Problem Set #2
Business 35150
Winter 2024

The due date for this assignment is February 3, 8:30am
(upload on canvas)

Write up the solution clearly, using tables and graphs where appropriate. For example, if the question asks you to calculate or compute some numbers, report these numbers in a table or some other form that clearly displays the results. Do not just copy-paste the unformatted output from running your code. In addition to showing the results, you must also explain and describe your results in words. **Submit everything in one pdf file**. Write your name, and if you worked in a group, the names of **all group members**, on the top of the first page.

The data file `PS1data.xlsx` posted with this problem set contains time series of monthly returns for the nine asset classes we discussed in class and monthly returns on Treasury bills (`rf`). (Note that the numbers are all of end of month returns. For example, what's shown in the file in the row for 1/1/1980 are the returns during month January 1980.) As a first step, use the T-bill returns series to calculate excess returns for the nine asset classes. Wherever I write "returns" in the rest of this problem, I mean "excess returns."

The file `PS2data2.xlsx` contains monthly returns on Treasury bills (`rf`), monthly returns on the CRSP value-weighted stock index (`rvwind`), and a series of monthly repurchase-adjusted dividend-price ratios (`repadjdy`). (Again the return numbers are all of end of month returns. For example, what's shown in the file in the row for 1/1/1980 are the returns during month January 1980 and the repurchase-adjusted divided-price ratio at the end of January 1980.)

Please save the data files in the same directory as the python script (`PS2_2024.ipynb`) that I posted.

1. Have a look at the Part 1 section of the python script and try to understand what it does. You are welcome to use ChatGPT to get help in understanding the python language. The first part of the code in section Part 1 uses the full sample of data to estimate mean returns and the covariance matrix and plugs these into the mean-variance optimal portfolio formula. The second part performs an out-of-sample analysis in which mean returns and covariances are estimated in 20-year windows, and then used to calculate the optimal portfolio weights, based on these weights, a portfolio return in the first month that follows the estimation window. For example, the code estimates means and covariance in the 20-year window from Jan 1980 to Dec 1999, and then applies the resulting portfolio weights to the returns in month Jan 2000. It records the portfolio weights and resulting

portfolio return. Then it moves the estimation window forward by one month to Feb 1980 to Jan 2000, and so on. Answer the following questions:

(a) Cell #3: what is `np.tile(rf.reshape(-1, 1), (1, N))` doing?

(b) Cell #4: What do the `m`, `S`, `w`, `psr` variables in cell #4 represent?

(c) Cell #4: What is the purpose of `wr = w / np.sum(w)`?

(d) Cell #4: Does rescaling of weights via `wr = w / np.sum(w)` affect the Sharpe ratio?

(e) Cell #5: What does the `win` variable control? If `win` is increased, does this lower or raise the standard error of the quantity estimated by `m = np.mean(x, axis=0)`?

(f) Cell #6: Have a look at `wr = wr * phi + (1 - phi) * (1 / N)`. This is a shrinkage estimator for portfolio weights that is a bit different from the shrinkage estimator we considered in class. For which value of `phi` is there no shrinkage? For which value is there total shrinkage in the sense that portfolio weights are equal to the shrinkage target? And how would you describe the type of the portfolio that is the shrinkage target?

2. Now run all cells in the Part 1 section of the script and look at the output.

(a) Focus on the in-sample plug-in part (cell #4): Examine the Sharpe ratio that you get based on this calculation. If you had to advise an investor now (in 2024) about the level of *future* Sharpe ratio that can likely be earned from a portfolio based on the weights calculated in cell #4 with the plug-in approach, what would you say?

(b) Now look at the out-of-sample analysis (cells #5 and #6): Compare the Sharpe ratios to the Sharpe ratio you obtained in the in-sample analysis above. Is this result what you would have expected? Why?

(c) Compare the Sharpe ratio that you obtain in the out-of-sample analysis without shrinkage (cell #5) and with optimal shrinkage (cell #6). Why are they different? Why does shrinkage improve the Sharpe ratio?

(d) If you had to advise an investor now (in 2024) about an optimal asset allocation, which of the three weight estimates (from the in-sample, out-of-sample without shrinkage, out-of-sample with shrinkage analysis) would you choose?

