Documentación descarga de datos con reuters eikon

Tabla de contenido

[Instalar en una nueva Maquina 2](#_Toc138746994)

[Clonar el repositorio al disco 2](#_Toc138746995)

[Crear Entorno Virtual e Instalar paquetes 3](#_Toc138746996)

[Configuración de Eikon 5](#_Toc138746997)

[Crear un App Key 5](#_Toc138746998)

[Configuración de .env 5](#_Toc138746999)

[Uso de Eikon 6](#_Toc138747000)

[Hacer llamadas con Python 7](#_Toc138747001)

[Uso de los Notebooks de Descarga 9](#_Toc138747002)

[mass\_download.ipynb 9](#_Toc138747003)

[download\_calcs\_ranking.ipynb 10](#_Toc138747004)

[Las funciones de Descarga - data\_retrieval.py 12](#_Toc138747005)

[vertical\_download 12](#_Toc138747006)

[reconstruction 13](#_Toc138747007)

[dfs\_list\_from\_dir 13](#_Toc138747008)

[download\_indicators 14](#_Toc138747009)

[continue\_download 14](#_Toc138747010)

[Errores de Descarga Conocidos 15](#_Toc138747011)

[Failed to decode response to json: 15](#_Toc138747012)

# Instalar en una nueva Maquina

Los notebooks de descarga, para correr, necesitan tener todos los ficheros ‘.py’ contenidos en la carpeta ‘eikon\_data\_retrieval’ del repositorio de GitHub. El notebook download\_calcs\_ranking.ipynb a demás necesita el los ficheros app\_functions.py y custom\_calculations.py ubicados en la carpeta ‘app’ del repositorio. También necesita la versión de Python 3.10.9 y cumplir con los requisitos de los paquetes supletorios especificados en ‘data\_requirements.txt’. La forma de cumplir con estos requisitos es opcional (uno se puede descargar el código y los paquetes de muchas formas), aunque aquí se detalla cómo hacerlo con GitHub Desktop y Anaconda, por ser los más simples. La última versión del cliente de Eikon también debe estar instalada en la máquina.

**Pagina descarga de Eikon:** <https://eikon.refinitiv.com/index.html>

**Pagina descarga de GitHub Desktop:** <https://docs.github.com/en/desktop/installing-and-configuring-github-desktop/installing-and-authenticating-to-github-desktop/installing-github-desktop>

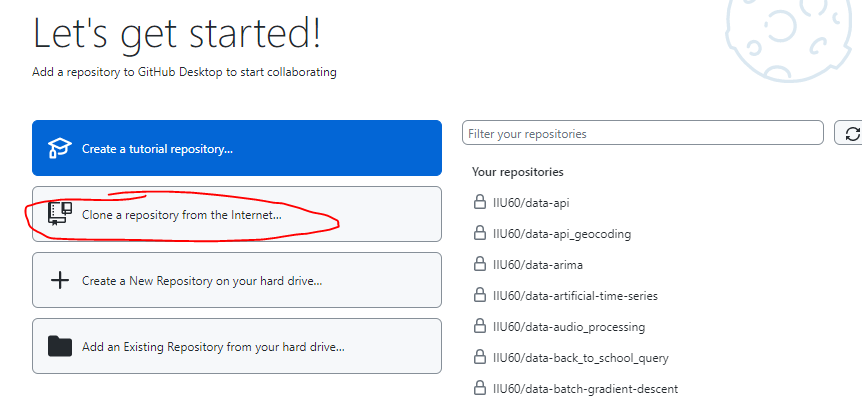
**Página de descarga de Anaconda3**: <https://docs.anaconda.com/anaconda/install/index.html>

