Trevor Feller
MBA 614: VBA
Final Project

Executive Summary

For my final project, I decided to develop an attribute & price harvester for a company called Oxspring Paul. Oxspring Paul is a product distribution company that specializes in e-commerce and media based sales channels. They primarily deal in watches. As part of their business, they have to manually collect attribute & pricing data on new watch products that they start to sell in order to create product profiles. In addition, they perform regular pricing checks on items currently in their inventory to ensure optimal marketplace competitiveness. Currently, to collect these two data types, an employee will manually look up each item on Amazon.com to harvest the available attribute data and also establish a marketplace pricing benchmark on initial product build. To maintain accurate pricing data, it is required that employees cycle through the entire existing inventory and retrieves pricing data on a regular basis. Collecting attribute & pricing data represents a very large amount of employee labor hours. With nearly 2000 new unique products being processed per year and over 4000 current SKUs that must be price checked regularly, it is estimated that new attribute & price data harvesting requires about two full-time employees. In addition, the pricing portion is just not being done as well as it should be done. Ideally, a price check would be performed on every item, every day. Oxspring Paul would be lucky if pricing data was updated on every item once a month.

In developing the attribute /price harvester, I had the following goals:

1. I wanted the harvester to be capable of operating on just one input, the Amazon unique identification number or ASIN. I chose this identifier because Oxspring Paul can easily get a list of the product ASINs of every item they sell on Amazon. In further developments, I would like to build out a version that can find an ASIN based off just the manufacturers model number.
2. Collect the following pricing attributes: List Price, Price, Buy Box Price, Buy Box Shipping, Buy Box Total Price, New Market Competitors, New Market Price, Amazon Sales Rank, Rank Category and Selling Status. In addition, I wanted it to collect the following product attributes: Brand Name, Model Number, Part Number, Model Year, Item Shape, Display Type, Clasp, Metal Stamp, Case Material, Case Diameter, Case Thickness, Band Material, Band Length, Band Width, Band Color, Dial Color, Bezel Material, Bezel Function, Calendar, Special Features, Movement, Water Resistant Depth and Dial Window Material Type.
3. I wanted the harvester to be able to diagnose what type of page (status) it was receiving from Amazon and process it accordingly. Amazon products pages are formatted in five possible statuses: In Stock, Available from These Sellers, Currently Unavailable, Only Available at These External Websites and Oops. The formatting of the page is designated based on the current selling status of the product or in the case of the Oops format, the lack of a product with the ASIN used.
4. Lastly, I wanted the program to save the sheet of data as its own .csv file in a folder (Data) based on the current location of the workbook. This .csv file would later be automatically processed by a SQL based system that would import and analyze the data.

Implementation Document
What the finished product does:
The finished product works off this procedure:

```
Sub runparsePriceData()
Prep
Do
PrepLoop
ResetVariables
ParsePriceData ASIN
Loop Until ActiveCell.Value = ""
ExportSheet
End Sub
```

Here is a detailed description of each of the procedures within Runparsepricedata:

Runparsepricedata
Prep

1. Harvester diagnoses the active sheet and switches to the appropriate sheet if needed.
2. Harvester selects that appropriate beginning cell.

PrepLoop

1. Now on the correct sheet and in the correct cell, the harvester sets RowCount to the active cell row and stores the ASIN as ASIN.

ResetVariables

1. In preparation to start taking data, the harvester resets the appropriate data so on each loop the data is not added to previous data.

