going to choose LinkedIn or Walmart? The right answer would be Walmart, because Walmart operates in a different industry. This gives you protection as an investor. If things don't go well in the Internet space—such as a decline in advertising spending, stronger competition, or rapid technological changes—your Walmart stock will likely be unaffected.

On the other hand, if Walmart doesn't perform as expected, you still have your Facebook stock, which operates in a completely different industry. By buying shares in two companies from different sectors, you diversify your portfolio and reduce excessive risk for the same level of expected return.

Understanding Relationships Between Companies' Stock Prices

Clearly, there is a relationship between the stock prices of different companies, and understanding this relationship is crucial for investors looking to build optimal portfolios. Understanding what drives this relationship and how to measure it is essential for effective portfolio management.

In our next lesson, we will learn how to measure the relationship between two variables. See you there!