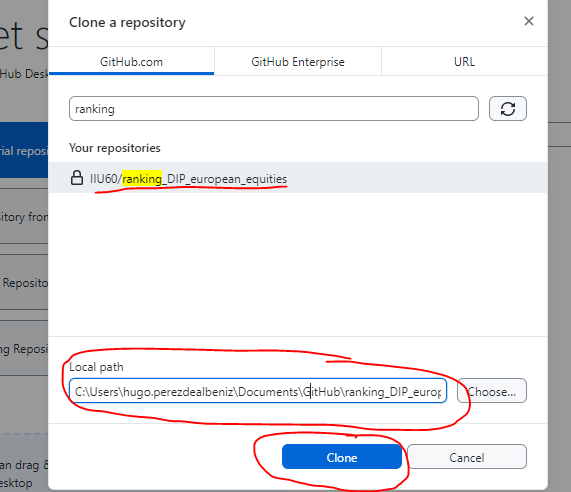
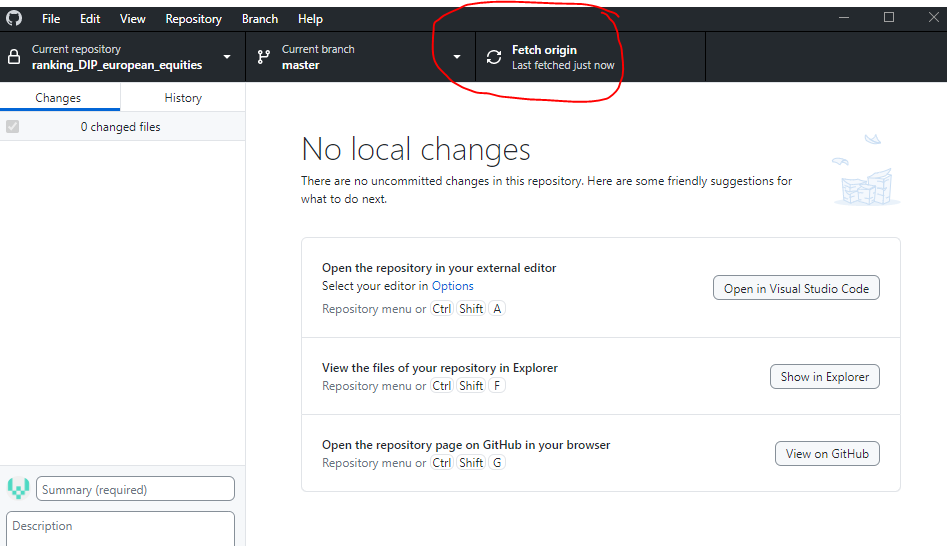
**Repositorio en GitHub:** <https://github.com/IIU60/ranking_DIP_european_equities>

Tras instalar ambas plataformas:

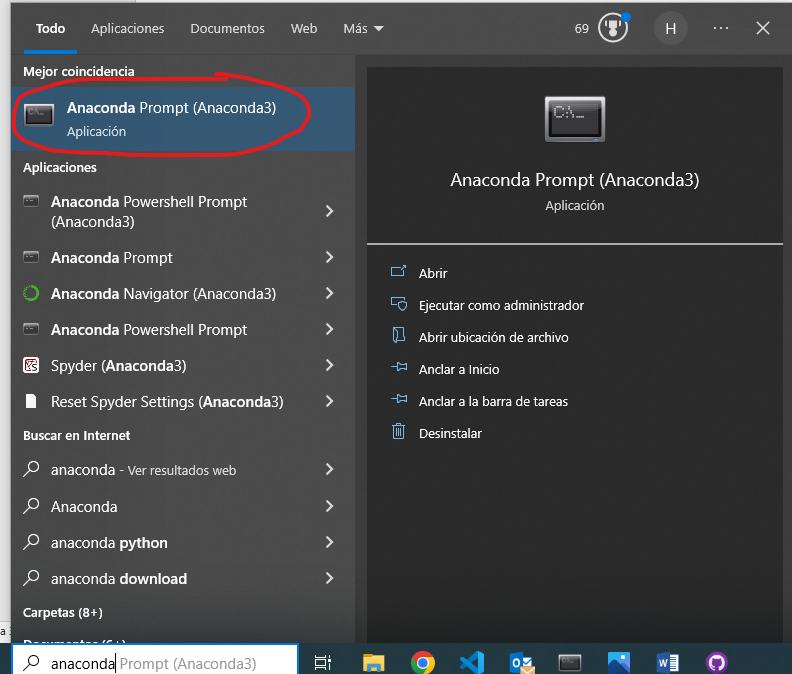
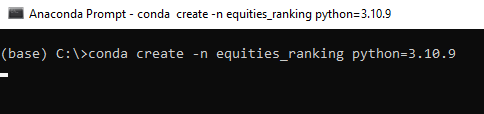
## Clonar el repositorio al disco

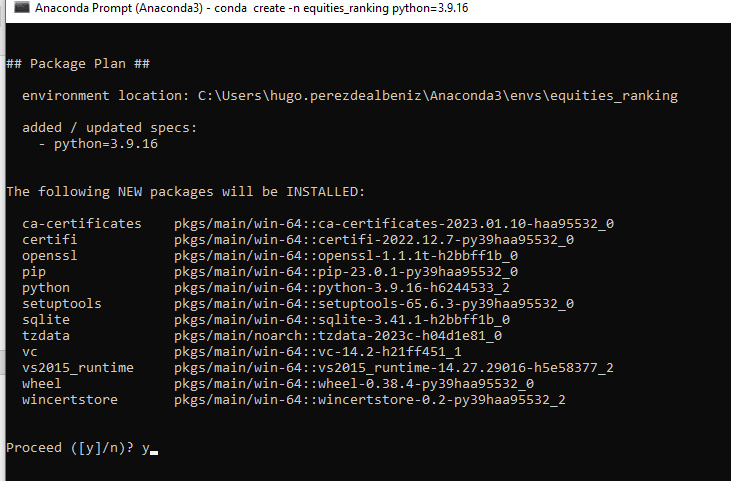
1. Abrir GitHub Desktop e iniciar sesión
2. Hacer clic en clonar repositorio de internet



