**2016\_OMI\_PCODE3 steps recording**

Initially, used OMI\_PCODE\_combined.xlsx (PCODE\_LINK) natively left join with DA\_2016.csv;

Soon realized, maybe need to update to 2016 PCODE\_LINK simply because 2006 PCODE\_LINK may change.

Data preparation for 2016 PCODE\_LINK:

./data\_preparation ==dataFormat.py==> uniq\_DA\_PCODE3.csv & uniq\_DA\_PCODE6.csv & dups\_DA\_PCODE3.csv & dups\_DA\_PCODE6.csv

Because dups\_DA\_PCODE3.csv has too many rows, split it into dups\_DA\_PCODE3-1.csv & dups\_DA\_PCODE3-2.csv in order to remove duplicates using excel functions

dups\_DA\_PCODE3-1.csv & dups\_DA\_PCODE3-2.csv => dups\_DA\_PCODE3-1-noDuplicate.csv & dups\_DA\_PCODE3-2-noDuplicate.csv => dups\_DA\_PCODE3-noDuplicate-merge.csv (13 duplicates)

dups\_DA\_PCODE3-noDuplicate-merge.csv & uniq\_DA\_PCODE3.csv => all\_DA\_PCDE3\_noDuplicate\_merge.csv (30506 duplicates)

How DAUID get generated, how is linked with PCODE

Map of density

<https://worldpostalcode.com/search>

<https://worldpostalcode.com/search>

Data comparation between (2016 PCODE\_LINK) and (2006 PCODE\_LINK):

./2006\_2016\_compare

all\_DA\_PCDE3\_noDuplicate\_merge.csv +++ FULL OUTER JOIN WITH +++ 2006\_PCODE\_LINK.csv == dataCompare.py ==> PCODE\_LINK\_compareALL.csv (64263 PCODE3 mismatch & 49226 rows of NULL, nearly 25% mismatch rate after remove NULL rows)

Mismatch reasons assumptions: data update;

Testing procedures:

<https://www12.statcan.gc.ca/census-recensement/2016/geo/geosearch-georecherche/index-eng.cfm>

<https://www12.statcan.gc.ca/census-recensement/2011/geo/map-carte/ref/index-CTDA-eng.cfm>

(Postal code search & The Dissemination Area Reference Maps will not be produced for the 2016 Census)

Seems hard to tell which one should use

~~Result achievement: average.py~~

~~Use PCODE\_LINK(2016):~~

~~2016\_left\_join.csv (empty 168: 23177 - 168) => average.py => 2016\_average.csv ==round(pop & \_q\_)==> 2016\_average\_round.csv~~

~~Use PCODE\_LINK(2006):~~

~~2006\_left\_join.csv (empty 2730: 22634 - 2730) => average.py => 2006\_average.csv ==round(pop & \_q\_)==> 2006\_average\_round.csv~~

~~Based on empty rows, it shows 2016 DA data match highly with 2016 PCODE\_LINK, so I suggest using 2016\_average.csv/2016\_average\_round.csv~~

~~Besides, except average is obvious and makes sense to use, cannot really think of other ways to find DA data to represent PCODE3; also cannot easily find relevant research (tried on uoft library and Google scholar)~~

~~Thus, my final proposal is 2016\_average.csv/2016\_average\_round.csv~~

**Weighted Average (user manual P5)**

2016\_left\_join.csv => weightedAverage.py => 2016\_weightedAverage\_round.csv

Based on user manual P5, after calculating weighted average of factor score, use factorScore\_to\_quintiles.py to get quintiles in 2016\_weightedAverage\_calculated\_quintiles.csv

Use 2016\_weightedAverage\_round.csv to merge with UHN\_Postalcode.csv and clean data (format error, invalid pcode3 like M6V), get UHN\_OMI.csv

**Standard derivation**

2016\_left\_join.csv => standev.py=> 2016\_standard\_derivation.csv

2016\_standard\_derivation.csv + 2016\_weightedAverage.csv = 2016\_weightedAverage\_and\_SD.csv

2016\_standard\_derivation.csv + 2016\_average.csv = 2016\_average\_and\_SD.csv (not use)

2016\_left\_join.csv + 2016\_weightedAverage\_and\_SD.csv => within\_n\_SD.py => 2016\_weightedAverage\_within\_1SD.csv

2016\_left\_join.csv + 2016\_weightedAverage\_and\_SD.csv => within\_n\_SD.py => 2016\_weightedAverage\_within\_2SD.csv

2016\_left\_join.csv + 2016\_average\_and\_SD.csv => within\_n\_SD.py => 2016\_average\_within\_1SD.csv (not use)

