COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Sprint 1. Historias de Usuario 1 y 2

Elena Conderana Medem y Sergio Cuenca Núñez

Tecnologías de Datos Masivos
Big Data Technology

Índice

Introducción

1. Contexto

El objetivo final del proyecto busca diseñar e implementar una arquitectura Big Data completa, que permita procesar y analizar datos procedentes de `TradingView` para poder asesorar a clientes en el trading de criptomonedas. Este primer sprint pretende implementar la adquisición de datos y definir la capa de almacenamiento de datos históricos.

2. Descripción del problema

Para poder asesorar a clientes en el trading de criptomonedas es necesario recopilar datos históricos de las mismas para poder inferir pronósticos futuros. El sprint gira en torno a obtener los datos históricos de los últimos 4 años con una frecuencia diaria de las criptomonedas Bitcoin (BTC), Ethereum (ETH), Ripple (XRP), Solana (SOL), Dogecoin (DOGE), Cardano (ADA), Shiba Inu (SHB), Polkadot (DOT), Aave (AAVE) y Stellar (XLM). Asimismo, es necesario almacenar y gestionar los datos de manera que sea sencillo acceder y trabajar sobre ellos.

3. Objetivos

Los objetivos principales de esta práctica comprenden:

- Familiarizarse con la herramienta `TradingView`.
- Obtener el histórico de los datos de las 10 criptomonedas nombradas con frecuencia diaria de los últimos 4 años.
- Diseñar una primera jerarquía de carpetas.
- Definir el formato de los archivos de datos históricos que se almacenarán en HDFS.

4. Justificación

Para poder proporcionar un asesoramiento fundamentado es necesario analizar tanto el comportamiento histórico como en tiempo real de las criptomonedas, que permitan extraer conclusiones razonadas. Para ello es necesario adquirir los datos correspondientes a las criptomonedas bajo estudio. Estos datos deben ser fácilmente accesibles y manipulables, jugando la jerarquía utilizada durante el almacenamiento un papel crucial en la eficiencia para interactuar con los mismos.

Metodología

En la siguiente sección se procede a describir en detalle cómo se ha implementado la obtención del histórico de los últimos 4 años de las 10 criptomonedas objeto del proyecto, abordando cuestiones técnicas sobre el proceso de desarrollo y sobre las tecnologías utilizadas.

5. Descripción del entorno de desarrollo

Para acceder a la información histórica de las criptomonedas se va a emplear la herramienta `TradingView`. Una plataforma de gráficos y red social, que permite seguir diferentes activos en los mercados globales y pone instrumentos de análisis a disposición de *traders* e inversores globalmente.

El acceso a esta herramienta se lleva a cabo a través del repositorio `https://github.com/ravalmeet/TradingView-Data.git`. Este repositorio proporciona un `downloader` de datos de `TradingView` desarrollado en Python, que permite descargar hasta 5000 velas japonesas de cualquier índice o stock en cualquier formato temporal soportado. Para hacer uso del `downloader` se ha clonado el repositorio y se han importado las clases `TradingViewData` e `Interval` al fichero en el que se va a implementar la extracción de datos. `TradingViewData` especifica los métodos que proporcionan acceso a los datos e 'Interval' concreta los formatos temporales admitidos.

La extracción de datos de las criptomonedas se realiza en un fichero Python. Además de hacer uso de los métodos propios de la clase `TradingViewData()` se emplea la librería `pandas` para generar un archivo CSV a partir de los datos extraídos.

La práctica se ha desarrollado en *VisualStudio*, un editor de código fuente ligero y eficiente que se ejecuta en local. Se ha optado por este editor por la necesidad de trabajar en local al tener que clonar el repositorio para acceder a la herramienta `TradingView`.

6. Diseño de la solución

Desde el fichero de extracción de datos se accede a la clase `TradingViewData()`, que realiza la conexión a la herramienta `TradingView`. Dentro de un bucle `for` se utiliza una de las funciones propias del `downloader` `get_hist` para descargar la información para cada criptomoneda.

Se itera sobre todas las criptomonedas cuya información se quiere recopilar aplicando la función propia `get_hist` para descargar la información contenida en las velas japonesas de los últimos 4 años con un intervalo diario. Antes de guardar los datos en un archivo CSV se transforman los instantes temporales de cada instancia recolectada, que están en segundos, a un formato de fecha.