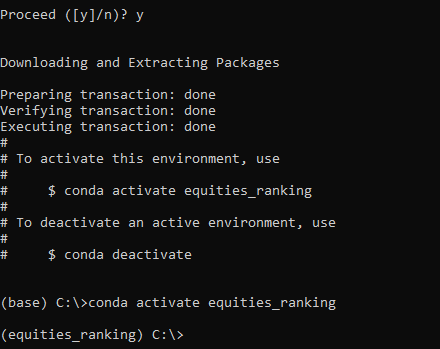
1. Clonar al disco local:
   1. Buscar y seleccionar el repositorio (en la barra de búsqueda si se tiene acceso, y si no, pegando el URL del repositorio público en la pestaña de ‘URL’)
   2. **Importante** copiar la ruta local de guardado
   3. Hacer clic en el botón de clonar
2. En la página del repositorio, actualizar el código con el botón ‘Fetch origin’, si ha habido algún cambio al programa.

## Crear Entorno Virtual e Instalar paquetes

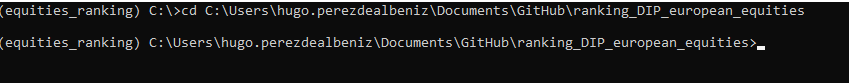
1. Abrir una terminal de Anaconda3 (Se llama Anaconda Prompt) 
2. Crear un entorno virtual con la versión de Python y nombre deseados usando el próximo comando:
   1. conda create -n equities\_ranking python=3.10.9 
   2. al ser preguntado si proceder a instalar paquetes responder que sí con ‘y’



1. Activar el Entorno Virtual:
   1. conda activate equities\_ranking

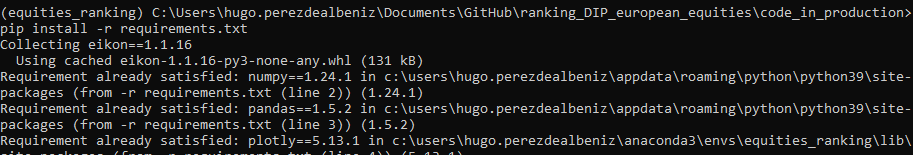


1. Navegar al directorio del repositorio:
   1. cd [ruta a la carpeta ‘ranking\_DIP\_european\_equities’]



*La ubicación predeterminada de los repos de GitHub Desktop está en Documentos/GitHub*

1. Cambiar al directorio con el código de descarga:
   1. cd eikon\_data\_retrieval
2. Instalar requisitos:
   1. pip install -r data\_requirements.txt



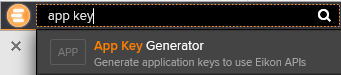
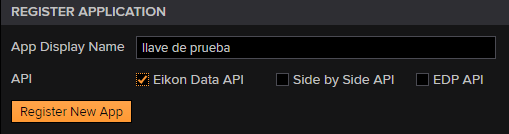
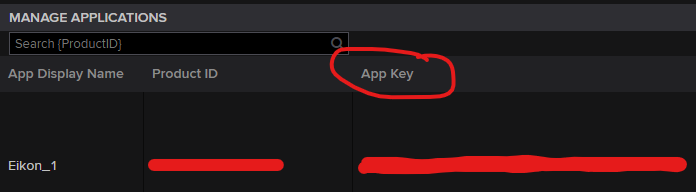
Una vez se ha instalado todo correctamente se pueden cerrar todas las aplicaciones usadas hasta ahora.

## Configuración de Eikon

La aplicación de Eikon le da al usuario acceso a todas sus funcionalidades, pero para usar la API, la aplicación funciona como vía de conexión (proxy) entre el programa que hace la llamada y los servidores de Reuters. Para esto es necesaria una llave única para la aplicación con la que el programa puede identificarse de forma segura.

### Crear un App Key

Para crear una llave de identificación para la maquina en uso hay que seguir los siguientes pasos:

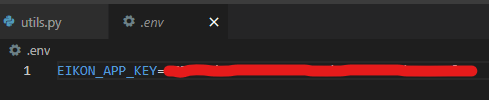
1. Abrir la aplicación de Eikon y entrar con usuario y contraseña
2. Abrir el ‘app key generator’ usando la barra de búsqueda de Eikon
3. Elegir un nombre apropiado y descriptivo, seleccionar ‘Eikon Data API’, y presionar en ‘Register New App’
4. Aceptar los términos y condiciones
5. Copiar la llave creada y guardar en algún lado temporalmente

Tras esto la aplicación de Eikon está preparada para usar la API.

## Configuración de .env

Por motivos de seguridad las llaves de aplicación de Eikon no deben ser compartidas. Para evitar fijarlas en el código y que se compartan sin querer, este programa hace uso de un paquete llamado python-dotenv el cual permite crear y añadir variables globales al entorno virtual dentro de un fichero .env. Este fichero y sus contenidos deben ser creados en cada máquina y nunca ser compartidos.

