Station 1

Product description

Our service aims to offer an easy and inexpensive access to long-term investing based on modern portfolio analysis. Utilizing modern finance algorithms based on Markowitz's portfolio theory enables steady and predictable returns. A diversified approach into large-cap companies limits the risk due to the spread and quality of the companies. Due to the nature of the product people from all backgrounds can have access to our platform.

By specifying their risk appetite, the algorithm will select a number of companies, customised to the risk profile and investment horizon of the customer. To further increase comfort, the whole service can be fully delivered online.

As the algorithm is based price movements, psychologic biases that occur with mutual funds will be mitigated, while at the same time, costs can be minimised as less managers are needed.

To further decrease the risk, current economic data will be assessed to anticipate an economic decline, so that the portfolio allocation can be adapted to the level of economic uncertainty.

Product use and interface

The product is intended to be used for long-term investing, preferably for pension. By increasing the time horizon, short-term price fluctuations will be minimised and the total return will be more predictable. As the advantages of the program are its accessibility, simplicity and low cost, we definitely recommend to limit the use to online customers. By doing that, the cost can be minimized.

Customers can register on our platform and transfer money onto it. Then by simply selecting their risk profile, the algorithm automatically selects customised allocations for this customer and invests the money in the assets. Similar to pension funds, the interface should be very clean and only require minimal inputs. The product should be appealing to customers that do not want actively invest themselves and want to spend as little time with investments as possible. Just like on an average brokerage account, daily financial updates are provided so that customers can see their current balance.

Input Data

Our current version of the model requires past prices of the investments, as well as economic data and client data. The data will be supplied via an excel worksheet and will be read into pandas DataFrames. From there we can clean and work with the data.

The asset data will enable us to maximize the sharp ratio and minimize the variance. Developing a covariance matrix, we can simulate future returns by considering the price move changes of different assets that are colinear.

The customer will only input one variable, which will be the risk appetite. On a scale of 1 to 10, we can decrease the risk by increasing diversification and including assets with negative correlation.

Our current codebase is flexible and allows additional input with minimal modification. Hence, we can add further data such as unstructured data, as well as company valuation ratios to further finetune our model, which we plan to do in further iterations.

Calculations and storage

In Station 1, we will read all your data from excel worksheets to pandas DataFrames. As Station 1 is primarily devoted towards cleaning the data, there are barely any calculations in this step.

We prepare the DataFrames to build our features in Station 2 where we will calculate the relationship between the different assets and create a covariance matrix. Then by randomly simulating the progression of the assets based on past data, we can maximize the sharp ratio and minimize variance to build the efficient frontier. Also we need to calculate the mean annual return, as well as give every asset a risk score based on the client data and a score for the economic data.

Output data

We output 3 DataFrames that are based on the asset, client and economic data. These are cleaned, sorted and removed any irregular NAN values. Thereby, we can avoid any issues in the following Stations.

Both, the asset DataFrame, as well as the economic DataFrame use a date as index. The asset DataFrame shows the daily price change of all assets, by displaying working day dates in the index and using one column for every asset. Unlike that, the economic data summarises the results in quarters. As not all economic indicators are measured on a monthly basis, we decided to use quarterly averages for monthly changes, so that all rows of our DataFrame have sufficient input to create meaningful features with them. The columns of the economic DataFrame feature various economic indicators. In contrast to that, the client DataFrame has the client ID as index and has risk profile, as well as all allocations for different assets in percent as column headers.

Station 1 description

The Station 1's primary goal is to import the data into python/pandas objects and clean it, so that we can work with the data in the following stations. Namely, as our data is currently supplied in forms of multiple excel sheets, we import the data through pandas into pandas DataFrames. Once the data is loaded, we modify the data in a way that it is easy to work with in the following stages.

When importing data from Factset directly, depending on the format, one worksheet needs to be read in multiple times into different DataFrames. As the economic data used quarterly, as well as monthly data in different sections of one work sheet, it was easier to read in the date rows as headers, and only read the number of rows that are referring to these dates. As the dates were presented as headers, we had to transpose the DataFrame to make the economic indicators the columns. As the economic indicators had sub-headers without data, we needed to delete these columns. Lastly we can concatenate the two DataFrames with monthly and quarterly data und use the monthly average as quarterly value. By doing that we can delete all the monthly data, so that only rows with quarterly data remain.