3. Now let's modify the code in the Part 1 section. For this it is best if you start a new Jupyter notebook and copy over the code until the end of the Part 1 section. Then you can modify it in the new notebook. The modification we want to consider concerns the short positions. If you examine the weights that the optimization produced so far, they often involve negative weights for some asset

classes, i.e., short positions. One issue with these short positions is that it may be difficult or costly to establish a short position in some asset classes. And even when it's possible without too much cost, some investors may not want to, or may not be allowed to, take short positions. The proper solution in this case is to build the no-shorting constraint into the optimization when one looks for the optimal portfolio weights. But this requires numerical solution techniques that are beyond what we will do in this course. For this reason, we look at a more crude approach here: Let's add a modification in the code such that after the calculation of the mean-variance optimal portfolio weights, you simply truncate the portfolio weights by replacing the negative entries in the weight vector with zeros. (I recommend asking ChatGPT for a suggestion on how to replace negative weights with zeros in a numpy array). Then recalculate portfolio returns and Sharpe ratios for the

- in-sample analysis
- out-of-sample analysis without shrinkage,
- out-of-sample analysis with optimal shrinkage (recalculate the optimal shrinkage parameter using the truncated portfolio weights)

Look at how the Sharpe ratios differ from those you calculated with the initial version of the code in question (2.). Where do you see the biggest deterioration in Sharpe ratio due to the truncation of portfolio weights, in the in-sample or in the out-of-sample analysis? Why?

4. In this problem we look at stock market excess return predictability. However, unlike in the lecture, where we looked for predictability associated with the slow-moving price-dividend ratio, here we investigate in addition whether there are exploitable short-run trends, or "momentum," in the stock market. Have a look at the Part 2 section of the python script and try to understand what it does. Answer the following questions:

   (a) Cell #9: What is the purpose of this line in the code: `X_lr = sm.add_constant(lr)`

   (b) Cell #9: What is `model = sm.OLS(xr,X_lr)` doing?

   (c) Cell #10: Look at `w = np.array([1, lr[i + win + 1]]) @ b`. This is construction of a weight variable. What is the weight used for? What determines the value of this weight?

   (d) Cell #10: Look at `pr[i, j] = xr[i + win + 1] * (w * phi + (1 - phi) * 0.01)`. What is being constructed with this expression? There is some shrinkage involved here. What is the shrinkage target?

5. Now run all cells in the Part 2 section of the code and look at the output.

(a) Examine the coefficient estimates, t-statistics, and R^2 from the two regressions run in Cell #9. Which of the two variables, past returns or log P/D seems to be a stronger predictor?

(b) How would you characterize the result shown in the regression with past returns as predictor: is this better described as a "momentum" (or "trend") effect or a "reversal effect"?

(c) Look at the results of the out-of-sample analysis. Compare the results for the constant-weight strategy, the no-shrinkage strategy, and the optimal shrinkage strategy. Based on this comparison, what is your conclusion about the exploitability of trends in the stock market?

6. Now let's modify the code in the Part 2 section. For this it is again best if you start a new Jupyter notebook and copy over the code of the Part 2 section (and the stuff at the very top that loads packages and sets the directory). Then you can modify it in the new notebook. Two different ways of doing an out-of-sample analysis are: (i) a rolling fixed-window approach where the number of observations used to estimate a forecasting model is always kept constant, but the window is moved forward in time step by step; (ii) an expanding window approach where the starting date of the window that contains the observations used for estimation of the forecasting model is always kept the same and so with every step, one adds additional observations to the window, i.e., the window expands. The approach used in the code that I posted is a rolling fixed-window approach. We now want to check whether the market-timing strategy based on past returns could be improved by using expanding window approach instead of the rolling fixed-window approach.

   (a) What are (statistical) reasons why an expanding window approach could potentially do better, in terms of forecast performance, and ultimately the Sharpe ratio of our timing strategy, than a rolling fixed-window approach? What are reasons why it could potentially do worse?

   (b) Now modify the code to implement an expanding window approach and rerun the out-of-sample analysis. What do you find? What is your updated conclusion now about the exploitability of trends in the stock market?