Apply AQoL-6D Utility Mapping Models To New Data

Matthew P Hamilton[1,*] Caroline X Gao[1,2,3]

06 April 2022

[1] Orygen, Parkville, Australia
[2] Centre for Youth Mental Health; University of Melbourne, Parkville, Australia
[3] School of Public Health and Preventive Medicine, Monash University, Clayton, Australia

[*] Correspondence: Matthew P Hamilton <matthew.hamilton@orygen.org.au>

This article provides brief instructions for using the AQoL-6D mapping models distributed in the https://doi.org/10.7910/DVN/DKDIB0 dataset, which were generated by a study described in this study.

This article provides information about:

- Searching, selecting and retrieving mapping models;
- Preparing a prediction dataset for use with a selected mapping model; and
- Applying the selected mapping model to a prediction dataset to predict Quality Adjusted Life Years (QALYs).

Before reading this article, it is recommended that you familiarise yourself with the model catalogues that are also available in the https://doi.org/10.7910/DVN/DKDIB0 dataset.

Finally, to illustrate this article we have used fake data so the analysis outlined in this article should should not be used to inform decision making.

1 Getting started

1.1 Install and load required software

To run the commands in this program, you need to have the software R installed. You will also need to install the `youthu` package using the following command.

```
devtools::install_github("ready4-dev/youthu")
```

You can now load the functions we will be using from the `youthu` package.

```
library(youthu)
```

2 Search, select and retrieve transfer to utility models

We can retrieve a lookup table of available mapping models using the `get_mdls_lup` function. The lookup table includes information on the names of models (which corresponds to the names in the model catalogues), the predictors used in each model and the analysis that generated each one.

```
mdls_lup <- get_mdls_lup(ttu_dv_dss_tb = get_ttu_dv_dss("TTU"),
                         utility_type_chr = "AQoL-6D")
```

To review the summary information about the predictive performance of a specific model, use the relevant name from the model catalogue:

```
get_dv_mdl_smrys(mdls_lup,
                 mdl_nms_chr = "PHQ9_SOFAS_1_OLS_CLL")
```

```
## $PHQ9_SOFAS_1_OLS_CLL
##        Parameter Estimate    SE          95% CI
## 1 SD (Intercept)    0.348 0.017   0.314 , 0.381
## 2      Intercept    0.421 0.131   0.167 , 0.681
## 3  PHQ9 baseline   -9.100 0.253  -9.601 , -8.61
## 4    PHQ9 change   -7.320 0.335 -7.974 , -6.655
## 5 SOFAS baseline    0.968 0.175   0.629 , 1.307
## 6   SOFAS change    1.149 0.231   0.703 , 1.603
## 7             R2    0.767 0.012   0.743 , 0.789
## 8           RMSE    0.925 0.004   0.922 , 0.928
## 9          Sigma    0.405 0.011   0.383 , 0.428
```

3 Prepare a prediction dataset for use with a selected transfer to utility model

3.1 Import data

You can now import and inspect the dataset you plan on using for prediction. In the below example we use fake data.

```
data_tb <- make_fake_ds_one()
data_tb %>% head()
```

```
## # A tibble: 6 x 5
##   UID              Timepoint Date       PHQ_total SOFAS_total
##   <chr>            <fct>     <date>         <int>       <int>
## 1 Participant_1    Baseline  2020-07-21         7          69
## 2 Participant_10   Baseline  2020-10-04        17          60
## 3 Participant_10   Follow-up 2021-01-11        17          64
## 4 Participant_100  Baseline  2020-09-20         0          76
## 5 Participant_1000 Baseline  2020-08-10         0          71
## 6 Participant_1000 Follow-up 2020-11-20         0          71
```

3.1.1 Confirm dataset can be used as a prediction dataset

The prediction dataset must contain variables that correspond to all the predictors of the model you intend to apply. The allowable range and required class of each predictor variable are described in the `min_val_dbl`, `max_val_dbl` and `class_chr` columns of the model predictors lookup table, which can be accessed with a call to the `get_predictors_lup` function.

```
predictors_lup <- get_predictors_lup(mdls_lup = mdls_lup,
                                     mdl_nm_1L_chr = "PHQ9_SOFAS_1_OLS_CLL")
predictors_lup
```

```
## # A tibble: 2 x 9
##   short_name_chr long_name_chr   min_val_dbl max_val_dbl class_chr increment_dbl
##   <chr>          <chr>                 <dbl>       <dbl> <chr>             <dbl>
## 1 PHQ9           PHQ9 total sco~           0          27 integer               1
## 2 SOFAS          SOFAS total sc~           0         100 integer               1
## # ... with 3 more variables: class_fn_chr <chr>, mdl_scaling_dbl <dbl>,
## #   covariate_lgl <lgl>
```