Our asset data and client data did not need these major changes. For all three DataFrames, we need to manually set the index. As the economic and asset data will provide data points over time, date serves as index, while for the client data, the client id can be used as index. Then we should rename our column headers. For our asset DataFrame, we used the ticker plus the attribute that is described, hence the format will be "AAPL: daily return" for example.

We can already sort out all columns that we do not intend to use at any point. Particularly in the economic data, there are multiple economic indicators that do not provide predictive power for our purposes, so we can already remove them at this point.

While our asset and client data is already very clean, it is good practice to check all the DataFrames for values that do not fit the format of that column. As all our data is numerical, we check for NAN values, which need to be replaced or removed. In case of the asset data, we decided to replace missing values in the total return column and volume traded with 0, while the other values such as the market cap will be forward filled. This intuitively makes sense because if an asset does not trade on any given day, there is no volume and price change, but the market cap and the close price remain the same. In case of our client data, we check if the allocations of each client add up to 100%. If they do not, then the row with that client data will be removed as the data is invalid. After cleaning the data, we will sort it according to the index.

That concludes the cleaning of the data and the DataFrames can be returned by Station 1 and used as parameters for Station 2.

Inputs required to achieve the results relevant to this station

We use our 3 cleaned DataFrames for Station 2. Most importantly are the daily price changes of our assets. These inputs are needed to develop our covariance matrix, as well as to determine the mean return of our assets. Other than that, economic indicators with predictive power, such as CPI inflation, unemployment rate or annual GDP growth can give us a broad economic outlook and therefore, we can use these factors to adapt the risk level to the macroeconomic outlook. The client data, tells us what assets are preferred by people with certain risk appetite, as well as how strongly the risk profile impacts diversification.

While these inputs are sufficient to yield satisfactory results, we plan to increase the inputs to provide even better overall results. These additional inputs will likely include company data, such as operating income growth and valuation metrics. Also unstructured data will be used to build a sentiment analysis to catch the momentum of our assets.

What are the requirements in terms of data collection and data formats

Our model so far only works with structured data. Hence, it is best to use services that allow the download or import of structured stock price data such as yfinance (yahoo finance), factset or Bloomberg. Our model can be expanded to take additional unstructured data into consideration. However, unstructured data will serve as additional data rather than a replacement of the already existing structured data.

So far, it is necessary to insert the data in form of tables like an excel spreadsheet. As pandas has functions that allow very easy storage of structured data from excel or csv files into DataFrames, this will be the preferred way to store the majority of data. Thus, only structured data will be processed in our Station1 at this point but changes are already planned

Also, the data needs to be verifiable and freely available. Using paid services to acquire data is a method to speed up the data collection for future iterations of the project.

Core Features

The most important features for our model, are the ones that are based on the asset prices directly. By analysing the asset prices, we can determine the mean annual growth of each asset (Appendix 1), as well as how the assets are correlated with each other (Appendix 2). These two features allow us to build a portfolio based on modern portfolio theory.

To further improve on the algorithms we decided to include two more features. One is based on the client data and determines the risk of an asset based on how heavily weighted that asset appears in clients' portfolios with a large risk appetite (Appendix 3). This risk score helps us to increase returns by increasing the risk through giving these risky assets more weight in customers' portfolios. Lastly, we decided to use 6 macroeconomic indicators to assess the macroeconomic environment. When an indicator reaches a certain threshold, then the risk score for the current period will increase and if it increases too much, then we expect a recession. We added an additional column to the economic DataFrame, which is just a Boolean that will be True if the indicators indicate a recession (Appendix 4). In that case, we will automatically decrease our investors' portfolios' risk by a certain margin, so that losses in uncertain times can be mitigated.