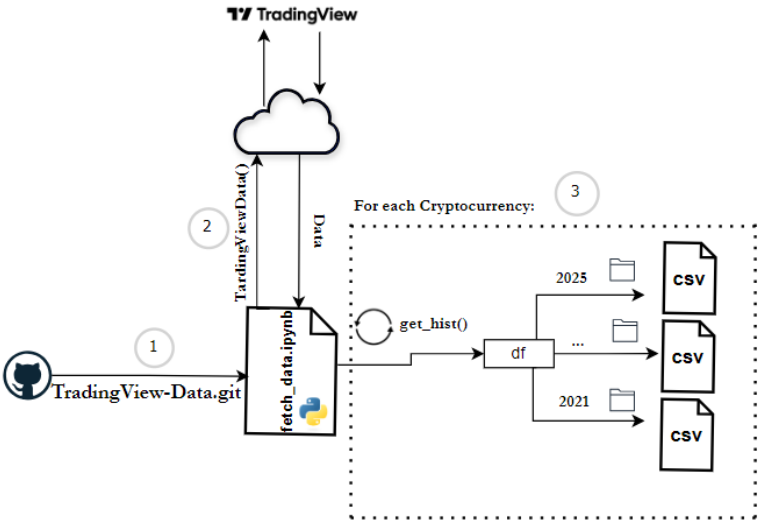

Figura 1. Funcionamiento de la Solución

7. Pruebas realizadas

El correcto funcionamiento se ha comprobado mediante distintos *prints*, que muestran secuencialmente las órdenes que va ejecutando el *script*. Además, el contenido de los archivos CSV se ha verificado manualmente para evaluar el contenido de los mismos.

Resultados

A continuación, se procede a estudiar los resultados obtenidos durante la ejecución de la práctica y cómo se ha diseñado la jerarquía de carpetas.

8. Descripción de los resultados

Con la metodología anterior se ha obtenido un fichero CSV con los campos `date`, `open`, `high`, `low`, `close` y `volume` para cada una de las criptomonedas. Para poder acceder de manera ordenada a los datos se ha establecido la siguiente jerarquía de carpetas:

- `Crypto_Historical_Data`
  - AAVEUSDT
    - 2021
    - 2022
    - 2023
    - 2024
    - 2025
  - ADAUSDT
  - BTCUSDT
  - DOTUSDT
  - ETHUSDT
  - SHIBUSDT
  - SOLUSDT
  - XLMUSDT
  - XRPUSDT

La carpeta `Crypto_Historical_Data` actúa como carpeta principal y contiene toda la información ordenada en subcarpetas. Se ha creado una carpeta para cada una de las criptomonedas para facilitar la identificación y selección de sus datos. Además, internamente cada criptomoneda se ha subdividido en los años recabados 2021, 2022, 2023, 2024 y 2025. Aunque se trate de una jerarquía un tanto genérica, al desconocerse las tareas futuras que se deberán desempeñar sobre los datos, esta estructura permite de manera sencillo realizar cualquier tipo de cambio para adecuarla a futuros requerimientos.

Puesto que se va a trabajar en HDFS para almacenar los datos de manera confiable y escalable, e implícitamente se va a usar *Hive* para interactuar con dichos datos, la partición interna de las criptomonedas en tablas más pequeñas distinguidas por año aporta muchos beneficios. La principal ventaja se manifiesta en la mejora del rendimiento, acelerando los tiempos de consulta y reduciendo la carga en el sistema. Al dividir la información en tablas particionadas únicamente es necesario escanear las tablas implicadas y no el conjunto completo de datos.

9. Pantallazos de la Ejecución

En este apartado se muestra la ejecución de la solución propuesta. En la Figura 2 se muestra la conexión a `TradingView` para extraer los datos de cada criptomoneda y su posterior almacenamiento por años en distintos archivos CSV.