Para hacer uso de esta funcionalidad simplemente hay que crear un fichero llamado .env y pegar la llave de Eikon en una variable llamada EIKON\_APP\_KEY (\*\*Ojo. Sin espacios\*\*):

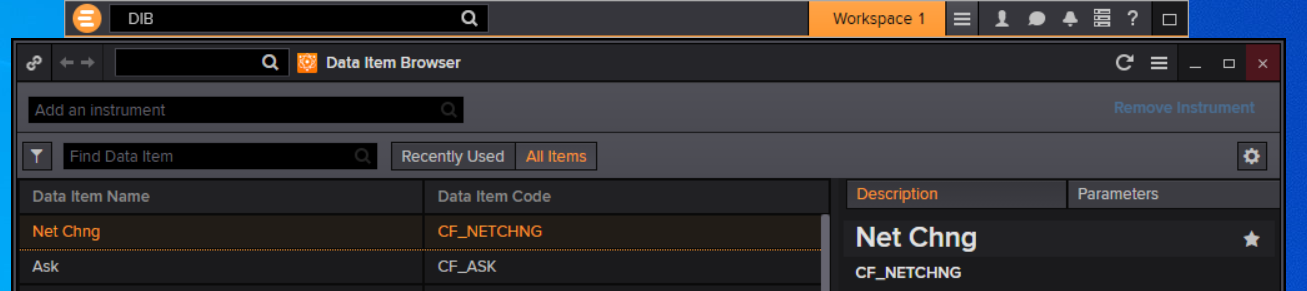


El programa buscará este fichero en el directorio ‘ranking\_DIP\_european\_equities’ por lo que debe ser creado/movido aquí.

Ahora ya está todo preparado para descargar datos de Eikon.

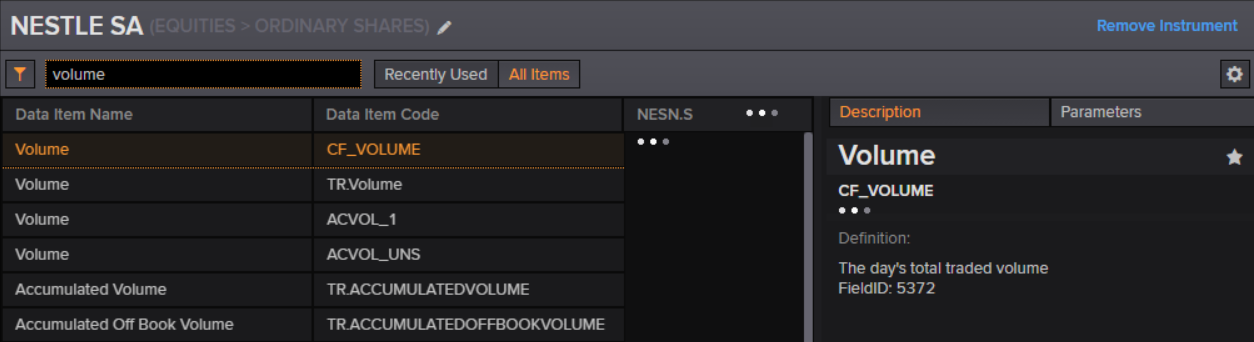
# Uso de Eikon

La principal aplicación de Eikon que se necesita para la descarga de datos se llama Data Item Browser, y sirve para acceder a los campos descargables para cada instrumento (acciones, índices, etc) y comprender sus parámetros. Se accede a ella buscando ‘DIB’ en la barra de búsqueda que aparece al abrir Eikon.

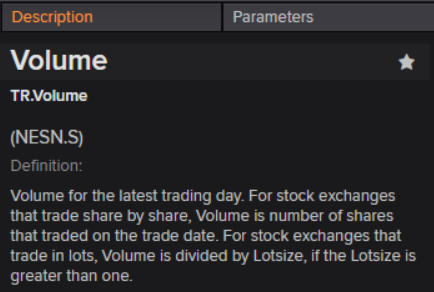
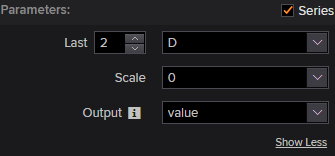


Dentro de este buscador se encuentran todos los campos disponibles para cada instrumento y es necesario comprender sus parámetros para hacer la llamada a la API de forma correcta.

Por ejemplo: supongamos que queremos descargar el volumen de mercado para Nestlé, con frecuencia diaria, desde el comienzo de año. Para esto hay que buscar una de las acciones deseadas en la barra de instrumentos (“Add an instrument”) y en “Find Data Item” buscar “volume” lo que mostrará todos los campos relacionados al volumen.



Cada campo tiene un apartado con una descripción y sus parámetros predeterminados, además de una pestaña adicional para modificar los parámetros. Generalmente las funciones con series temporales tienen ‘TR.’ de prefijo.

