1. Overview

This document is to give a brief overview of the targets of this Proof of Concept and how it works.

The main goal was to see to what extend Dash/Plotly in python can be used to build a single webpage application that goes a step beyond displaying static data or data purely derived from single selections.

Since my background is linked to C# I chose to add interoperability with C# to see how well those languages could work together. In this case I have chosen to use a text file/command line interface, but conceptual only the interfacing layer changes when the communication would be wrapped in a rest API communicating via JSON format messages.

1. Prerequisites

To run the sample code the following is required:

* Python; developed on python 3.7.8 under windows,
* Visual Studio for compilation of C# code,
* Access to internet for downloading historical data,
* Read/write access to the working.

The additionally installed packages, also contained in the requirement file, are:

* dash
* dash\_bootstrap\_components
* pandas
* plotly
* scikit\_learn
* yfinance

1. Start-up

To start there is a batch script included, this will build the C# solution and start the python application. After opening an internet browser, the dashboard can be opened on localhost:888

1. Quick introduction

The main screen of the dashboard consists of 3 tabs, the first tab is a quick introduction, the second tab is the location where all the action takes place, and the third tab is for after-thoughts and possible additional directions.

After selecting the main tab, there are three additional tabs with the following functions:

* Data selection
* Apply domain specific language on the data
* Run a simple rolling window multiple linear regression on the data
  1. Data selection

On the data input tab, it is possible to download data from yahoo finance. For convenience several instruments, the Dow Jones index, is configured in the pull-down menu. Next to that it the interval can be selected, by default it selects hourly snapshots for which it will download the open, high, low, close and volume. As there is limitation on yahoo for datapoints that can be downloaded it will automatically adhere to this limit when downloading data.

After submitting the data is loaded and it is displayed on the right-hand side. Since it is a scatter plot the data can be plotted against all available properties.

Graphical user interface, chart

Description automatically generated

* 1. Domain specific language

On the domain language input-tab it is possible to set basic equations to create alphas. These alphas can serve as an input for trading opportunities.

In this example they are mostly random equation for demonstration purposes. In this example Alpha1 is the close price minus the close price of 5 steps ago. Both this text as well as the data from the previous page are fed into a C# program. This program parses the commands and applies this on the data. The output is the original input table enriched with alphas.

For visualization purposes the output of the domain language is plotted on a graph. With the dynamic features it is possible to plot any of the properties against any of the other properties.

Graphical user interface

Description automatically generated with medium confidence

* 1. Multiple linear regression

In the last step the output from the domain language is taken as input. On this tab the Alphas, also additional user defined ones, can be used as input values for the data analysis. In this case it is limited to a multiple linear regression.

After selecting the input, a rolling window can be selected. This rolling window indicates how much percentage of the data set will be used for an estimation. So, in the default (30%) case, the first 30% samples are used as input for the regression model and that model is used to predict a single step. In the next iteration the start point of the data is shifted with 1 step and from that point 30% of data is used and so forth till all the data has been predicted.

The selected alphas can be normalized, from each of the alpha the mean can be subtracted and/or it can be divided by the std. dev. This will create a zero centered distribution with a std. dev. of 1.

Finally, it can be chosen what to estimate, this can be either the return (close\_tmr – close)/close or the different directly (close\_tmr – close).

Again, this data can be plotted again in the scatter plot. This time it only plots the target value and the estimated (model output) value.

Graphical user interface, application

Description automatically generated

1. Coding thoughts

The code has been split over python and C#. For practical reasons python was used to build the front-end / application layer while C# focused more on the backend processes. Technically it is surely possible to write all code in python one of the goals was to see if it is possible to interact with different application easily with a web based front end. Arguably there are better communication protocols, for demonstration purposes this works fine.

In C# a few test cases have been added, those cases primarily cover the domain language processing as this is the most complex part of the program. The coverage of the test cases can be checked via generic testing tooling.

Additionally, the C# code can be further refactored to make classes smaller and be better testable as well.

In python the code has been structured in different file each containing roughly a tab page. This way it is easy to see the coherency between the pages and it makes understanding single pages relatively straightforward. As data is loaded and saved in a store there is a limitation on the amount of data that can be carried between the pages.

1. Final thoughts

It is interesting to see what can be done with simple tools. Although this is by no means production ready code, it has good demonstration practices that highlights the strengths and weaknesses of the different approaches and as such it was also used; to learn about best practices.