1.2 - Data Gathering with Pandas

Giovanni Della Lunga
giovanni.dellalunga@unibo.it

Introduction to Machine Learning for Finance

Bologna - February-March, 2025

Financial Data Gathering using Pandas

Subsection 1

Pandas Datareader

Data Gathering with Pandas

Introduction

This manual provides an educational guide on how to use `pandas-datareader` to extract and analyze financial data from various online sources. The `pandas-datareader` library is a powerful tool that enables direct access to a wide range of financial datasets, making it an essential tool for financial analysts and data scientists.

Requirements

To follow along, ensure you have the following installed:

- Python 3.8 or higher
- pandas (latest version)
- pandas-datareader (latest version)
- matplotlib (optional, for visualization)

You can install these packages using pip:

```
pip install pandas pandas-datareader matplotlib
```

Data Gathering with Pandas

Why Use Pandas DataReader?

Pandas DataReader provides easy access to multiple financial data sources, including:

1. **FRED (Federal Reserve Economic Data)** (Macroeconomic indicators)
2. **Alpha Vantage** (Stock prices, forex, and technical indicators)
3. **Quandl** (Financial and economic datasets)
4. **World Bank** (Global economic indicators)

Data Gathering with Pandas

Basic Setup

Import Required Libraries

Begin by importing the necessary libraries:

```
import pandas as pd
import pandas_datareader.data as web
import matplotlib.pyplot as plt
from datetime import datetime
```

Data Gathering with Pandas

Downloading Macroeconomic Data from FRED

FRED provides access to economic indicators such as GDP, inflation, and unemployment rates.

```
gdp = web.DataReader("GDP", "fred", start, end)
print(gdp.head())
```

Multiple Economic Indicators

```
indicators = ["CPIAUCSL", "UNRATE"]  # CPI and Unemployment Rate
econ_data = web.DataReader(indicators, "fred", start, end)
print(econ_data.head())
```

Data Gathering with Pandas

Using Alpha Vantage (Requires API Key)

Alpha Vantage provides stock market data, forex, and technical indicators.

```
from pandas_datareader.av import AlphaVantageReader
api_key = "your_api_key"
data = web.DataReader("AAPL", "av-daily", start, end, api_key=api_key)
print(data.head())
```

Data Gathering with Pandas

Fetching Data from the World Bank

The World Bank provides global economic indicators such as GDP growth and inflation rates.

```
data = web.DataReader("NY.GDP.MKTP.CD", "wb", start=2020, end=2023, country="US")
print(data.head())
```

Data Gathering with Pandas

Visualization

Plot Economic Data

```
gdp.plot(title="GDP Over Time")
plt.xlabel("Date")
plt.ylabel("GDP (Billions USD)")
plt.show()
```

Data Gathering with Pandas

Saving and Exporting Data

Save to CSV

```
data.to_csv("economic_data.csv")
```

Load from CSV

```
loaded_data = pd.read_csv("economic_data.csv", index_col=0, parse_dates=True)
print(loaded_data.head())
```

Data Gathering with Pandas

Example Projects

1. Economic Indicator Correlation Analysis

```
correlation = econ_data.corr()
print(correlation)
```

2. Inflation and Unemployment Rate Trends

```
econ_data.plot(subplots=True, title="Economic Indicators Over Time")
plt.show()
```

Subsection 2

Yahoo Finance

Using the yfinance python library

Data Gathering with Pandas

Introduction

This manual provides an educational guide to using Python's pandas library, in conjunction with other tools, to extract and analyze financial data from Yahoo Finance. The guide assumes you have a basic understanding of Python and some familiarity with data analysis.

Requirements

To follow along, ensure you have the following installed:

- Python 3.8 or higher
- pandas (latest version)
- yfinance (latest version)
- matplotlib (optional, for visualization)

You can install these packages using pip:

```
pip install pandas yfinance matplotlib
```

Data Gathering with Pandas

Why Yahoo Finance?

Yahoo Finance is a widely used platform for accessing financial market data, including stock prices, historical data, and company financials. By using the `yfinance` Python library, we can programmatically access this data and process it with pandas.

