Ricky Fonbuena
IS 520
December 2014
Dr. Meservy

Advanced Penny Stock Finder

Executive Summary:

Some stock traders like the low price and volatility of penny stocks, but finding good ones can be a pain because you want both low price and high price movement. Also, any stocks that do not have a chart available are useless for technical traders since all decisions are made according to the stock's charts. Stock screeners like those on *yahoo!finance* are problematic because they do not allow a very specific price range, and they list all stocks, even those with no price history to analyze, and even those with stocks valued below one cent. For my project I want to allow the user to choose a lower and an upper price limit and then have all stocks in that price range listed from *yahoo!finance*, along with each stock's company name, current price, and volatility (beta). That list will then be cleaned up to take out the stocks with no charts available, and hyperlinks will be added to link each stock to its respective chart on *yahoo!finance*. The list will then be converted into a table, which will allow the user to sort the list based on price, volatility, and alphabetically by company name. Each new search using the user form will add a new page to the current workbook that includes the table for that search. This project could be useful for anyone that practices technical trading, from beginners to more advanced traders. The program also extends to trading stocks in any price range, since a user form will be used to collect the desired price range. The end product is a concise list of usable, interesting stocks to analyze.

Implementation:

The methods and order I used to complete the project are as follows:

1. Scraping the web table from *yahoo!finance*
   I Scraped data from the stock screener for stocks with price between one cent and one dollar. Since it would only show me twenty stocks at once, I made a *for* loop to scrape the data for each page. Depending on how many stocks the desired price range woud yield, however, there were a different number of iterations for the loop to do. To figure out how many pages this would be, I used the *mid* function, which extracted the total number of stocks that my search yielded from one of the tables that I scraped from the web.The end result was a complete list of all stocks in the price range specified by the user.

2. Cleaning up the data
   Since nonvolatile stocks, stocks priced below one cent, and stocks with no

price charts are virtually useless for technical analysis, I looped through each row of data and cleared any row associated with a stock that did not have a chart available on *yahoo!finance*. At first I tried to delete these rows, but the iteration number on the loop was getting thrown off as rows were deleted. So I decided to clear the rows instead and then delete the blank rows. This required me to write another sub, and I found some of the needed code at *mrexcel.com*. The end result was a concise list of only the most volatile, interesting, and usable stocks.

3. Adding hyperlinks
   Since the remaining list included only stocks that had charts available online, I used another *for* loop to add a link to each stock's particular price chart on *yahoo!finance*. My reasoning for adding links instead of scraping each individual chart is that the program would take too long to run and there is also a lot of extra information in the link that most technical traders would not need. Also, this way the user can choose which charts to specifically look at, instead of needlessly having all of the charts in front of them.

4. Making a table
   In order to make the data sortable and easier to navigate, I turned the list into a table. I did this by recording a macro. While I was looking at the code from what I had recorded, I noticed that the recording had modified each row, one by one, without a loop. I felt very accomplished when I spotted that and created a loop instead. Hundreds of lines of code were rewritten as just four lines. The end result was a table that could be filtered and sorted to satisfy all of the user's preferences.

5. Comments
   I added a few comments to inform the user about some of the columns that could be ambiguous. I did this by recording a macro.

6. Creating the User Form
   I tried to create a user form earlier in this process but failed, so I resorted to using input boxes to get the user's desired price range. However, the project just looked sloppy that way so I decided to try the user form again. This time I succeeded. The user form gathers the user's minimum and maximum desired price, and when "OK" is clicked, the whole rest of the project runs; the web is scraped and the table is ultimately created. Each time the user enters a different price range and clicks "OK" a new page is created that includes a table with that price range to analyze.

Learning and Conceptual Difficulties:

A list of some things that I learned:

- Scrape data from multiple pages with a loop.
- In recorded macros, identify redundancies and make code more efficient.

- An object ”querytable” exists in VBA.
- **Create a user form from scratch.** (In previous projects the user form was already coded to initialize properly and we were given a skeleton to work with.)
- Add working hyperlinks.
- Delete blank rows to consolidate information.
- The property ”UsedRange” was very, very helpful.
- creating a table from data.
- Debugging code. (Knowing what to look for.)

While the things I learned are very useful, another thing that this project has afforded me is the chance to really solidify my skills in the subjects that we covered this semester in class and in homework. Some of these include web queries, all kinds of loops, public and private subroutines and variables, user forms, etc. There were a few difficulties that I encountered. Honestly, everything that I mentioned in *”A list of some things that I learned”* was difficult for me, but I spent enough time to figure it out. As I said before, building a user form from scratch was hard. I gave up on it and then tried again and got it.

Assistance:

No one assisted me in this project. The only help I had came from observing code online from websites such as *mrexcel.com*.