El valor de esto es poder investigar los parámetros que requiere cada campo y las distintas opciones con las que se puede llamar su función. Abriendo el drop-down con la letra ‘D’ se muestra un menú con todos los valores que se pueden pasar al parámetro de frecuencias (‘Frq’). Abriendo el menú de Output se ven todos los sufijos que se pueden añadir a la función para descargar información adicional sobre cada dato; cosas como la fecha, fecha de publicación/fecha de calculo, divisa, etc.

Cada campo tiene su propia parametría por lo que hay que investigarlos siempre antes de hacer las llamadas a la API. Primordialmente los únicos parámetros que hay que especificar son:

* **SDate:** La primera fecha de la llamada. Si la llamada no es para varias fechas, este es el parámetro de fecha.
* **EDate:** la última fecha del rango deseado.
* **Frq:** La frecuencia temporal con la que hacer la llamada (diaria, semanal, mensual, trimestral, etc).
* **Period:** el periodo relativo a la fecha de la llamada. En esencia es como un lag; para la fecha 2023-05-01, period=FY0 devolverá el dato del ‘last financial year’ (2022-12-31), y period=FY-1, la del fin del año anterior (2021-12-31).
* **Curn:** La divisa deseada para el dato (EUR,USD,GBP, etc.)

En la esquina inferior izquierda se va construyendo la formula con la que llamar a la API. En el siguiente apartado se verá como traducir esto a los notebooks de Python.



# Hacer llamadas con Python

Eikon tiene un paquete de Python que, si se han seguido los pasos del primer apartado, estará ya instalado en el entorno virtual. Cualquier notebook/programa con el que descargar datos debe conectarse a la API. Para establecer la conexión a los servidores de Reuters para descargar datos dos cosas deben ocurrir:

1. La aplicación Eikon debe estar abierta y con conexión a internet
2. El programa debe identificarse con la llave de la aplicación

El paso 2 se hace así:

from dotenv import load\_dotenv

import os

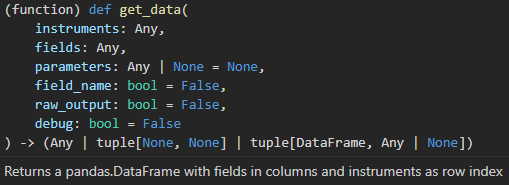
load\_dotenv(r'..\.env') #load environment variables from .env file

import eikon as ek

ek.set\_app\_key(os.environ['EIKON\_APP\_KEY']) #set Eikon API key

ek.set\_app\_key(‘LLAVE\_DE\_APPLICACION’) es la forma de identificar al programa. Lo anterior es para importar la llave desde el fichero .env, lo que es innecesario si no se usa dicha barrera de seguridad.

El paquete de Eikon contiene funciones para descargar noticias y demás, pero para la descarga de indicadores financieros la función más útil es ek.get\_data(), cuya firma es así:



La documentación del resto de funciones se halla en el próximo hipervínculo: <https://developers.refinitiv.com/en/api-catalog/eikon/eikon-data-api/documentation#eikon-data-ap-is-for-python-reference-guide>

Para el ejemplo del Volumen diario desde comienzo de año, la llamada se hace así:

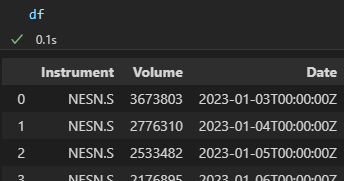
df, err = ek.get\_data(instruments='NESN.S',

            fields=['TR.Volume','TR.Volume.date'],

            parameters={'SDate':'2023-01-01',

                        'EDate':'2023-05-09',

                        'Frq':'D'})

Lo que devuelve:

‘df’ es la variable a la que se asigna el pd.DataFrame devuelto por la función y ‘err’ es donde se asigna cualquier error que ocurra en la llamada.

Dentro de ek.get\_data se especifica el ticker a usar instruments='NESN.S' (Nestlé), los campos a descargar como una lista en ‘fields’ (el volumen y la fecha), y los parámetros deseados como diccionario en ‘parameters’. Los mismos parámetros son pasados a todas las funciones en ‘fields’.

Si se desea descargar varios tickers simplemente hay que pasarlos en una lista a ‘instruments’.

Todas las llamadas siguen este mismo formato.

