Paper 2830-2018

Automate the Tedious Stuff – Use Cases in Taking Your Time Back

Danni Bayn Ph.D., FCB Chicago

ABSTRACT

With all of the buzz around big data, machine learning, and predictive analytics, it is easy to forget just how much of our day-to-day is spent on the far less glamorous task of data preparation. Analysts, researchers, and scientists alike spend a lot of their time trying to get the data into a workable state before they can begin getting to understand the story and the insights hidden within. This paper outlines three programs that were developed to handle common data wrangling tasks (e.g. cleaning, filtering, transforming, and merging). These programs took 60 hours of manual effort and replaced them with automated processes that run in less than 30 minutes.

Code was developed with ActiveState Perl 5.24.2 and SAS® 9.4 TS1M4 running under Windows 10. Novice users will be able to use all three programs, while further development and modification may require intermediate to advanced SAS expertise.

INTRODUCTION

There is so much data out there for us to tap into, that it is hard to wade through it all to find that glimmer of insight or that pattern in the noise that helps us to understand what is truly important; to help us understand what the data is trying to tell us. For many industries, data is coming from many different sources and in many different forms. This leads to the all too common task of data wrangling, where analysts, researchers, and scientists have to massage, merge, and clean data before they can get to the task of trying to understand it. This processing step is often a tedious, manual effort that takes a lot of time and is extremely error prone. Depending on the amount of data and how much cleanup is needed, any errors that do occur can be difficult to impossible to track down.

This paper discusses three use cases where we automated the data processing for three separate projects to save time, increase quality, and reduce frustration.

USE CASE 1 – EXCEL EXTRACT AUTOMATION

Often times, data extracts will come in the form of multi-sheet Excel files. The data in these files are frequently arranged as wide tables with nested rows, with each worksheet containing data for a different metric. Data wrangling for a single one of these files takes about two hours to clean and filter the data by metric, convert the wide tables to tall ones, merge the data across worksheets, and to do a final quality assurance (QA) check.

For this particular project, we were aiming to do a cross-category comparison of different brands at different retailers. This meant we had 17 different files with differing numbers of brands and retailers in each. While a single file takes about two hours to process, additional files take even longer as they need to be merged with the previous files. The data wrangling component of this project was estimated to take approximately 50 hours. Additionally, since there were so many manual steps, data errors were common and, due to the size of the data and its complexity, tracking down these errors was extremely time consuming.

To automate this step, we developed a SAS program that ran each file through a Perl script to save each worksheet into a separate CSV file, and then to clean and remerge them all into a single stacked data set ready for deeper analysis.

The development and testing of this program took approximately ten hours and it takes less than ten minutes to run. The cross-category comparison gets refreshed twice a year and this program saves our

team 49+ hours each time. Additionally, the program can be easily modified for other projects that require merging of multi-sheet excel workbooks.

USE CASE 2 – CSV DATA MERGE

There are many analytic platform services available on the market today with varying levels of functionality (depending on your subscription level). One of our clients primarily uses Adobe Analytics and has the service plan that allows data extracts for up to five metrics at a time. Very rarely are we ever looking at only five metrics.

Our team was asked to create a dashboard that displayed all of client's key metrics (about a dozen) that could be viewed at different levels of granularity (like customer segment, geography, campaign, etc.). This required 108 separate extracts to CSV files that then needed to be merged to create a single data set for the dashboard. The process to manually merge these CSV files took about five hours. Unfortunately, this manual remerging process had a very high error rate. About a third of the time, the process to track down the error took longer than starting the remerge process over. Due to these limitations, updates to the dashboard were limited to once a quarter and led to some very frustrating nights at work.

To automate this CSV merge, we created a very short program in SAS using the macro facility. The code (with comments) is less than 100 lines long and took less than an hour to develop. It runs in under a minute and eliminates the possibility of manual error during the merging process. The process is so simple that junior team members who have not used SAS previously can be quickly trained on how to run the program going forward.

USE CASE 3 – API TO PDF AUTOMATION

Any time we are looking to explore new industries or talk to new clients, senior leadership asks our team to provide quick-read reports using YouGov BrandIndex data to compare different brands within a given industry. These reports are meant to communicate basic brand positioning and YOY growth as compared to competitor brands. As straightforward as these requests are, they need to be very visual and they need to be produced very quickly (turn-around times are often less than a day).