Station 2

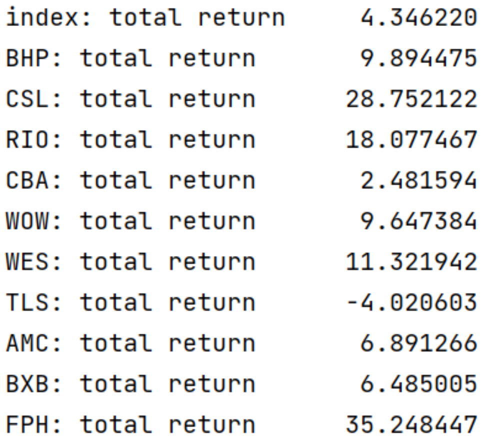
Station 2 description

Station 2's primary goal is to select our key features. We receive our cleaned DataFrames from Station 1 and will now use that data to create features. As described in the paragraph above, we came up with 4 main features that will be calculated in Station 2. Firstly, we will use the asset data to calculate the mean annual return of each asset. Then we create a covariance matrix. These two calculations can be easily automated due to pandas built-in functions. For the risk score of the assets, we multiply the risk profile of a customer with the allocation in an asset. Therefore, the assets with the highest weightings are the one that are heavily selected by clients with a high risk profile. This score is easy to calculate and accurate if we assume that the risk score of the client data is correct. Lastly, for the economic data, we used 6 indicators with high predictive power that are not too heavily correlated with each other. Each of the indicators has a certain threshold, such as annual GDP growth, which has its cutoff point at 0%, while the unemployment rate has its cutoff point at 7%. If the indicator reaches that threshold in any given period, then the recession score increases and the Boolean variable in the economic DataFrame will switch to True if multiple of these indicators reach their threshold.

All these features will be returned as DataFrames, Series, Dictionaries or other python objects, so they can be used in the following stations.