2016\_left\_join.csv + 2016\_average\_and\_SD.csv => within\_n\_SD.py => 2016\_average\_within\_2SD.csv (not use)

UHN\_OMI.csv (use pcode) + 2016\_average\_within\_1SD.csv = UHN\_2016\_weightAverage\_within\_1SD.csv

UHN\_OMI.csv (use pcode) + 2016\_average\_within\_2SD.csv = UHN\_2016\_weightAverage\_within\_2SD.csv

above\_x.py:

OMI:

percentage of all domains at least 50% within its corresponding one standard derivation is 52.56%

percentage of all domains at least 70% within its corresponding one standard derivation is 2.66%

percentage of all domains at least 90% within its corresponding two standard derivation is 73.43%

(in other words, for all OMI data, by using weight average, the percentage of 50% of the data being captured within one standard derivation is 52.56%; the percentage of 70% of the data being captured within one standard derivation is 2.66%; the percentage of 90% of the data being captured within two standard derivation is 73.43%)

patients:

percentage of all domains at least 50% within its corresponding one standard derivation is 47.88%

percentage of all domains at least 70% within its corresponding one standard derivation is 1.70%

percentage of all domains at least 90% within its corresponding two standard derivation is 72.88%

(in other words, for our patients data, by using weight average, the percentage of 50% of the data being captured within one standard derivation is 47.88%; the percentage of 70% of the data being captured within one standard derivation is 1.70%; the percentage of 90% of the data being captured within two standard derivation is 72.88%)

**Improvement: only care about quintiles**

Remove factor score columns:

2016\_weightedAverage.csv => 2016\_weightedAverage\_q.csv

2016\_weightedAverage\_within\_1SD.csv => 2016\_weightedAverage\_q\_within\_1SD.csv => q\_above\_x.py

2016\_weightedAverage\_within\_2SD.csv => 2016\_weightedAverage\_q\_within\_2SD.csv => q\_above\_x.py

UHN\_2016\_weightAverage\_within\_1SD.csv => UHN\_2016\_weightAverage\_q\_within\_1SD.csv => q\_above\_x.py

UHN\_2016\_weightAverage\_within\_2SD.csv => UHN\_2016\_weightAverage\_q\_within\_2SD.csv => q\_above\_x.py

q\_above\_x.py:

OMI:

percentage of all quintile domains at least 50% within its corresponding one standard derivation is 55.93%

percentage of all quintile domains at least 70% within its corresponding one standard derivation is 8.73%

percentage of all quintile domains at least 90% within its corresponding two standard derivation is 75.90%

(in other words, for all OMI data, by using weight average, the percentage of 50% of the data being captured within one standard derivation is 55.93%; the percentage of 70% of the data being captured within one standard derivation is 8.73%; the percentage of 90% of the data being captured within two standard derivation is 75.90%)

patients:

percentage of all quintile domains at least 50% within its corresponding one standard derivation is 54.66%

percentage of all quintile domains at least 70% within its corresponding one standard derivation is 7.20%

percentage of all quintile domains at least 90% within its corresponding two standard derivation is 75.85%

(in other words, for our patients data, by using weight average, the percentage of 50% of the data being captured within one standard derivation is 54.66%; the percentage of 70% of the data being captured within one standard derivation is 7.20%; the percentage of 90% of the data being captured within two standard derivation is 75.85%)

2016\_weightedAverage\_q.csv + 2016\_weightedAverage\_q\_within\_1SD.csv => 2016\_weightedAverage\_q\_1SD.csv => q\_above\_x.py

2016\_weightedAverage\_q.csv + 2016\_weightedAverage\_q\_within\_2SD.csv => 2016\_weightedAverage\_q\_2SD.csv => q\_above\_x.py

(hasn’t done the same for UHN and these files are not used, but these .csv can be used to set the final value for domains, now focus on percentage-count graph)

2016\_weightedAverage\_q\_within\_1SD.csv => 2016\_weightedAverage\_q\_within\_1SD\_noPcode.csv => percentage-count.py => Dependency\_1SD.png + Deprivation\_1SD.png + Ethniccon\_1SD.png + Instability\_1SD.png

2016\_weightedAverage\_q\_within\_2SD.csv => 2016\_weightedAverage\_q\_within\_2SD\_noPcode.csv (all around 0.8/0.9, no value)

UHN\_2016\_weightedAverage\_q\_within\_1SD.csv => UHN\_2016\_weightAverage\_q\_within\_1SD\_noPcode.csv => percentage-count.py => UHN\_Dependency\_1SD.png + UHN\_Deprivation\_1SD.png + UHN\_Ethniccon\_1SD.png + UHN\_Instability\_1SD.png