Data Science Notes

Barclays Research Data & Investment Sciences blog, where we dive into some of the frequently asked questions about data science and how it affects investing.

BARCLAYS

22 August 2022

We’re Open-Sourcing Our Alt-Data Tools

Investment Sciences
Ryan Preclaw
+1 212 412 2249
ryan.preclaw@barclays.com

Data Science
Adam Kelleher
+1 212 526 5697
adam.kelleher@barclays.com

Diverse data sources are a valuable input into the investment research process. Many small data vendors have become popular in recent years. Alongside traditional data such as financial filings, earnings call transcripts, and others, these data have created a rich landscape of sources for research. At the same time, they’ve created additional complexity. To leverage these inputs, researchers need to be adept at using modern tools for data analysis, including databases and cloud data stores (e.g., Amazon’s S3). A single project might require SQL queries, boto get requests, Spark read operations, manipulating local excel sheets, and more.

To navigate this complexity, Barclays’ Research Data Science team has built a tool set aimed at hiding the differences between these data sets and data stores from the user, and instead exposing an extremely simple interface to the analyst. We built this tool for ourselves, and it is a major contributing factor to our more-than-linear rate of productivity growth with time. When we work with a new data set, usually as part of a research project, we “encapsulate” the ETL (extract, transform, load) code and add it to the tool. This makes it a push-button capability that is ready the next time we need to use it. We can usually perform the same operation again later with just a single line of code.

We believe these tools will be useful for others who do financial research. They create higher quality control for queries that often otherwise only exist on analysts’ computers. They enable reproducibility by standardizing pre-processing on broadly available data sets.

Our software is designed to be extensible. It’s divided into sections by data source, so it’s easy to add code to support new data vendors or sets. Commercial and academic users will likely find the vendor data set readers useful. As more modules become available, we hope commercial and academic researchers, as well as the general public, will find the open-data readers useful as well.

We invite contributions of new data modules and queries within existing modules from the Python community. This might include researcher contributions to make their data more available and work more reproducible; vendor contributions to make their data sets easier to on-board; and student, data scientist, and developer contributions as they increase their productivity by encapsulating past work for easier re-use.

Scope

The scope for this tool is narrow. It is meant to provide data reading and simple data manipulation capabilities. The data analysis tool on which it is built, Python's Pandas library, can be too verbose for some simple operations. We expand its functionality when common operations for finance become verbose or error-prone using the "analysis" namespace in our tool.

Our data readers are narrowly scoped as well. Initially, we're releasing only classic finance data sources. Gradually, we'll add to those by releasing open-data and vendor data readers as well. We invite contributors to reach out to us directly before contributing to verify that their contributions are in scope for the library.

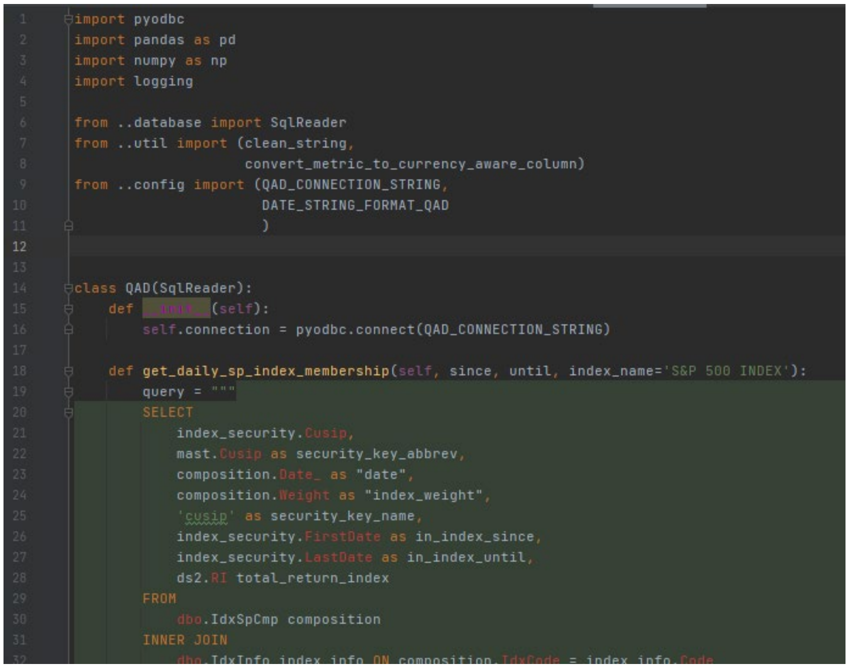
Technical Description

The design of our tool is meant to abstract data sources from the end user and to present a standardized interface for working with financial data. These goals are achieved by two different layers: a data access layer and an API layer.

