2018-09-13 13:49:14

The idea is to use subsets of SNPs from the PGC 2017 GWAS, which will be similar to doing PRS, except that we'll have information on individual SNPs, and won't be using the beta weights.

let's use the same dataset we used for PRS, after cleaning:

```
[sudregp@cn3443 geno3]$ pwd
/home/sudregp/data/prs/geno3
plink --bfile merged_noDups_clean_flipped --out
merged_noDups_clean_flipped.tab --recodeA

awk '$9 < 1e-04' ~/pgc2017/adhd_jun2017 | cut -f 2 - >
~/tmp/snps_pgcadhd2017_1e04.txt
```

That generates several subsets of SNPs. Then, in R:

```
library(h2o)
h2o.init()
df =
h2o.importFile('/Users/sudregp/data/baseline_prediction/merged_noDups_clea
n_flipped.tab.raw')
snps = read.table('~/data/baseline_prediction/snps_pgcadhd2017_1e04.txt',
colClasses='character')[,1]
cnames = sapply(colnames(df), function(x) { gsub('_[GACT]$','',x) })
colnames(df) = cnames
keep_me = c('IID'); for (i in snps) { if (i %in% cnames) { keep_me =
c(keep_me, i) } }
df2 = df[, keep_me]
write.csv(as.data.frame(df2),file='~/data/baseline_prediction/geno3_snps1e
04_09132018.csv', row.names=F)
```

And then repeat that for the other thresholds.

```
for (p in 5:9) {
    print(p)
    snp_fname =
sprintf('~/data/baseline_prediction/snps_pgcadhd2017_1e0%d.txt', p)
    snps = read.table(snp_fname, colClasses='character')[,1]
    keep_me = c('IID'); for (i in snps) { if (i %in% cnames) { keep_me =
c(keep_me, i) } }
    df2 = df[, keep_me]
    out_fname =
sprintf('~/data/baseline_prediction/geno3_snps1e0%d_09132018.csv', p)
    write.csv(as.data.frame(df2),file=out_fname, row.names=F)
}
```

I saved everything as CSV to make it easier to access in the future. Now, the only processing that makes sense ot use here is raw and pca, as the univariate was replaced by the GWAS results.

2018-09-14 10:44:20

I converted the .csv files to be able to use with all the other functions (well, just raw.R, as there's no point in doing univariate analysis here). Here I'm also calling NSBs to be mask.id just to conform with the automl functions.

```
for (p in 4:9) {
    print(p)
    fname =
sprintf('~/data/baseline_prediction/geno3_snps1e0%s_09132018.csv', p)
    data = read.csv(fname)
    nsb_long = as.vector(data$IID)
    need_reformat = grepl('-', nsb_long)
    clean_nsb = sapply(nsb_long[need_reformat],
                       function(x) strsplit(strsplit(x, '-')[[1]][3], '@')
[[1]][1])
    nsb = nsb_long
    nsb[need_reformat] = clean_nsb
    data$mask.id = nsb
    var_names = grepl('^rs', colnames(data))
    cnames = sapply(colnames(data)[var_names], function(x) sprintf('v_%s',
x))
    colnames(data)[var_names] = cnames
    fname =
sprintf('~/data/baseline_prediction/geno3_snps1e0%s_09132018.RData.gz', p)
    save(data, file=fname, compress=T)
}

data =
read.csv('/Volumes/Shaw/prs/FINAL_FILES_08242018/REGRESSION/PRS2017_geno3_
1KG_noFlip_genop05MAFbtp01rsbtp9.csv')
data$NSB = as.numeric(data$NSB)
data$MRN = NULL
colnames(data)[1] = 'mask.id'
var_names = grepl('^PROF', colnames(data))
cnames = sapply(colnames(data)[var_names], function(x) sprintf('v_%s', x))
colnames(data)[var_names] = cnames
save(data, file='~/data/baseline_prediction/geno3_prs_09142018.RData.gz',
compress=T)
```

Then, we just need to create the gf file. I used the new ranef files to create ~/data/baseline_prediction/long_clin_0913.csv. So, let's use that:

```
prs =
read.csv('/Volumes/Shaw/prs/FINAL_FILES_08242018/REGRESSION/PRS2017_geno3_
1KG_noFlip_genop05MAFbtp01rsbtp9.csv')
clin = read.csv('~/data/baseline_prediction/long_clin_0913.csv')
m = merge(clin, prs, by.x='X...MRN', by.y='MRN')
m$X...MRN = NULL
m[, 15:27] = NULL
colnames(m)[13] = 'mask.id'
m$PROFILES.0.00001.profile = NULL
write.csv(m, file='~/data/baseline_prediction/geno3_gf_09142018.csv',
row.names=F)

clin = read.csv('~/data/baseline_prediction/long_clin_0913.csv')
clin$mask.id = clin$MRN
write.csv(clin, '~/data/baseline_prediction/cog_gf_09142018.csv')
```