ParsePriceData (ASIN)

1. The harvester uses the input ASIN to create a direct URL and goes directly to the Amazon product level page.
2. The harvester saves the product level pages source as Parser1 and saves an html copy in a log file location based on the current location of the workbook.
3. The harvester then diagnoses the status of the item by checking for unique data points based on each status, logs the status type and then routes to different appropriate contingency codes based on the diagnosed status.
    a. The first check is to see if the URL returned a valid product (Oops status check).
    b. If this test is passed, the data is then checked to diagnose whether the data is statuses In Stock or Available from these Sellers.
    c. If it is neither of these, an additional check is performed to diagnose if the data type is Currently Unavailable.
    d. If once again, the data is not diagnosed, a final check is performed to diagnose if the data is status type Only Available at External Websites.
    e. If the data goes through all these checks and is not properly diagnosed, it is classified as “Error”, the status is updated in the data table and the program moves to the next item.
4. Each Status type does the following:
    a. In Stock
        i. Harvests: List Price, Price, Buy Box Price & Buy Box Shipping (these data points require a contingency to accurately get the data every time based on whether the shipping cost is free or not), New Market Competitors & New Market Price (Once again, a contingency is required to properly collect these two points of data every time), Rank, Rank Category.

ii. While the data is harvested, the data is also cleaned and stored consistently.
iii. The data and the status type are then input into the appropriate data fields on the data sheet.
iv. Runs AttributeHarvester (Explained in detail below)
v. Lastly, the harvester prepares itself for the next loop.

b. Available From These Sellers
i. Harvests: New Market Competitors & New Market Price, Rank (Requires a contingency to properly retrieve the data consistently), Rank Category.
ii. While the data is harvested, the data is also cleaned and stored consistently.
iii. The data and the status type are then input into the appropriate data fields on the data sheet.
iv. Runs AttributeHarvester (Explained in detail below)
v. Lastly, the harvester prepares itself for the next loop.

c. Currently Unavailable
i. Harvests: Rank (Requires a contingency to properly retrieve the data consistently) and Rank Category.
ii. While the data is harvested, the data is also cleaned and stored consistently.
iii. The data and the status type are then input into the appropriate data fields on the data sheet.
iv. Runs AttributeHarvester (Explained in detail below)
v. Lastly, the harvester prepares itself for the next loop.

d. Only Available At External Websites
i. Harvests: Rank (Requires a contingency to properly retrieve the data consistently) and Rank Category.
ii. While the data is harvested, the data is also cleaned and stored consistently.
iii. The data and the status type are then input into the appropriate data fields on the data sheet.
iv. Runs AttributeHarvester (Explained in detail below)
v. Lastly, the harvester prepares itself for the next loop.

e. Oops
i. Harvests: Rank (Requires a contingency to properly retrieve the data consistently) and Rank Category.
ii. While the data is harvested, the data is also cleaned and stored consistently.
iii. The data and the status type are then input into the appropriate data fields on the data sheet.
iv. Lastly, the harvester prepares itself for the next loop.

AttributeHarvester

1. Uses unique code anchors to find the above mentioned attributes and stores the data for each in an array as string data.
2. Selects “Reference Data” sheet and finds “Brand Name:”
3. Once Brand Name is found, it uses the row count to move down the appropriate number of cells and then enters the data from the array by moving over 1 cell at a time.

ExportSheet

1. The harvest selects all the data now in the “Reference Data” sheet and saves it to the “Data” folder located in the current workbook location.

What I Learned:

Honestly, everything above represents new learning for me. I loved this class! As can be seen from my final project, I really spent the majority of my time working with web harvesting. I still have quite a few things that I didn't do that I just simply didn't have time to do, not because I didn't know how. Here are the things I think I would like to add to the project in the future:

1. I want the harvester to be less dependent on the ASIN. I would like to add a function that when given a model number or search term, it goes and harvests the ASIN and uses an algorithm to diagnose the appropriate ASIN.
2. I wouldn't mind figuring out a way to do the attribute portion in a more scalable way. Currently it looks for watch specific terms. I would like it not to be hard coded, but be able to harvest the available attributes regardless of product category. I think I could figure it out given time.

Additional things that once again I didn't have time to implement and also don't know how:

1. I would like to build this in a language that would work well on a server. I would want portions (price harvester portion) of this to fire off daily for Oxspring Paul.
2. I would really like to add the ability to mask or divide the work load so that websites couldn't get wise to all this data coming from one IP address. I wouldn't want them to throttle or shut down my access.