The data access layer consists of a collection of modules, with an example in Figure 1. Each module corresponds to one collection of data sets in one data store, usually representing a vendor's data. For example, the Refinitiv module represents a collection of data products distributed by Refinitiv that are stored in an SQL database. A separate module represents a set of earnings call transcripts and presentations from Refinitiv, which we store as unstructured documents on AWS's S3.

FIGURE 1

Part of a data layer module for data products distributed by Refinitiv



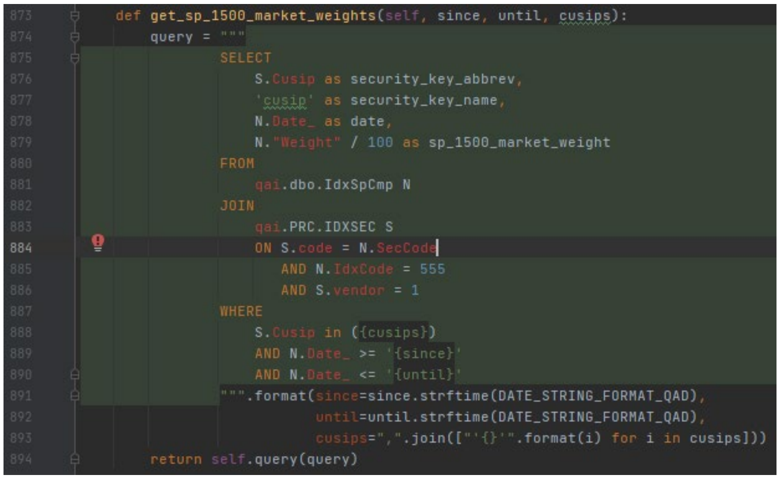
```
import pyodbc
import pandas as pd
import numpy as np
import logging

from ..database import SqlReader
from ..util import (clean_string,
                    convert_metric_to_currency_aware_column)
from ..config import (QAD_CONNECTION_STRING,
                      DATE_STRING_FORMAT_QAD
                      )


class QAD(SqlReader):
    def __init__(self):
        self.connection = pyodbc.connect(QAD_CONNECTION_STRING)

    def get_daily_sp_index_membership(self, since, until, index_name='S&P 500 INDEX'):
        query = """
        SELECT
            index_security.Cusip,
            mast.Cusip as security_key_abbrev,
            composition.Date_ as "date",
            composition.Weight as "index_weight",
            'cusip' as security_key_name,
            index_security.FirstDate as in_index_since,
            index_security.LastDate as in_index_until,
            ds2.RI total_return_index
        FROM
            dbo.IdxSpCmp composition
        INNER JOIN
            dbo.IdxInfo index_info ON composition.IdxCode = index_info.Code
```

Source: Barclays Research

These data layers provide a clean interface to raw data. When they provide access to an SQL database, for example, they usually expose a Python method that accepts parameters, then passes those to an SQL query that is sent to the database, as in Figure 2.

FIGURE 2

A data layer request for S&P 1500 market weights for a set of CUSIPS over a defined date range



```
def get_sp_1500_market_weights(self, since, until, cusips):
    query = """
            SELECT
                S.Cusip as security_key_abbrev,
                'cusip' as security_key_name,
                N.Date_ as date,
                N."Weight" / 100 as sp_1500_market_weight
            FROM
                qai.dbo.IdxSpCmp N
            JOIN
                qai.PRC.IDXSEC S
                ON S.code = N.SecCode
                   AND N.IdxCode = 555
                   AND S.vendor = 1
            WHERE
                S.Cusip in ({cusips})
                AND N.Date_ >= '{since}'
                AND N.Date_ <= '{until}'
            """.format(since=since.strftime(DATE_STRING_FORMAT_QAD),
                       until=until.strftime(DATE_STRING_FORMAT_QAD),
                       cusips=",".join(["'{}'".format(i) for i in cusips]))
    return self.query(query)
```