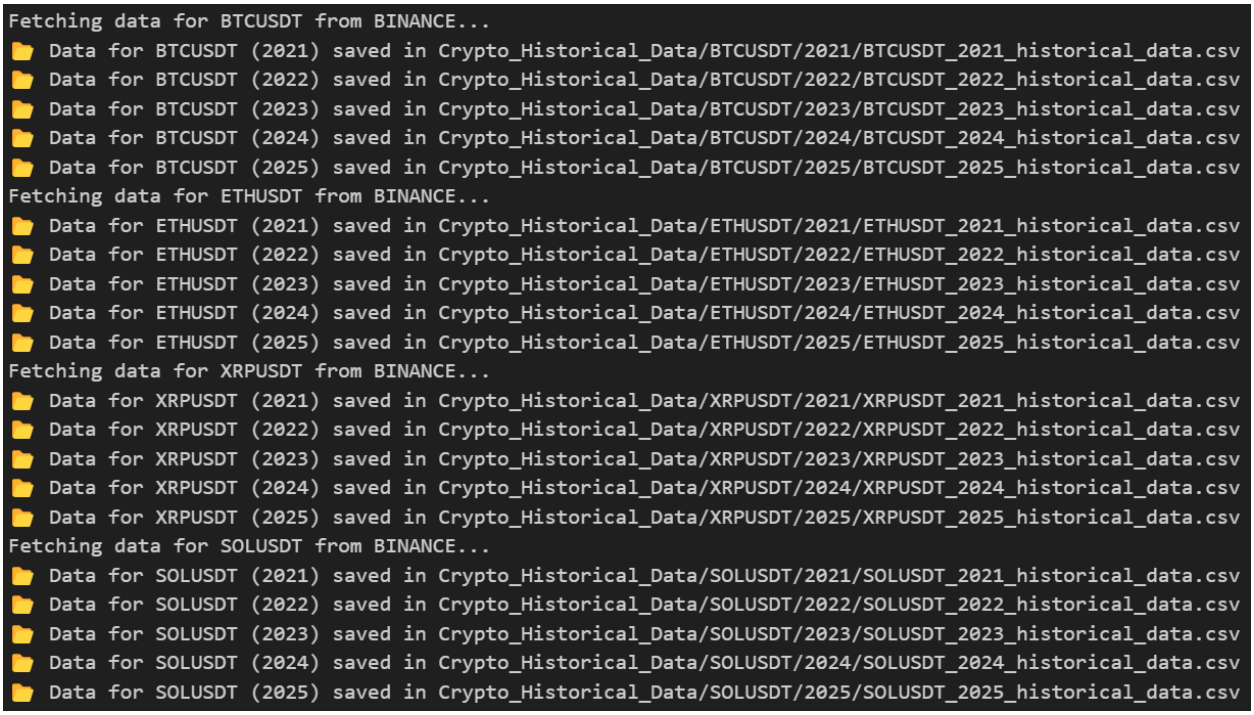

Figura 2. Recolección y Almacenamiento de Datos de `TradingView`

10. Discusión de los resultados

La jerarquía de carpetas y el formato de archivos de los datos históricos de las 10 criptomonedas se corresponde con lo previsto. Cada criptomoneda consta de 5 carpetas correspondientes a los valores de los últimos años. Dentro de cada una de estas carpetas se ubica un archivo CSV con los datos correspondientes de campos `date`, `open`, `high`, `low`, `close` y `volume`.

Conclusión

Por último, se van a resumir los principales hallazgos y aprendizajes obtenidos durante la práctica, y la relevancia de la solución implementada.

11. Resumen del Proceso

Para la adquisición del histórico de los datos de las criptomonedas y su almacenamiento se han seguido los siguientes pasos.

1. Identificación del problema: Adquisición desde herramienta `TradingView` de los datos históricos de 10 criptomonedas.
2. Diseño de la arquitectura e implementación: Clonación del repositorio con el `downloader` de `TradingView` y desarrollo en Python de un script para hacer uso del `downloader`, acceder a los datos y almacenarlos en la jerarquía de carpetas diseñada.
3. Pruebas y análisis: Ejecución del código y comprobación manual del contenido de las carpetas y los archivos CSV.

12. Principales Logros

La solución implementada ha mostrado su eficacia para adquirir los datos históricos de cualquier criptomoneda mediante el downloader `TradingViewData()` y almacenar la información acorde a la jerarquía deseada. Para concluir se van a recorrer los logros más importantes alcanzados en base a los resultados que se han analizado en la sección anterior.

- Familiarización con la herramienta `TradingView`.
- Se ha logrado desarrollar una **solución funcional** que adquiere los datos históricos de las criptomonedas deseadas de `TradingView`.
- Adecuación del formato y contenido de los archivos CSV al diseño deseado.
- Almacenamiento exitoso en el diseño de jerarquía inicial de carpetas.