The original process for developing these reports was to go through YouGov's online portal to extract the necessary data (for a variable number of brands at different levels of granularity). Like in Use Case 1, this required multiple data extracts that were all in the form of multi-sheet excel files with wide tables and nested scores. To define the brand set, extract the data, merge the data, and then to create the final quick-read report, took a minimum of four, very tedious, hours.

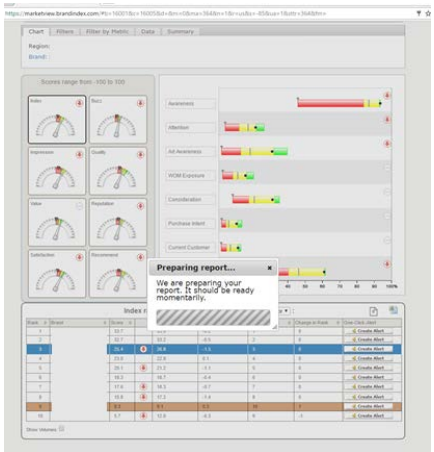

Figure 1: YouGov's BrandIndex Online Portal

Using SAS we were able to create an end-to-end solution to automate these reports. SAS was used to connect to BrandIndex's API tool to get the data and then the Output Delivery System was used to create the final PDF. Using the macro facility we were able to package the entire process into a few user-facing macros so that all members of the team, regardless of programming experience, could complete the request on their own and would only need to specify a maximum of six inputs (e.g. target brand, start date, end date, etc.). From beginning to end, the program takes less than 10 minutes to get a year's worth of data for approximately 15 brands and then about 20 seconds to create the PDF.

This has led to three distinct benefits: (1) Turnaround time is drastically reduced, so senior leadership often gets the information they need in less than an hour from the initial request, (2) Additional views of the data, such as customers versus non-customers, are trivially easy to produce if more detail or granularity is needed, (3) All members of the team can use the program to create these reports, whereas, previously, there were only a few team members that had the expertise and know-how to create these within the tight turnaround time.

HOW MUCH TIME HAVE WE GOTTEN BACK?

Taking into account development time, our cost savings in the first year of implementation is 99%, 94%, and 62.5% respectively across the three use cases. They have saved our team 180+ hours of manual effort within the first year alone. Going forward, we are saving over 4½ weeks of manual, tedious, labor every year.

	Use Case 1 YouGov Profiles	Use Case 2 CSV Merge	Use Case 3 New Business Brief
Manual Cost / run	50 hours	5 hours	4 hours
Frequency	2x / year	4-6x / year	5 pitches / yr x 3 slices / pitch (Limited by bandwidth)
1 time Dev Cost	10 hours	1 hour	20 hours
Run Cost	10 min	1 min	10 min
Cost Savings / yr	~99 hours	20-30 hours	57+ hours

CONCLUSION

SAS can do some amazing things in terms of visualizing insights, predictive modelling, and advanced analytics, but sometimes we forget just how powerful it can be in the less glamorous task of helping us get the data ready for the fancy stuff.

Automating the tedious stuff has huge benefits in terms of time savings, increased accuracy, and improving the quality of our time spent doing work. Through automation, we spend less time fighting with the data trying to get it to talk, and more time listening to the story it is trying to tell.

REFERENCES

BrandIndex. *API Documentation.* https://api.brandindex.com/v0

Clay, Ted. (2006). *Tight Looping with Macro Arrays.* SUGI 31.

Henry, Joseph. (2016). *REST at Ease with SAS: How to Use SAS to Get Your REST.* SAS Global Forum 2016.

Kummer, Daniel. (2014). *Toe to Toe: Comparing ODS LAYOUT and the ODS Report Writing Interface.* SAS Global Forum 2014.

SAS Institute Inc. *Sample 41880: Read all files from a directory and create separate SAS data sets with unique names.* http://support.sas.com/kb/41/880.html

Zhang, Sijian. (2012). *ODS Report Writing Interface Makes Our Reporting Simple and Better.* SAS Global Forum 2012.

CONTACT INFORMATION

If you are interested in more detail on the projects described above, would like the code used, or if you have any comments or questions, please contact the author at:

Danni.Bayn@fcb.com
875 N Michigan Ave
Chicago, IL 60611

SAS® and all other SAS® Institute Inc. product or service names are registered trademarks or trademarks of SAS® Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

APPENDIX

Full code and walk through for Use Case 1 – Excel Extract Automation is in the "When All You Have is a Hammer" (Bayn, est. 2018). Below is the code for Use Case 2 and the end-user facing code for Use Case 3. Full code for Use Case 3 – API to PDF Automation is available upon request.

