**LA\_tools Python Package**

| **module** | **function** | **description** | **input** | **output** |
| --- | --- | --- | --- | --- |
| **tagreport**  (pandas/numpy depends) |  | methods for pulling, validating & exporting tag reports |  |  |
|  | cap\_query | query to pull tag report from PE | PR\_rev (string) | report (dataframe) |
|  | check\_PRdates | main function to run cap/trans date checks | PR\_rev (str)  report (df) | cap and/or trans errors (dataframes) |
| look for != (H or S) instead of == E | * cap\_check | checks for missing start/stop dates, bad dates, est tags, zero tags, tag gaps | PR\_rev  report | error flags (dataframe) |
| * trans\_check |
|  | get\_report | main function to get tag report | PR\_rev (str) | report (dataframe) |
|  | export\_report | writes report to .csv in directory | PR\_rev  directory  (strings) | prints:  -file written  -overwrite prompt  -error |
| **db\_query**  (cx\_Oracle depends) |  | query DB  CHANGE BASED ON PRODUCTION  PLATFORM… | …BLOB STORAGE, DB FROM SERVER… |  |
|  | oracle\_pe | queries PE w/ credentials | query (str) | db results (list) |
|  | oracle\_lpss | TO DO? |  |  |
|  | oracle\_alps | TO DO? |  |  |
| **offer\_summary**  (pandas/numpy) |  | calculate statistics based on meter forecasts at PR level |  |  |
|  | offer\_summary | TO SPLIT INTO FUNCTIONS |  |  |
| ADD CP DATES ALL MARKETS | offer  APPLY FUNCS | calc annual use, meter peaks, CP, % diff bt tag & CP/actual | ch 3 data  (df) | offer sum  (dataframe) |
| check > act\_tag\_diff OR > cap\_tag\_diff | peaktag\_check | subset where tag, diff > min\_tag, abs(min\_diff) | ch3 data, offer sum (df)  min tag, min diff  (float) | problem meters  (df) |
|  | iter\_plot | plots multiple graphs | ch3 data  (df) | plots |
| **forecast** |  | tools to allow more direct access to EWX forecasting tools |  |  |
|  | parse\_engie | parses predeal json to csv | filename (str) | ingestion payload (csv)  ch1, sca, caps, hb |
|  | parse\_ewx | parses response json to csv | filename (str) | response payload (csv)  cap tags |
| *validation* | periodic\_zero | check if zeros are periodic or not | idr (df) margin, threshold (float) | idr w/ pz col attch’d  (df) |
|  | int\_gap\_check | check for gaps in idr | idr | idr w/ time, val diff’s & gap after index flag |
|  | variance\_val | check for spikes & dips | idr (df) n\_sd (float) window (int) | roll mean & sd, center use, roll cen mean & sd, variation, spike & dip flags |
|  | dst\_check | checks DST | idr | dst flag |
|  | fix\_interval | fixes nonhourly intervals | idr | checks time diff  (15 min, need all) |
| *estimation* | dst\_fix | adds or subs hour | idr | idr |
|  | interp | interpolates vals (based on overall error flag) | idr | idr |
| adjust to end of yr | gen\_year | generates next year dates | idr, num days (int) | idr |
|  | timeshift | shifts data by weekday, hour, mon | idr, num\_yrs (int) | timeshift (df) |
| split into val and ts funcs (take parameters for val?) | forecast\_main | uses json/csv to validate & timeshift | idr (df), read\_dir, write\_dir (str) | writes timeshift, val csv  return ts |
| **NEPOOL IDR Drop** |  | automation workaround for dropping NEPOOL IDR data |  |  |
|  | emailscrape | scrapes GSERNA box for utility portal info | past days (int) | usernames, passwords, accounts, NGRID or not  (json) |
| TIMEOUT ERROR IN CHECKING BOXES & DOWNLOADING STAGE | webscrape | uses login to download IDR for <= 5 meters at a time | email scrape json | raw meter data (csv) |
|  | raw\_split | splits raw data into individual raw IDR files | raw meter data (csv) | ind raw IDR (csv) |
|  | data\_drop | filters raw to clean ch 1 data | ind raw IDR (csv) | IDR (csv)  writes to local, possible to write to DATARESPONSE |

**TO DO:** convert code to work w/ classes,

expand database query functionality?,

expand offer summary across markets,

small improvements (listed above)

create setup.py, readme & dependencies