Paper 929-2017

Building a member centric world from a transactional data galaxy

SAS® Global Forum 2017

Brian P. Mitchell, Humana Inc.

ABSTRACT

Health insurers have terabytes of transactional data. However, transactional data does not tell a member-level story. Humana Inc. is often faced with requirements for tagging (identifying) members with various clinical conditions such as diabetes, depression, hypertension, hyperlipidemia, and various member-level utilization metrics. For example, Consumer Health Tags are built to identify the condition (that is, diabetes, hypertension, and so on) and to estimate the intensity of the disease using medical and pharmacy administrative claims data. This case study takes you on an analytics journey from the initial problem diagnosis and analytics solution using SAS®.

INTRODUCTION

Humana Inc. has developed member level clinical condition indicator flags based on the pattern of medical and prescription claims. Humana Inc. has defined these as Consumer Health Tags (CHT). Currently, the entire Humana Inc. membership is flagged for these conditions. Development of these Consumer Health Tags, benefit our members in several ways. First and foremost research and development of CHTs assist our members' to achieve improved well-being by identifying gaps in care, prescription drug adherence issues, enrollments in clinic health improvement programs offered by Humana Inc. This research helps our members have improved clinical outcomes.

HEALTH INSURANCE COMPANIES PROCESS THE MEDICAL AND PRESCRIPTION CLAIMS DATA, DOESN'T THAT TELL YOU THE CONDITION?

The International Classification of Diseases, 9th Revision (ICD9) code 250.xx on a medical claim, which means the member has diabetes doesn't it? Not exactly, the member could have diabetes or maybe the member is undergoing testing for diabetes. Typically, a single ICD9 code is insufficient evidence to conclude a member has a condition such as diabetes.

Figure 1. Consumer Health Tag Methodology Summary

Figure 1. Demonstrates the overall flow of data and logic to create member level characteristics.

WHO, WHERE, WHEN, FOR WHAT AND … WHY?

Who: members, what cohort, coverage, plan, geography and demographics. Where: point of patient care –outpatient clinic, emergency department, urgent care, hospital inpatient and mail-order medication. When: date(s) of service and/or product rendered. What: medical claims data, prescription claims, laboratory tests and results. Why: we need to infer the members' condition.

DEFINING THE COHORT:

```
%let run_dt = 20170202; /*YYYYMMDD*/
%let member_cov =ca_prod.medr_mbr_cov;
%LET MON = JUL;
%LET YR = 16; /*Year. Example 2012 will be 12)*/

    data work.MAPD_Active_pop_&run_dt.;
      set &member_cov;
           if IND_MAPD_HMO_&MON.&YR.=1 or
               IND_MAPD_RPPO_&MON.&YR.=1 or
               IND_MAPD_PFFS_&MON.&YR.=1 or
               IND_MAPD_LPPO_&MON.&YR.=1;
    run;
```

APPENDING PRESCRIPTION CLAIMS DATA UTILIZING A HASH STATEMENT:

```
%let rx_table = tsg_tech.mth_rxclm;
   data prod.mapd_rxclms_&run_dt.;
        if _N_ =1 then do;
            declare hash h(dataset:'WORK.FINAL_TABLE');
            h.defineKey('mbr_pers_gen_key');
            h.defineDone();
        end;
      set &rx_table;
         WHERE (service_date>='  30JAN2016:00:00:00'DT);
           if h.find()=0 then output;
   run;
```

UTILIZING PROC SQL FOR PULLING DATA:

```
proc sql;
      create table work.rsp as
        select   A. *,
                 B. ndc as rs_ndc,
                 B. rs_drug

      from prod._rxclms_&run_dt. as A left join WORK.RS_DRUG_LIST_RD as B
           on A.ndc_id = B.ndc
             order by mbr_pers_gen_key;
quit;
```

UTILIZING PROC SQL FOR CREATING MEMBER LEVEL METRICS:

```
 proc sql;
          create table work.RXCLMS_12MAPDB_1 as
               select  MBR_PERS_GEN_KEY,
                     sum( case when (ind_generic_drug  = 1) then
                     PAY_DAY_SUPPLY_CNT else 0 end) as pds_generic,

                     sum(PAY_DAY_SUPPLY_CNT) as pds_total,
```

```
                    (sum(MBR_RESPONS_AMT)) as Member_Paid

          from WORK.CONTROL_MEDS
                group by MBR_PERS_GEN_KEY
                order by MBR_PERS_GEN_KEY;
  quit;
```

UTILIZING PROC SQL FOR PREFORMING A LEFT JOIN:

```
proc sql;
      create table work.total_rp_1 as
            select
                  A. *,
                  B. CNT_GENERIC_DRUGS
      from WORK.RXCLMS_12MAPDB_L as A left join WORK._NOT_OTC_RP_S2 as B
            on A.mbr_pers_gen_key = B.mbr_pers_gen_key
                  order by mbr_pers_gen_key;
quit;
```

UTILIZING DATA STEP FOR MEMBER LEVEL METRIC BUILDING:

```
data work.TOTAL_RP_2_bdg;
     set WORK.TOTAL_RP_2;
          format generic_pert 6.2;
          format generic_paid_pert 6.2;
          format brand_paid_pert 6.2;
    if pds_generic >0 then generic_pert=(pds_generic / pds_total) * 100;
                else generic_pert=0;
    if g_Member_Paid >0 and Member_Paid >0 then
    generic_paid_pert=(g_Member_Paid / Member_Paid) * 100;
                else generic_paid_pert=0;
    if b_Member_Paid >0 and Member_Paid >0 then
    brand_paid_pert=(b_Member_Paid / Member_Paid) * 100;
                else brand_paid_pert=0;
run;
```

CONCLUSION

The ability to transform transactional data to member-level is paramount to bring meaningful information to the discussion. This paper has demonstrated various SAS coding examples to help the researcher reduce the amount of data to a more useable form.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Brian P. Mitchell, MPH, MS
Humana Inc.
502-345-9221
bmitchell@humana.com