****

**FINTECH 536: ROBO-ADVISING**

**Build Lab 2c: Extending the robo workflow and data sources**

**Work completed in Build Lab 2a**

* Problem Definition
* Getting Started - Load Libraries and Dataset
  + Load Libraries
  + Load Dataset
* Data Preparation and Feature Selection
  + Preparing the predicted variable
  + Feature Selection-Limit the Feature Space

**Work completed in Build Lab 2b**

* Evaluate Algorithms and Models
  + Train/Test Split
  + Test Options and Evaluation Metrics
  + Compare Models and Algorithms
* Model Tuning and Grid Search
* Finalize the Model
  + Results on test dataset
  + Feature Importance
  + Feature Intuition

**BUILD LAB 2c**

**1.0 Introduction**

In Build Lab 2a and Build Lab 2b, we used Python to automate, and analyze, the first steps in the robo-advisor’s workflow – We collected the demographic data that describes a prospective investor and then analyzed, and derived, the prospect’s risk tolerance. As might be expected, income and net worth, followed by age and willingness to take risk, were the key variables that determined the risk tolerance of the people represented in the sample data.

In Build Lab 2c and Build Lab 2d, we will extend the basic robo workflow capabilities (capturing investor demographics and deriving risk tolerance), previously created to now include asset selection, portfolio creation, and performance reporting. Specifically, the new workflow will span:

**Build Lab 2c activities:**

* Loading and processing a new dataset of investor characteristics;
* Loading and processing a new dataset of asset-level timeseries data (from the S&P 500);
* Creating an interactive “user interface”;

**Build Lab 2d activities:**

* Applying the regression model (from project 1b) to predict the risk tolerance of the new investors;
* Using Markowitz mean variance analysis to allocate user-selected S&P 500 equities;
* Producing graphs to show the asset allocation and portfolio performance over time;
* Provide an url to access the interactive web page created by the Python code.

This extended robo-advisor workflow will, again, be built in Python and incorporate the “plotly dash” package. “Dash” is a Python framework for building graphically-oriented web applications. The additional Python code required to automate the steps to create the extended robo-advisor will be provided in a new Jupyter Notebook – as described below.

**2.0 Starting work – preparation steps**

**2.1 Ensure the code and model, from robo team technical project 1a and project 1b, are saved and ready for re-use**

Once saved, the investor demographic data, regressions to derive investor risk tolerances, analytical models, and results from team-technical project 1a and project 1b, can be “re-used” to produce pieces of the new robo workflow.

The following Python code should already be included at the end of the Jupyter Notebook submitted as your team’s technical project 1 deliverables:



The file created by the code module above, named *finalized\_model.sav,* will be processed by the new Python code and provide the reusable components from technical project #1.

**2.2 Ensure that your repository (directory), from Project 1, contains the data file, Jupyter Notebook, and saved model**

The directory you used as the repository for project 1, on your Mac or PC computer, should contain the following project 1a and project 1b files:

* Excel data input file: ***SCFP2009panel.xlsx***
* Jupyter Notebook: ***InvestorRiskToleranceAndRoboAdvisor.ipynb***
* Saved model (after successfully running Jupyter Notebook): ***finalized\_model.sav***

Ensure all three files are located in the same directory before proceeding.

**2.3 Download the two data files (CSV) and new Jupyter notebook from the “resources” folder on Sakai**

Log onto Sakai and go to our FINTECH 536 site. Navigate to the “resources” directory and then to the “Robo advisor project 2” sub-directory. Once there, you will see two files:

* CSV data file: ***InputData.csv***
* CSV file containing S&P 500 asset data: ***SP500Data.csv***
* ***Sample-code-project-1c.ipynb***

Download both files to the directory you used as the repository for project 1, on your Mac or PC computer.

Once the new data files and the new Jupyter Notebook have been downloaded, your repository directory should contain the following files:

* Saved model from project 1:
  + ***finalized\_model.sav***
* Three data files (one in Excel format and two in CSV format):
  + ***SCFP2009panel.xlsx***
  + ***InputData.csv***
  + ***SP500Data.csv***
* Two Jupyter Notebooks:
  + ***InvestorRiskToleranceAndRoboAdvisor.ipynb***
  + ***Sample-code-project-1c.ipynb***

**3.0 Launch your “Python” environment (Anaconda and Jupyter)**

First, launch Anaconda on your computer.

From Anaconda Navigator, select and launch Jupyter.

Once the Juypter environment is active, open your repository directory. You should see all the saved file from project 1, the three data files, and the two Jupyter Notebooks.