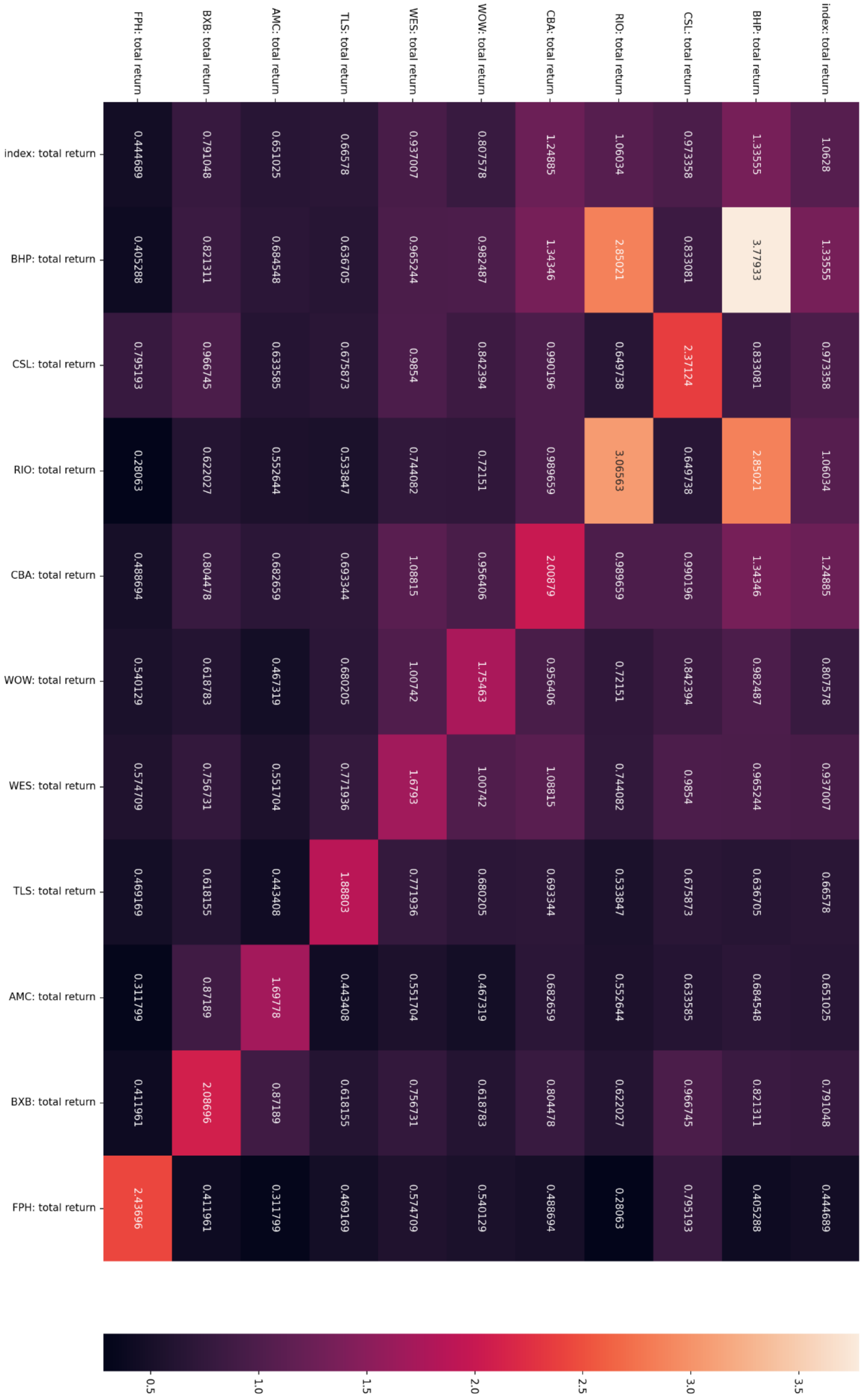
Appendix

Appendix 1: Mean annual return of assets



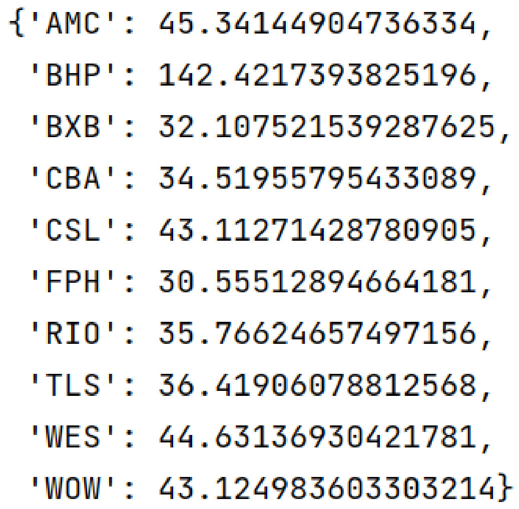
```
index: total return     4.346220
BHP: total return       9.894475
CSL: total return      28.752122
RIO: total return      18.077467
CBA: total return       2.481594
WOW: total return       9.647384
WES: total return      11.321942
TLS: total return      -4.020603
AMC: total return       6.891266
BXB: total return       6.485005
FPH: total return      35.248447
```