# Uso de los Notebooks de Descarga

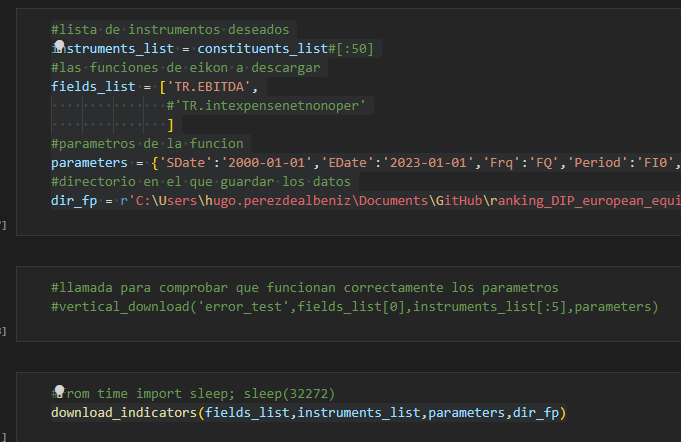
En la carpeta ‘eikon\_data\_retrieval’ hay dos notebooks: mass\_download.ipynb que sirven para descargar datos de forma masiva (muchos campos y tickers a la vez), y download\_calcs\_ranking.ipynb el cual está diseñado para descargar datos actuales, combinarlos si se desea con las funciones de cálculos disponibles en la plataforma (halladas en app/custom\_calculations.py), rankear los resultados, y extraer los tickers de cada cuantil.

Para descargar datos, ambos se nutren de las funciones escritas en eikon\_data\_retrieval/data\_retrieval.py

En estos notebooks no hace falta incluir el paso de identificación de Eikon ya que lo hace data\_retrieval.py de forma automática.

## mass\_download.ipynb

Después de importar los paquetes necesarios el notebook tiene estas tres celdas:



En la primera se definen las variables a pasar a ‘download\_indicators’ - la función de descarga. Estos son: la lista de tickers a descargar, la lista de campos deseados, el diccionario de parámetros, y la ruta a la carpeta donde se deben guardar los datos.

La segunda celda hace uso de la función ‘vertical\_download’ como pequeña prueba, para comprobar que los parámetros funcionan correctamente, antes de comenzar la descarga masiva.

La última celda llama a la función de descarga. Comentada en esta celda hay una línea que hace esperar al programa antes de hacer la llamada a la API lo cual es útil en caso de recibir una penalización de temporal por sobrepasar el límite de descarga por minuto.

Para comenzar a descargar datos simplemente hay que ajustar los campos de la primera celda y correr todo el notebook.

## download\_calcs\_ranking.ipynb

El segundo notebook está diseñado para descargar datos recientes y manipularlos con facilidad (para obtener las listas de tickers que comprar a la hora de rotar cartera). Para ello, además de las funciones de descarga, están importados los scripts con la función de ranking, y el que contiene las funciones de cálculos (rate\_of\_change, exponential\_ma, beta,etc).

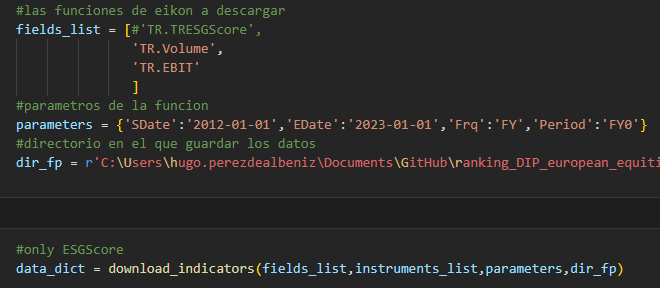
El notebook contiene el ejemplo de cómo obtener los rankings de los actuales constituyentes del stoxx600 para un indicador personalizado (una combinación de cálculos a distintos indicadores).

Para descargar los actuales miembros del Stoxx 600 se usa la próxima llamada:

instruments\_list = ek.get\_data('0#.STOXX','TR.RIC')[0].RIC.tolist()

El instrumento ‘0#.STOXX’ es una lista de constituyentes creada por Reuters para la cual llamamos al campo de tickers ‘TR.RIC’ (Refinitiv Identificación Code’). Del DataFrame devuelto, se convierte la columna llamada RIC en una lista asignada a la variable ‘instruments\_list’.

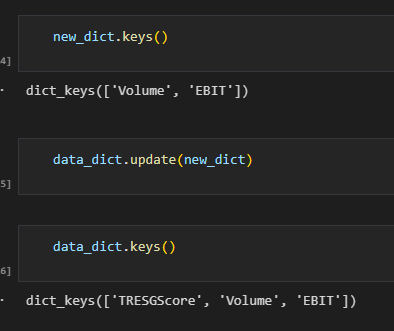
Tras adquirir la lista de instrumentos deseados, se procede a descargar los datos de la misma manera que en el notebook anterior: ajustando los parámetros, el directorio de guardado, y corriendo la función ‘download\_indicators’.



Aquí se hace uso del ‘output’ de ‘download\_indicators’ que es un diccionario con los DataFrames de los datos descargados.

Si hay una equivocación con los campos (se quieren cambiar tras la primera llamada, o se quieren añadir más), se puede modificar la lista de campos, descargar estos campos y guardarlos en un nuevo diccionario;



para después añadirlos al diccionario original y complementarlo:

Una vez adquiridos los datos se puede comenzar a calcular con ellos.