Use Jupyter to open the Jupyter Notebook named ***Sample-code-project-1c.ipynb***

**4.0 Install new Python libraries and import them into Jupyter environment**

The first code module will scan the Python libraries installed on your computer and, if additional new libraries are required, install the new libraries into your library repository. The new libraries will provide the pre-written Python libraries to create graphs and other “visual components.” If the Python code detects that these libraries has been previously installed, it will skip over each of these installation steps.

**4.1 Install new Python libraries on your computer**

Execute the following code cell. Note: You will not see any messages or output in the Jupyter environment. However, you will be able to see progress messages in your computer’s “terminal” window.



**4.2 Import Python libraries into the Jupyter environment**

Execute the following code cell. Note: You will not see any messages or output in the Jupyter environment. However, you will be able to see progress messages in your computer’s “terminal” window.

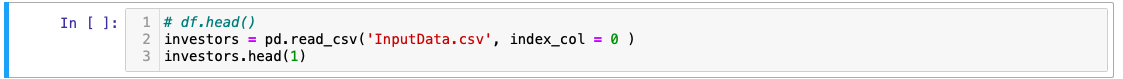
****

**5.0 Load investor and market data**

**5.1 Load investor data**

Check that the file containing the investor data has been downloaded into the correct directory. The file is named *InputData.csv*

Execute the following code cell.

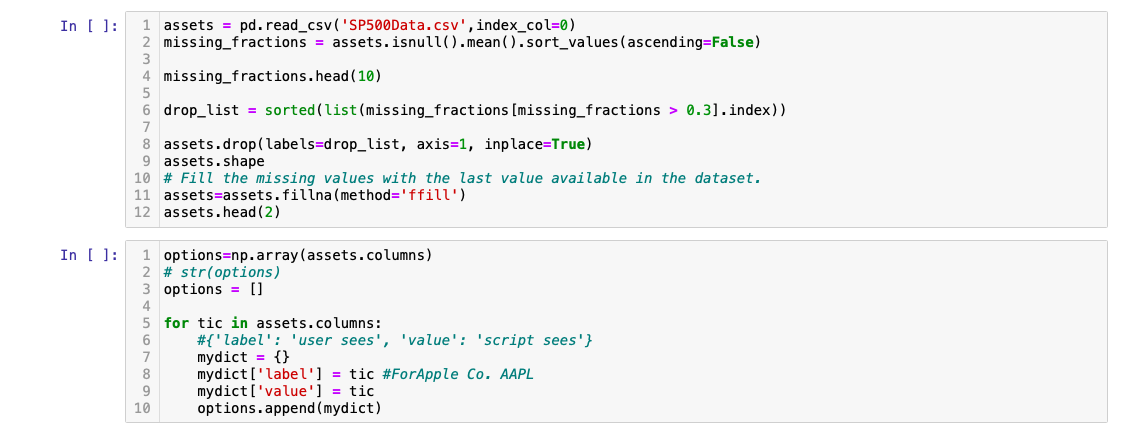


**5.2 Load market data**

This step will load and pre-process the S&P 500 equity timeseries data.

Check that the file containing the S&P 500 data has been downloaded into the correct directory. The file is named *SP500Data.csv*

Execute the following two code cells.



**6.0 Load the “style sheet” and create the “template” for the interactive user interface**

**6.1 Load the “style sheet” (*Sample-code-project-1c.ipynb*)**

The next code cell will load an open source “style sheet” that will be used to create and format the visual elements of the robo advisor’s graphical interface. These visual elements include text libraries, pre-written code to create common objects, and utilities to perform common actions. For example:

* Text libraries (fonts and spacing);
* Pre-written code for objects (tables, buttons, lists, forms)
* Utilities for actions (load, clear, and reload web pages and objects)
* Launch code modules and activate links

The “style sheet” hosted on the internet at:

<https://codepen.io/chriddyp/pen/bWLwgP.css>

It is loaded by running the following code cell:



**6.2 Create the “template” for the interactive data entry form and interactive user interface**

The next cell uses the “style sheet” and Python code to create a “template” for a dynamic web page that will:

* Present, and enable a user to change, the investor’s attributes

*Note: The following workflow steps and code will be created in project 1d (not in this project 1c)*

* Run the risk tolerance algorithm
* Select equities from the S&P 500 to use in portfolio allocation
* Run the Markowitz mean variance algorithm and construct an asset allocation for the selected equities
* Present graphs of the portfolio allocation and portfolio performance (returns and loses) overtime

Note: No output is “visible” when this code cell is executed.

After the “template” is created, additional code cells will be linked to the “template” to enable the user to: view and modify the investor’s attributes; apply the regression model (from project 1a and project 1b) to profile investor’s risk tolerance; select equity assets from the S&P 500; use the Markowitz mean variance analysis to construct a portfolio; use time series data to show the portfolio performance over time.

**END OF BUILD LAB 2c**