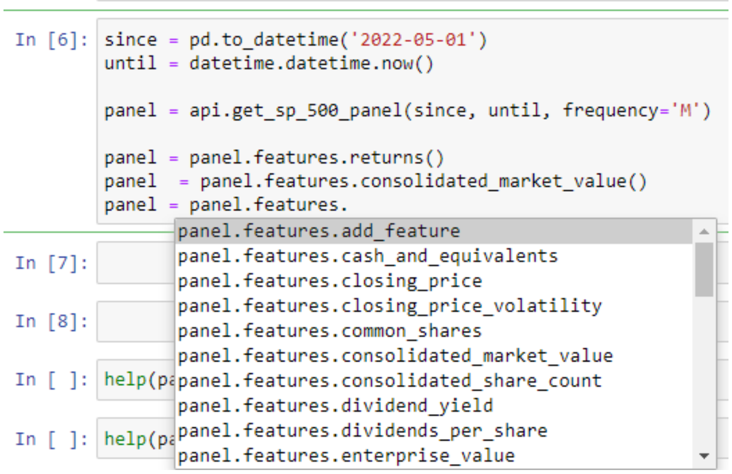
Source: Barclays Research

This serves a few nice functions. First, SQL queries are source controlled. Instead of having a team of analysts who are all writing (potentially slightly different) queries, there's a ground-truth query for a feature that has been reviewed by the team before adding to our shared repository. Second, this abstracts the data store from application code. If there are 100 applications that all use data returned by this method, but then the database changes, we have to make one change to accommodate the new database (in the data access layer), not 100 (in queries implemented in the application code). This remains true even if we change the underlying technology. For example, we could replace an SQL database with parquet files on S3 and use Spark SQL (or even PySpark code) to query it.

The API layer uses the data access layer to grab data, then reformats them into a standard format for analysis. It is meant to abstract data sources from the user, so it only exposes a set of intuitively named methods that are made to "just work." A typical example is given in Figure 3, where we are adding features to an S&P 500 panel. These "features" methods infer their parameters from the panel when possible, finding appropriate date ranges and universes of securities for the underlying queries.

FIGURE 3

We construct panels using simple, high-level methods; tab completion improves usability



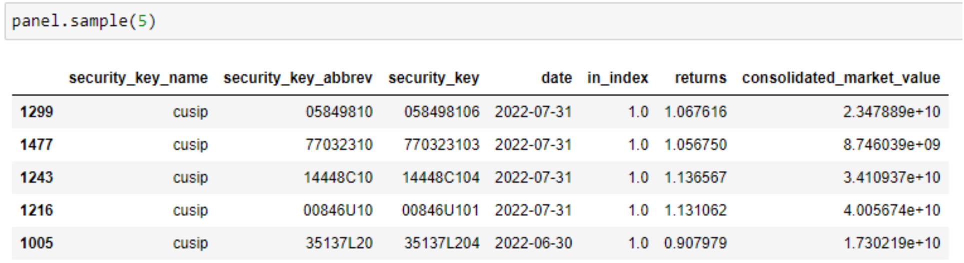
```
In [6]: since = pd.to_datetime('2022-05-01')
        until = datetime.datetime.now()

        panel = api.get_sp_500_panel(since, until, frequency='M')

        panel = panel.features.returns()
        panel  = panel.features.consolidated_market_value()
        panel = panel.features.
                       panel.features.add_feature
In [7]:                panel.features.cash_and_equivalents
                       panel.features.closing_price
                       panel.features.closing_price_volatility
In [8]:                panel.features.common_shares
                       panel.features.consolidated_market_value
In [ ]: help(pa        panel.features.consolidated_share_count
                       panel.features.dividend_yield
In [ ]: help(pa        panel.features.dividends_per_share
                       panel.features.enterprise_value
```

Source: Barclays Research

For us, this is a long-format panel of securities, as shown in Figure 4. The panel is a pandas.DataFrame, where each row corresponds to one security on a given date. The dates are expected to be regular, e.g., one for each day. Thus, a panel minimally has a date column and a pair of columns to provide security identifiers: the security key and the name of the security key (e.g., "CUSIP" for US securities or "SEDOL" for global securities). They typically also have features of these securities, where each additional column corresponds to a feature (R users will recognize this as "tidy data").