Extrayendo los datos del diccionario:

esg\_df = data\_dict['TRESGScore']

volume\_df = data\_dict['Volume']

ebit\_df = data\_dict['EBIT']

Es importante comprobar la calidad de los datos mirando el número de NaNs que hay en cada DataFrame:



Y después diseñar la función de cálculo deseada:

calc1 = calcs.exponential\_ma(volume\_df.apply(pd.to\_numeric),13)

calc2 = calcs.simple\_ma(ebit\_df.apply(pd.to\_numeric),4)

calc3 = calcs.rsi(esg\_df,4)

custom\_calc\_df = calc1\*calc2/calc3

Aquí, es una media móvil exponencial del Volumen a un año, multiplicada por la media móvil a 3 meses del EBIT, dividido por el RSI a 3 meses del ESGScore.

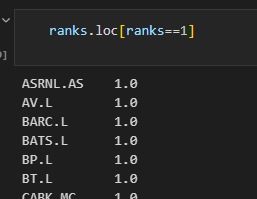
Con los datos finales calculados, se puede pasar el DataFrame a la función de ranking:

n\_quantiles = 10

ranks = af.rank\_data(custom\_calc\_df,n\_quantiles,'high').iloc[-1,:]

‘.iloc[-1,:]’ coge la última fila de la tabla – la fecha más reciente (debería ser la actual).

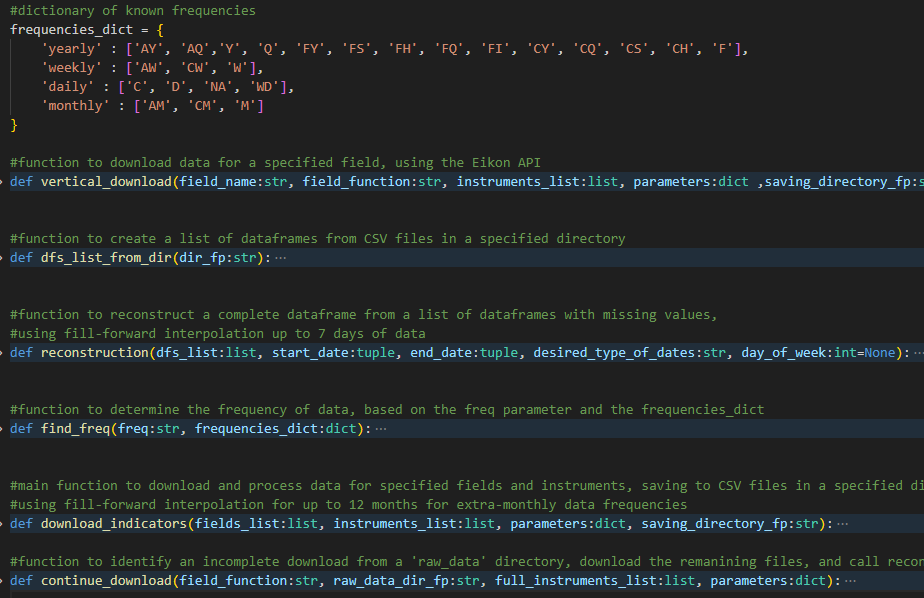
¡Ahora ya son accesibles las listas de tickers de cada cuantil!



# Las funciones de Descarga - data\_retrieval.py

El fichero data\_retrieval.py contiene todas las funciones usadas por los notebooks para hacer llamadas a Eikon y limpiar + procesar los datos. Esta sección solo documenta las partes del fichero que pueden ser relevantes para el uso de sus funciones en los notebooks de descarga (en caso de haber algún error usando su configuración actual). En caso de desear comprender todas las funciones con exactitud, el fichero está adecuadamente documentado con comentarios, lo que debería ser suficiente para su entendimiento.

La estructura del código es así:



## vertical\_download

Esta función crea las carpetas donde guardar los datos (una con el nombre del indicador y una subcarpeta llamada ‘raw\_data’ donde guardar los ficheros de cada acción), y hace llamadas al API ‘de manera vertical’ – llamadas individuales para cada acción en vez de iterando sobre las fechas.

Por si sola se le pueden dar dos usos:

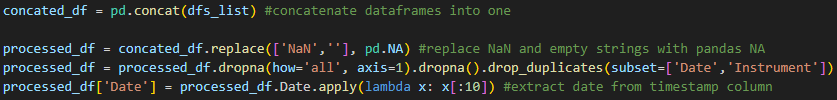
1. Si ha habido algún error de descarga (y se ha tenido que cortar la llamada), en caso de no querer tener que volver a empezar, se podría modificar la lista de tickers para solo contener los tickers que no se han descargado y juntar los ficheros manualmente.
2. Usarla para una llamada pequeña con intención de comprobar que los parámetros establecidos funcionan correctamente (como se ha mostrado en mass\_download.ipynb)

La función devuelve una lista con los DataFrames de los datos descargados (guardarlos en .csv es una precaución).