Data Gathering with Pandas

Basic Setup

Import Required Libraries

Begin by importing the necessary libraries:

```
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
```

Enable Pandas DataFrame Options (Optional)

For better visualization of DataFrames:

```
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', 20)
```

Data Gathering with Pandas

Downloading Historical Data

Single Ticker

To download historical data for a single stock:

```
# Download historical data for Apple (AAPL)
apple_data = yf.download("AAPL", start="2020-01-01", end="2023-01-01")
print(apple_data.head())
```

This returns a pandas DataFrame containing open, high, low, close, volume, and adjusted close prices.

Multiple Tickers

To fetch data for multiple stocks:

```
tickers = ["AAPL", "MSFT", "GOOGL"]
data = yf.download(tickers, start="2020-01-01", end="2023-01-01")
print(data.head())
```

This returns a multi-index DataFrame where the first level of the index contains the tickers.

Data Gathering with Pandas

Extracting Specific Information

Closing Prices

To extract the closing prices for all tickers:

```
close_prices = data["Close"]
print(close_prices.head())
```

Resampling Data

Resample the data to a monthly frequency:

```
monthly_data = apple_data.resample("M").mean()
print(monthly_data.head())
```

Calculating Daily Returns

```
daily_returns = apple_data["Adj Close"].pct_change()
print(daily_returns.head())
```

Data Gathering with Pandas

Advanced Data Manipulation

Merging DataFrames

If you have additional data sources to combine:

```
other_data = pd.DataFrame({"Date": ["2023-01-01", "2023-01-02"], "Extra": [1, 2]})
other_data["Date"] = pd.to_datetime(other_data["Date"])
merged_data = pd.merge(apple_data.reset_index(), other_data, how="left", on="Date")
print(merged_data.head())
```

Filtering Specific Date Ranges

```
filtered_data = apple_data[(apple_data.index >= "2022-01-01") & (apple_data.index <= "
print(filtered_data)
```

Data Gathering with Pandas

Visualization

Plotting Closing Prices

```
apple_data["Close"].plot(title="Apple Closing Prices")
plt.xlabel("Date")
plt.ylabel("Price")
plt.show()
```

Comparing Multiple Stocks

```
close_prices.plot(title="Closing Prices of Selected Stocks")
plt.xlabel("Date")
plt.ylabel("Price")
plt.legend(tickers)
plt.show()
```

Data Gathering with Pandas

Real-Time Data

While Yahoo Finance data is not truly "real-time," you can fetch the latest prices:

```
live_data = yf.Ticker("AAPL").history(period="1d")
print(live_data)
```

Handling Missing Data

Yahoo Finance data may have missing values. Handle them with pandas:

```
# Fill missing values with the previous value
apple_data_filled = apple_data.fillna(method="ffill")

# Drop rows with missing data
apple_data_dropped = apple_data.dropna()
```

Data Gathering with Pandas

Working with Company Financials

You can also access company financials such as income statements, balance sheets, and cash flows:

```
apple = yf.Ticker("AAPL")

# Income Statement
print(apple.financials)

# Balance Sheet
print(apple.balance_sheet)

# Cash Flow
print(apple.cashflow)
```

Data Gathering with Pandas

Saving and Exporting Data

Save to CSV

```
apple_data.to_csv("apple_data.csv")
```

Load from CSV

```
loaded_data = pd.read_csv("apple_data.csv", index_col=0, parse_dates=True)
print(loaded_data.head())
```

Data Gathering with Pandas

Example Projects

1. Stock Price Correlation Analysis

Analyze the correlation between the daily returns of multiple stocks:

```
returns = close_prices.pct_change()
correlation = returns.corr()
print(correlation)
```

2. Portfolio Performance

Simulate a portfolio's performance:

```
weights = [0.4, 0.4, 0.2]  # Allocation weights for AAPL, MSFT, GOOGL
portfolio_returns = (returns * weights).sum(axis=1)
portfolio_cumulative = (1 + portfolio_returns).cumprod()
portfolio_cumulative.plot(title="Portfolio Performance")
plt.show()
```