FIGURE 4

A sample from a panel



```
panel.sample(5)
```

	security_key_name	security_key_abbrev	security_key	date	in_index	returns	consolidated_market_value
1299	cusip	05849810	058498106	2022-07-31	1.0	1.067616	2.347889e+10
1477	cusip	77032310	770323103	2022-07-31	1.0	1.056750	8.746039e+09
1243	cusip	14448C10	14448C104	2022-07-31	1.0	1.136567	3.410937e+10
1216	cusip	00846U10	00846U101	2022-07-31	1.0	1.131062	4.005674e+10
1005	cusip	35137L20	35137L204	2022-06-30	1.0	0.907979	1.730219e+10

Source: Barclays Research

We usually want to analyze a universe that is specified at a single point in time, such as the S&P 500 index's securities. Since we use a panel, all assets exist in the dataframe at each point in time. We include an indicator for whether the asset is a member of our cross-sectional analysis universe, which is the "in_index" flag. That way, selection processes are more explicit, and we're set up to begin modeling selection biases (e.g., survivorship bias, selection into the index) using these panels.

This setup is right for a number of applications. A panel is a common data structure for econometrics and social sciences. The long format makes it formatted correctly for machine learning, as the popular sklearn API takes data in long format.

The simplicity of the interface makes data logistics trivial, so analysts can focus their time and mental energy on the aspects of their work where their unique skills are additive, rather than on rote data logistics. There are also a number of more technical benefits from a data infrastructure perspective.

Technical Benefits

Separating Storage and Compute

Modern tools separate data storage and compute. This contrasts with the last generation of “data lakes”, where all data live local on a Hadoop (or similar) cluster, and the data are analyzed on that cluster as well.

With the older model, there was contention when many analysts needed to use the cluster at the same time. They would have to wait to run their code, so their productivity was hampered by the available hardware. This was resolved when cloud services became popular for analytics. Clusters can now be created as ephemeral, on-demand resources and scaled up or down, depending on analysts’ needs. This requires a different solution for data storage because a new cluster won’t automatically have all of the data available. Instead, data are now more commonly stored in large-scale stores such as S3, Redshift, BigQuery, SQL databases, and other more specialized products.

Balancing Technology Fit with Accessibility

Many types of data deserve specialized infrastructure. Text data might be accessed through ElasticSearch or in a less structured form such as flat files in S3. Geolocation data might be stored in a geodatabase, with special location-based indexing and geographic data types and operations. APIs store data behind REST interfaces, with optimizations such as caching in place to speed up response time. Some data might be in real-time data stores, where queries must always return the newest data when they’re run.

In this varied data ecosystem, we think it makes sense to have a strict division between analysis and data storage. That function is performed by our data access layers. These are custom-built for a specific data set and store, and they standardize the data into a simple pandas dataframe before returning it to the user.

This flexibility comes with its own technical requirements. Some data sets might be very large. Analysis operations can require a lot of compute power and memory. We take the view that analysis should be arbitrary, not something usually off-loaded to a data store. We make sure our analysis environments are properly scaled for a task (usually Amazon compute clusters) and make our queries as simple and readable as possible. This contrasts with writing arbitrarily complex SQL queries and doing as much as possible in the database before returning the data. We do leverage database indexing, but generally do joins in-memory at analysis time using PySpark or Pandas.

This latter point isn’t something we take lightly. PySpark especially is not optimal for key-based operations. With large-scale data sets such as geolocation data, we usually do a lot of highly optimized pre-computation (such as place attachment, a geometric join operation) with our data access layer. We’d like to push this task onto a geodatabase, but the instance size to keep so much data (many terabytes) live would be extremely expensive. That brings us to a rule of thumb: if a database is needed to scale an operation, the data set is too big to make it worth storing in one. Instead, we generally build indexes outside of a database (e.g., with GeoSpark) and keep the data separate from compute.

This choice also allows us to join structured and unstructured data. If we can recognize a company name in the text of a raw document, we can map it to a security identifier and join it to a panel of securities data. This allows us to do arbitrary join operations where there may even be machine learning components (company recognition).

Disclaimer:

BARCLAYS

This communication has been prepared by Barclays.

"Barclays" means any entity within the Barclays Group of companies, where "Barclays Group" means Barclays Bank PLC, Barclays PLC and any of their subsidiaries, affiliates, ultimate holding company and any subsidiaries or affiliates of such holding company.

DATASETS

The licence for the tool/library described above does not extend to any of the content of the datasets with which the tool is designed to function or any such related services. An appropriate contract with the relevant data vendor is required in order to use this tool and for the tool to function. This tool is not affiliated, endorsed or in any way connected with any of the proprietors of the relevant data sets or any other third party.

CONFLICTS OF INTEREST

BARCLAYS IS A FULL SERVICE INVESTMENT BANK. In the normal course of offering investment banking products and services to clients, Barclays may act in several capacities (including issuer, market maker and/or liquidity provider, underwriter, distributor, index sponsor, swap counterparty and calculation agent) simultaneously with respect to a product, giving rise to potential conflicts of interest which may impact the performance of a product.