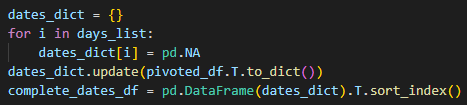
## reconstruction

Esta es la función de procesamiento de datos. Recibe una lista de DataFrames (normalmente devuelta por vertical\_download) y devuelve una tabla de datos estandarizada. Además de los DataFrames, requiere dos tuples (año,mes,dia) para la fecha de comienzo y fin del periodo, y otro argumento que especifica la frecuencia temporal de los datos.

La función primero junta y limpia los datos, quitando NaNs y filas duplicadas:



Luego hace un pivot de los datos para crear la tabla con las acciones como columnas y fechas en el índice:

Rellena la tabla con todas las fechas (diarias) entre la fecha de comienzo y la fecha fin:

Hace un relleno para repetir los datos que faltan hasta un máximo de 7 dias:

Y finalmente selecciona solo las fechas deseadas según el parámetro ‘desired\_type\_of\_dates’:

Si la frecuencia es menor a mensual (trimestral, bianual, anual, etc.) hace un relleno de 12 meses:

## dfs\_list\_from\_dir

Esta es una pequeña ‘helper function’ que crea una lista de DataFrames de los ficheros de un directorio. En caso de haber algún error con ‘reconstruction’ se puede usar esta función para obtener el mismo resultado que ‘vertical\_download’, y manipular los datos como sea deseado.

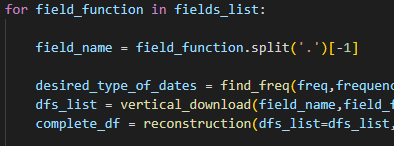
## download\_indicators

Esta es la función principal del fichero y junta la funcionalidad de todas las otras funciones. Es la que se llama en los notebooks. Recibe los mismos argumentos que ek.get\_data() además de la ruta donde se deben guardar y descargar los datos.

Lo primero que hace es interpretar los parámetros de la llamada:



Estas líneas especifican el formato del diccionario de parámetros de la llamada: fechas en formato año-mes-día, y debe contener el parámetro de frecuencia.

Después itera sobre la lista de campos a descargar, y para cada uno hace llamadas a las otras funciones para descargar los datos y procesarlos:

Y finalmente guarda el fichero en la carpeta creada por ‘vertical\_download’



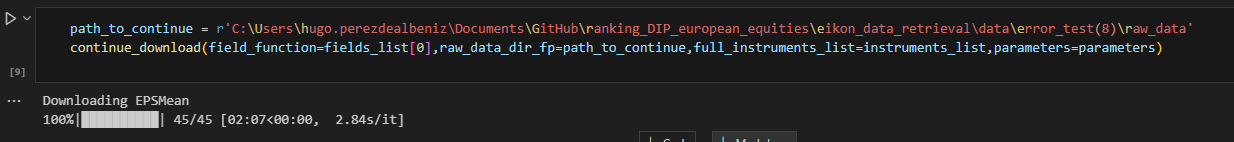
## continue\_download

Esta es una función diseñada para continuar una descarga que ha fallado por cualquier razón. Toma la ruta a una carpeta de ‘raw\_data’ (donde se estaban descargando los ficheros de la descarga fallada), lee los archivos que ya se ha descargado, los compara con la lista total de acciones, y continua la descarga de las que faltan. También llama a ‘reconstruction’ y guarda el fichero generado en el directorio superior a ‘raw\_data’. Su firma es así:



La llamada es similar a una mezcla entre ‘vertical\_download’ y ‘download\_indicators’. ‘field\_function’ es la función de Eikon a descargar (‘TR.EBIT’ por ejemplo), raw\_data\_dir\_fp es la ruta absoluta al directorio con los ficheros de la descarga incompleta, ‘full\_instruments\_list’ es la lista de todas las acciones a descargar, y ‘parameters’ es el diccionario de parámetros con el que se estaba haciendo la llamada anterior.

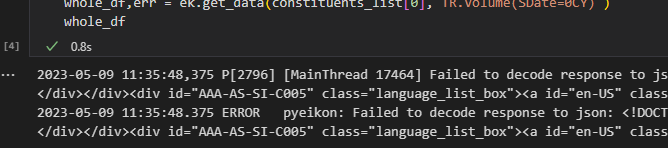
En el notebook ‘mass\_download\_vertical’ está la próxima muestra de esta llamada:



# Errores de Descarga Conocidos

Esta sección detalla la serie de errores conocidos que pueden ocurrir durante la descarga de datos, y que hacer para arreglarlos.

### Failed to decode response to json:



Este error ocurre cuando Eikon está abierto, pero no hay una sesión iniciada, ya sea porque otra persona ha entrado con la misma cuenta, o por haber perdido la conexión a internet, por ejemplo. En cualquier caso, la solución no es simple, no se limita a reiniciar la sesión. Hay que reiniciar Eikon del todo (cerrando todas sus instancias en el administrador de tareas, o reiniciando el ordenador).

Si este error salta durante una descarga masiva, es de suma importancia cortar la llamada cuanto antes para evitar sobrepasar el limite de llamadas por minuto/segundo.