Use Case 2 – CSV Data Merge:

```
%let path = C:\Example;
/*************************************\
  Reference files for merchant names
\*************************************/
  PROC IMPORT OUT= ref  DATAFILE= "&path.\name_reference.csv" DBMS=DLM REPLACE;
    DELIMITER=',';
    GETNAMES=NO;
    guessingrows=32767;
  RUN;

/*************************************\
  Read Directory
\*************************************/
  filename DIRLIST pipe 'dir "C:\Example\Raw Extracts" /b ';
  data dirlist ;
     infile dirlist lrecl=200 truncover;
     input file_name $100.;
  run;
  proc sql noprint;
      select distinct
        prxchange('s/(.+)\.csv/$1/', -1, file_Name  )
        into :t separated by ' '
      from dirlist
    ;quit;

%macro loopit (
  FILE);
  /*************************************\
    Import file
  \*************************************/
  %let tfile = C:\Raw Extracts\&file..csv;
    PROC IMPORT OUT= t1  DATAFILE= "&tFILE" DBMS=DLM REPLACE;
      DELIMITER=',';
      GETNAMES=NO;
      guessingrows=32767;
    RUN;
    %let x = %FileAttribs(&tfile, createdt);  /*Get file creation date    */
  /*************************************\
    Get list of metrics in this file
      (each A, B, C, etc files have different metrics)
  \*************************************/
    data _null_;
      set t1;
      if prxmatch('/.+Selected Metrics.+/', var1) then do;
        %do j = 1 %to 5 %by 1;
          call symput("var&j.", strip(scan(var2, &j, ',')));
        %end;
      end;
    run;
  /*************************************\
    Clean up data file –
       Specific to project extracts
  \*************************************/
  data &file (drop=var1-var7 d m y a);
```

```
    format date datepulled date9.;
    set t1 (firstobs=24);  /* Remove headers   */
    retain merchant datepulled;
    Merchant = "&file";
    DatePulled = input(scan("&x", 1, ':'), date9.);
      d =put(input(prxchange('s/(\d+?).(\w+).(\d+)/$1/', -1,var2),8.), z2.);
      m =upcase(prxchange('s/(\d+)-(\w+)-(\d+)/$2/', -1,var2));
      y =prxchange('s/(\d+)-(\w+)-(\d+)/$3/', -1,var2);
      a=compress(d ||M ||y);
      Date=input(a, date9.);
    &var1 = input(var3, 8.);
    &var2 = input(var4, 8.);
    &var3 = input(var5, 8.);
    &var4 = input(var6, 8.);
    &var5 = input(var7, 8.);
    output;
  run;
%mend loopit;

/************************************\
  Back to program
\************************************/
%do_over(values=&t, macro=loopit);
data prep;
   format merchant $25.;
   set %do_over(values=&t, phrase=?);
run;

  proc sql;
  create table  out  as
    select distinct
      a.merchant as filename
      , b.var2 as merchant
       , a.*
    from prep as a
      join ref as b on a.merchant=b.var1
    where date ^= .
  ;quit;
  %csvexport(out, filename=OUT_&sysdate., path=C:\Example\Reports, replace=yes);
```

Use Case 3 – API to PDF Automation:
End-user facing code

```
/************************************\
  1. Run this
\************************************/
  %include 'B:\BrandIndex\BrandIndex_API-Macros_v1.sas';
  %setup;

/************************************\
  2. Select Sector ID from BrandInd.BI_Sectors
  3. Get all Brands in Sector
\************************************/
  %getBrands(99);  /*Replace 99 with the Sector ID*/
  %let table_id=99; /*Replace 99 with the Sector ID*/

  proc sql noprint;
    select distinct
      cID
      into :ids separated by ' '
    from brandind.brands_&table_id
```

```
  ;quit;

/************************************\
  4. Get Timelines for all Brands in Sector
    Broken out by age group
    -   Start and End dates can be adjusted
        Use dates in the form 01JAN2017
\************************************/
  %GetTimeline(BrandID = %str(&ids)
                , sector = &table_id
                , gtstartdate=&sdate
                , gtenddate= &edate
                , scoring=&scoring  /*Default is Total    */
                , moving_avg=&mov_avg
                , out=&out
                , demoLoop=AGE
                , export=yes
              );
/************************************\
  5. Run New Business Brief
    (have logos ready)
\************************************/
  %let filename=BrandX;
  %let title = BrandX Brand Health;
  %let keep = 999; /* Replace 999 with Brand ID */
  %NBB;
```