Appendix 2: Covariance matrix



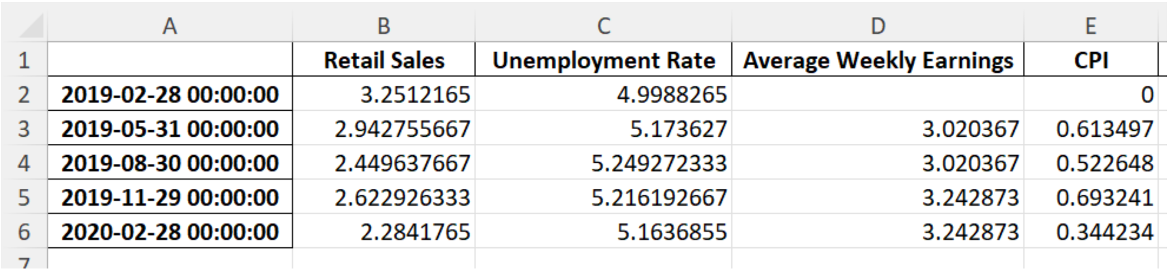
	FPH: total return	BXB: total return	AMC: total return	TLS: total return	WES: total return	WOW: total return	CBA: total return	RIO: total return	CSL: total return	BHP: total return	index: total return
index: total return	0.444689	0.791048	0.651025	0.66578	0.937007	0.807578	1.24885	1.06034	0.973358	1.33555	1.0628
BHP: total return	0.405288	0.821311	0.684548	0.636705	0.965244	0.982487	1.34346	2.85021	0.833081	3.77933	1.33555
CSL: total return	0.795193	0.966745	0.633585	0.675873	0.9854	0.842394	0.990196	0.649738	2.37124	0.833081	0.973358
RIO: total return	0.28063	0.622027	0.552644	0.533847	0.744082	0.72151	0.989659	3.06563	0.649738	2.85021	1.06034
CBA: total return	0.488694	0.804478	0.682659	0.693344	1.08815	0.956406	2.08879	0.989659	0.990196	1.34346	1.24885
WOW: total return	0.540129	0.618783	0.467319	0.680205	1.00742	1.75463	0.956406	0.72151	0.842394	0.982487	0.807578
WES: total return	0.574709	0.756731	0.551704	0.771936	1.6793	1.00742	1.08815	0.744082	0.9854	0.965244	0.937007
TLS: total return	0.469169	0.618155	0.443408	1.88803	0.771936	0.680205	0.693344	0.533847	0.675873	0.636705	0.66578
AMC: total return	0.311799	0.87189	1.69778	0.443408	0.551704	0.467319	0.682659	0.552644	0.633585	0.684548	0.651025
BXB: total return	0.411961	2.08696	0.87189	0.618155	0.756731	0.618783	0.804478	0.622027	0.966745	0.821311	0.791048
FPH: total return	2.43696	0.411961	0.311799	0.469169	0.574709	0.540129	0.488694	0.28063	0.795193	0.405288	0.444689

Appendix 3: Risk score of assets based on client data

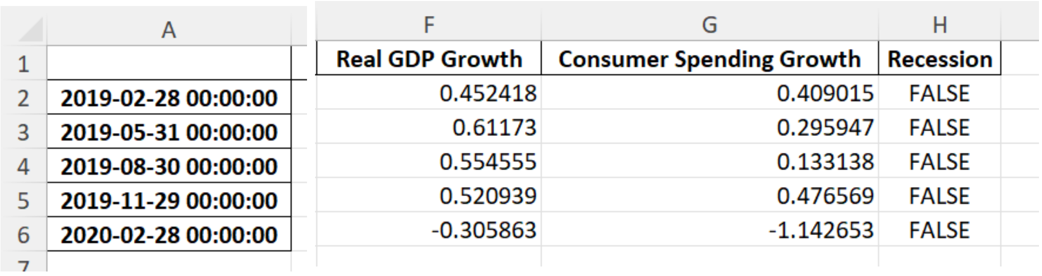


```
{'AMC': 45.34144904736334,
 'BHP': 142.4217393825196,
 'BXB': 32.107521539287625,
 'CBA': 34.51955795433089,
 'CSL': 43.11271428780905,
 'FPH': 30.55512894664181,
 'RIO': 35.76624657497156,
 'TLS': 36.41906078812568,
 'WES': 44.63136930421781,
 'WOW': 43.124983603303214}
```

Appendix 4: Economic data and recession indicator (DataFrame written to xlsx file)



	A	B	C	D	E
1		**Retail Sales**	**Unemployment Rate**	**Average Weekly Earnings**	**CPI**
2	**2019-02-28 00:00:00**	3.2512165	4.9988265		0
3	**2019-05-31 00:00:00**	2.942755667	5.173627	3.020367	0.613497
4	**2019-08-30 00:00:00**	2.449637667	5.249272333	3.020367	0.522648
5	**2019-11-29 00:00:00**	2.622926333	5.216192667	3.242873	0.693241
6	**2020-02-28 00:00:00**	2.2841765	5.1636855	3.242873	0.344234



	A	F	G	H
1		**Real GDP Growth**	**Consumer Spending Growth**	**Recession**
2	**2019-02-28 00:00:00**	0.452418	0.409015	FALSE
3	**2019-05-31 00:00:00**	0.61173	0.295947	FALSE
4	**2019-08-30 00:00:00**	0.554555	0.133138	FALSE
5	**2019-11-29 00:00:00**	0.520939	0.476569	FALSE
6	**2020-02-28 00:00:00**	-0.305863	-1.142653	FALSE