NOT RESEARCH

The information provided does not constitute 'investment research' or a 'research report' and should not be relied on as such, although it may contain references to views or information published by the Barclays Research department. Investment decisions should not be based upon the information provided.

BARCLAYS POSITIONS

Barclays may at any time acquire, hold or dispose of long or short positions (including hedging and trading positions) and trade or otherwise effect transactions for their own account or the account of their customers in the products referred to herein which may impact the performance of a product.

FOR INFORMATION ONLY

This information has been prepared by the Research Department within the Investment Bank of Barclays. The information, analytic tools, and/or models referenced herein (and any reports or results derived from their use) are intended for informational purposes only. Barclays has no obligation to update this information and may cease provision of this information at any time and without notice.

NO OFFER

Barclays is not offering to sell or seeking offers to buy any product or enter into any transaction. Any offer or entry into any transaction requires Barclays' subsequent formal agreement which will be subject to internal approvals and execution of binding transaction documents. The products mentioned may not be eligible for sale in some states or countries, nor suitable for all types of investors.

NO LIABILITY

Neither Barclays nor any of its directors, officers, employees, representatives or agents, accepts any liability whatsoever for any direct, indirect or consequential losses (in contract, tort or otherwise) arising from the use of this communication or its contents or reliance on the information contained herein, except to the extent this would be prohibited by law or regulation.

NO ADVICE

Barclays is not acting as a fiduciary. Barclays does not provide, and has not provided, any investment advice or personal recommendation to you in relation to any transaction and/or any related securities described herein and is not responsible for providing or arranging for the provision of any general financial, strategic or specialist advice, including legal, regulatory, accounting, model auditing or taxation advice or services or any other services in relation to the transaction and/or any related securities described herein. Accordingly Barclays is under no obligation to, and shall not, determine the suitability for you of the transaction described herein. You must determine, on your own behalf or through independent professional advice, the merits, terms, conditions and risks of any transaction described herein.

NOT A BENCHMARK

The information provided does not constitute a financial benchmark and should not be used as a submission or contribution of input data for the purposes of determining a financial benchmark.

INFORMATION PROVIDED MAY NOT BE ACCURATE OR COMPLETE AND MAY BE SOURCED FROM THIRD PARTIES

All information is provided "as is" without warranty of any kind. Because of the possibility of human and mechanical errors as well as other factors, Barclays is not responsible for any errors or omissions in the information contained herein. Barclays is not responsible for information stated to be obtained or derived from third party sources or statistical services. Barclays makes no representation and disclaims all express, implied, and statutory warranties including warranties of accuracy, completeness, reliability, fitness for a particular purpose or merchantability of the information contained herein.

PAST & SIMULATED PAST PERFORMANCE

Any past or simulated past performance including back-testing, modelling or scenario analysis contained herein is no indication as to future performance. Past performance is not necessarily indicative of future results. No representation is made as to the accuracy of the assumptions made within, or completeness of, any modelling, scenario analysis or back-testing and no representation is made that any returns will be achieved.

OPINIONS SUBJECT TO CHANGE

All opinions and estimates are given as of the date hereof and are subject to change. The value of any investment may also fluctuate as a result of market changes. Barclays is not obliged to inform the recipients of this communication of any change to such opinions or estimates.

NOT FOR RETAIL

This document is being directed at persons who are professional investors and is not intended for retail customer use.
Not For Further Distribution or Distribution To Retail Investors.
For Discussion Purposes Only.

IMPORTANT DISCLOSURES

For important regional disclosures you must read, visit the link relevant to your region. Please contact your Barclays representative if you are unable to access.
EMEA
https://www.home.barclays/disclosures/important-emea-disclosures.html.
APAC
https://www.home.barclays/disclosures/important-apac-disclosures.html.
U.S.
https://www.home.barclays/disclosures/important-us-disclosures.html.

CONFIDENTIAL

This communication is confidential and is for the benefit and internal use of the recipient for the purpose of considering the securities/transaction described herein, and no part of it may be reproduced, distributed or transmitted without the prior written permission of Barclays.

ABOUT BARCLAYS

Barclays Bank PLC is authorised by the Prudential Regulation Authority and regulated by the Financial Conduct Authority and the Prudential Regulation Authority and is a member of the London Stock Exchange. Barclays Bank PLC is registered in England No. 1026167 with its registered office at 1 Churchill Place, London E14 5HP.

COPYRIGHT