FINM 32500 Computing for Finance in Python

Project Part 3

Data User Experience Document

Name
Andy Andikko
Harrison Zhang
Matheus Raka Pradnyatama

Data User Experience Document

Data Discovery*: Describe how researchers find relevant datasets. Explain how the Data Catalog aids in efficient data exploration by categorizing data and providing metadata.*

Efficient data discovery is critical for researchers aiming to design and implement effective trading strategies in a timely manner. A well-structured DataCatalog significantly enhances this process by categorizing data and enriching it with metadata for efficient exploration and analysis.

From the perspective of data discovery, researchers should first ponder the metrics they would like to generate the trade signals. These metrics could range from technical indicators to sentiment scores. The next step would be identifying where to obtain the information necessary for those metrics.

- **Stock Prices**: Utilize databases like Yahoo Finance (yfinance) for historical and intraday stock prices.
- **Fundamentals**: Access company earnings reports, typically available from SEC filings or specialized financial platforms.
- **Financial News**: Extract data from news feeds, using tools for sentiment analysis to gauge the market's reaction to specific events like product launches, earnings forecasts, or mergers and acquisitions.

The **Data Catalog** is a centralized repository, organizing datasets in an intuitive, hierarchical structure while providing descriptive metadata for each dataset. Our implementation of the DataCatalog allows the tagging of any available dataset in the DataLake with a user-defined category or a preexisting repo of categories used by all analysts, which may include Equities or Economic Data etc. These are stored as a JSON file called categories.json in our implementation. By logically grouping datasets into categories, the catalog minimizes the time researchers spend searching for relevant information. For example, a researcher seeking historical Apple stock prices can directly navigate to the **Equities** category instead of searching through unrelated data, such as economic indicators.

Each dataset in the catalog is accompanied by metadata, providing essential details that enhance usability:

- **Data Source**: Indicates the origin of the dataset (e.g., Yahoo Finance, SEC filings, or a news feed).
- **Description**: Summarizes the dataset's content and purpose.
- **Key Statistics**: Includes computed metrics such as:
    - Highest and lowest historical prices.
    - Mean and standard deviation of stock prices.
    - Summary statistics like total rows, column names, and data types.

- **Timestamps**: Details the time range covered by the dataset, ensuring relevance to the researcher's analysis.

A well-designed **Data Catalog** streamlines the process of discovering and organizing datasets for financial research. Categorizing data and providing rich metadata minimizes searching time and ensures researchers focus on analysis and strategy design. Whether the goal is technical, fundamental, or event-driven analysis, the Data Catalog is an indispensable tool for accelerating research and enabling data-driven trading decisions.

Data Access and Retrieval: *Outline how the Data Lake and Data Catalog facilitate easy access to raw and processed datasets.*

Our **DataCatalog** class inherits from the **DataLake**, extending its functionality to create a robust framework for accessing and managing raw and processed datasets. By combining these two systems, researchers and analysts can efficiently explore, retrieve, and organize data for quantitative analysis and research.

The **DataCatalog** leverages the functionalities of the **DataLake**, offering a seamless interface for accessing datasets and their associated metadata. For instance, the get_metadata() function from the DataLake enables quick lookup of datasets already stored in the system. This is particularly useful in large-scale environments, such as financial institutions, where data is abundant and often needs to be quickly searched and utilized.

To illustrate this point, consider a Quantitative Researcher working in an organization with vast amounts of financial data. Using the DataCatalog Through the DataCatalog, the quant researcher simply searches for a specific keyword (e.g., "Apple stock price"). The system retrieves matching datasets and their metadata, including details such as the dataset name, author, and category. This streamlined access significantly reduces time spent searching for relevant datasets, allowing the researcher to focus on analysis.

To understand the functionalities of our DataCatalog, we need to understand how it integrates with the DataLake. The DataLake is a centralized repository for storing raw and processed datasets, ensuring seamless and efficient data access. It accommodates raw data, such as CSV files containing unprocessed stock prices, alongside processed datasets, such as calculated metrics and transformations. This versatility eliminates the need for predefined schemas, enabling researchers to transform and process data as needed.

Our DataLake is designed flexibly, utilizing **pickle files** as the primary storage format. Pickle files support structured data (e.g., data frames) and unstructured data (e.g., text or JSON-like objects). This flexibility ensures compatibility with a wide range of data types and formats.

Before storing data in the Data Lake, metadata is enriched to enhance usability and traceability. For example:

- If the data is a **data frame**, metadata includes details such as column names, data types, index names, and row counts.
- For **JSON-like or unstructured data**, metadata captures key structural information, such as item counts and data types.

All metadata is systematically stored in a separate **JSON file** (metadata.json) to ensure independent access and interoperability with other system architecture components, such as the **DataCatalog**. This structured approach streamlines data management, simplifies integration, and enhances the overall utility of the Data Lake.

Data Preparation and Transformation*: Explain how the Data Workbench supports transformations, such as cleaning, aggregating, and filtering data for analysis.*

The DataWorkbench integrates with its parent class DataLake, and plays a central role in preparing, transforming, and interacting with the DataLake. Our implementation of the DataWorkbench possesses several key features:

1) **In-Memory Storage:** The DataWorkbench enables quick access to datasets by allowing temporary in-memory storage. Datasets can be cached with associated metadata (e.g., storage timestamp, data properties) for faster retrieval during iterative data processing.
2) **Transformation Management:** The class allows users to define, register, and apply single and chained transformations to datasets. Transformations can be as simple as adding new columns or as complex as computing derived metrics like rolling volatility or technical indicators.
3) **Processing Logs:** The DataWorkbench maintains detailed logs of all transformations applied to a dataset, ensuring transparency and reproducibility. These logs include timestamps, transformation names, and relevant metadata.

Raw data often needs to be more consistent, missing values or outliers. DataWorkbench simplifies these tasks by enabling users to apply pre-defined cleaning functions. For instance, removing null rows or standardizing column formats can be implemented as custom transformations. The class supports aggregation tasks, such as grouping data by time intervals or symbols and calculating aggregate statistics like mean or sum. This is particularly useful for financial datasets where intraday data needs to be summarized by hour or day. Filters can be applied to extract subsets of data based on conditions. For example, a transformation function could filter rows where volume exceeds a threshold, ensuring that only high-activity periods are analyzed. Transformed datasets can be stored persistently in the DataLake or kept in memory for temporary use. By leveraging its parent class, the DataWorkbench allows seamless transitions between temporary and long-term storage.

***Analysis Integration**: Describe how data moves from the workbench to a researcher's analysis environment, supporting tasks like backtesting or event studies.*

- **Data Preparation in the Workbench**: The **DataWorkbench** enables cleaning, transforming, and filtering data to create analysis-ready datasets. Processed data can include adjusted prices, rolling averages, and cumulative returns. Data is stored either in memory for immediate use or in the **DataLake** for persistence.
- **Exporting Processed Data**: Processed datasets are exported from the Workbench to the DataLake for reuse and collaboration, enriched with metadata for easy discovery. Data can also be saved in formats like CSV, Excel, or Parquet for compatibility with external tools.
- **Loading Data into the Research Environment**: Researchers retrieve the exported data into their preferred tools or programming languages (e.g., Python) for further analysis.
- **Analysis and Modeling**: In the analysis environment, researchers perform tasks such as:
  - **Backtesting**: Simulating historical trading strategies to evaluate performance.
  - **Event Studies**: Assessing the impact of events like earnings announcements or policy changes on stock prices.
  - **Quant Models**: Integrate with Quant Models for more complex transformation tasks. Serve as an interface between Quant Models and DataLake.