The prediction dataset must also include both a unique client identifier variable and a measurement time-point identifier variable (which must be a `factor` with two levels). The dataset also needs to be in long format (ie where measures at different time-points for the same individual are stacked on top of each other in separate rows). We can confirm these conditions hold by creating a dataset metadata object using the `make_predn_metadata_ls` function. In creating the metadata object, the function checks that the dataset can be used in conjunction with the model specified at the `mdl_nm_1L_chr` argument. If the prediction dataset uses different variable names for the predictors to those specified in the `predictors_lup` lookup table, a named vector detailing the correspondence between the two sets of variable names needs to be passed to the `predr_vars_nms_chr` argument. Finally, if you wish to specify a preferred variable name to use for the predicted utility values when applying the model, you can do this by passing this name to the `utl_var_nm_1L_chr` argument.

```
predn_ds_ls <- make_predn_metadata_ls(data_tb,
                                      id_var_nm_1L_chr = "UID",
                                      msrmnt_date_var_nm_1L_chr = "Date",
                                      predr_vars_nms_chr = c(PHQ9 = "PHQ_total",SOFAS = "SOFAS_total"),
                                      round_var_nm_1L_chr = "Timepoint",
                                      round_bl_val_1L_chr = "Baseline",
                                      utl_var_nm_1L_chr = "AQoL6D_HU",
                                      mdls_lup = mdls_lup,
                                      mdl_nm_1L_chr = "PHQ9_SOFAS_1_OLS_CLL")
```

4 Apply the selected transfer to utility model to a prediction dataset to predict Quality Adjusted Life Years (QALYs)

4.1 Predict health utility at baseline and follow-up timepoints

To generate utility predictions we use the `add_utl_predn` function. The function needs to be supplied with the prediction dataset (the value passed to argument `data_tb`) and the validated prediction metadata object we created in the previous step.

```
data_tb <- add_utl_predn(data_tb,
                         predn_ds_ls = predn_ds_ls)
```

```
## Joining, by = c("UID", "Timepoint")
```

```
data_tb %>%
  head()
```

```
## # A tibble: 6 x 6
##   UID              Timepoint Date       PHQ_total  SOFAS_total AQoL6D_HU
##   <chr>            <fct>     <date>     <ythvrs_9> <ythvrs_s>      <dbl>
## 1 Participant_1    Baseline  2020-07-21  7         69              0.857
## 2 Participant_10   Baseline  2020-10-04 17         60              0.323
## 3 Participant_10   Follow-up 2021-01-11 17         64              0.583
## 4 Participant_100  Baseline  2020-09-20  0         76              0.964
## 5 Participant_1000 Baseline  2020-08-10  0         71              0.995
## 6 Participant_1000 Follow-up 2020-11-20  0         71              0.862
```

By default the `add_utl_predn` function samples model parameter values based on a table of model coefficients when making predictions and constrains predictions to an allowed range. You can override these defaults by adding additional arguments `new_data_is_1L_chr = "Predicted"` (which uses mean parameter values), `force_min_max_1L_lgl = F` (removes range constraint) and (if the source dataset makes available downloadable model objects) `make_from_tbl_1L_lgl = F`. These settings will produce different predictions. It is strongly recommended that you consult the model catalogue (see above) to understand how such decisions may affect the validity of the predicted values that will be generated.

Our health utility predictions are now available for use and are summarised below.

```
summary(data_tb$AQoL6D_HU)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0700  0.4220  0.6299  0.6208  0.8320  1.0000
```

4.2 Calculate QALYs

The last step is to calculate Quality Adjusted Life Years, using a method assuming a linear rate of change between timepoints.

```
data_tb <- data_tb %>% add_qalys_to_ds(predn_ds_ls = predn_ds_ls,
                                       include_predrs_1L_lgl = F,
                                       reshape_1L_lgl = F)
data_tb[c(1:6,9)] %>%
  head()
```

```
## # A tibble: 6 x 7
##   UID             Timepoint Date       PHQ_total SOFAS_total AQoL6D_HU qalys_dbl
##   <chr>           <fct>     <date>     <ythvrs_> <ythvrs_s>      <dbl>     <dbl>
## 1 Participant_1   Baseline  2020-07-21  7        69              0.857     0
## 2 Participant_10  Baseline  2020-10-04 17        60              0.323     0
## 3 Participant_10  Follow-up 2021-01-11 17        64              0.583     0.123
## 4 Participant_100 Baseline  2020-09-20  0        76              0.964     0
## 5 Participant_10~ Baseline  2020-08-10  0        71              0.995     0
## 6 Participant_10~ Follow-up 2020-11-20  0        71              0.862     0.259
```