Noah L. Schrick
1492657

Lab 7. Statistical methods for finding associations between Single Nucleotide Polymorphisms (SNPs) and phenotypes (case/control) in genome-wide association study (gwas) data. We will use the chi-square test and logistic regression to test for association.

0. Reading PLINK files. Use the code below with the snpStats library to read the toy PLINK data in the files extra.ped and extra.map. **1.** How many subjects and predictors are there (use dim)? **2.** How many phenotype-0 samples are heterozygous for the SNP below? How many phenotype-1 samples are heterozygous (see the output of the table)?

```
library(snpStats) #BiocManager::install("snpStats")

ex.data <- read.pedfile(file="extra.ped", snps="extra.map")
ex.data$fam
phenotype <- ex.data$fam$affected-1  # change pheno from 1/2 to 0/1
genotypes <- ex.data$genotypes       # encoded as AA/AB/BB
snp.ids <- as.character(ex.data$map$snp.names)
genotypes.df <- data.frame(as(genotypes, "character"))
colnames(genotypes.df) <- snp.ids
# observed contingency table for SNP rs630969
table(phenotype,genotypes.df$rs630969,
                dnn=c("phenotype","genotype")) # dnn dimension names of table
```

1. 89 subjects, 17 predictors
2. Phenotype-0 samples that are heterozygous: 17
Phenotype-1 samples that are heterozygous: 21

A. Chi-Square Test. The chi-square test compares the observed and expected SNP x Phenotype contingency tables. The expected table shows how the subjects would be distributed in the SNP x Phenotype cells if there were no enrichment of genotypes in one of the phenotype groups. A difference suggests the SNP may increase susceptibility to the disease. We will use `fisher.test`, an implementation of Fisher's exact test, used when table cells are sparse.

Use the following code to create a list of contingency tables for all SNPs with sapply. **3.** Fill in the first blank and show the observed contingency table for SNP rs634228 (hint: type `observed.tables.list` at the interpreter). Then using `test.table` fill in the remaining blanks of code to calculate the values for the

genotype margins (`genoMarg.vec`), the phenotype margins (`phenoMarg.vec`), and the total number of subjects (`totalSubj`). Hint: use functions **rowSums**, **colSums** and **sum**. Fill the results into the **Observed** Excel Table below.

```
# creates list of observed contingency tables for all SNPs
# sapply acts on each column of genotypes.df
observed.tables.list <- sapply(genotypes.df, function(x)
table(phenotype,x,dnn=c("phenotype","genotype")))

test.table <- observed.tables.list$rs634228          # grab one table
genoMarg.vec  <-  colSums(test.table)                # margin vector
phenoMarg.vec <- rowSums(test.table)                 # margin vector
totalSubj <- sum(genoMarg.vec)                       # total subjects


> observed.tables.list$rs634228
         genotype
phenotype A/A A/B B/B
        0  29  12   0
        1  37  10   1
```

Observed	A/A	A/B	B/B	
Case:	29	12	0	41
Control:	37	10	1	48
	66	22	1	89

4. Once you create the observed matrix, use the formulas to fill in the **Expected** table below. Fill in table as fractions. Then check your answer with this code:

```
expect.test <- outer(phenoMarg.vec,genoMarg.vec/totalSubj,'*')
```

Expected	A/A	A/B	B/B	
Case:	66*(41/89)	22*(41/89)	1*(41/89)	41
Control:	66*(48/89)	22*(48/89)	1*(48/89)	48
	66	22	1	89

Given the observed and expected tables, the chi-square is calculated by the following equation, but some of the cells are sparse or empty, which leads to numerical problems.

$$\chi^2 = \sum_{c=Cells} \frac{(O_c - E_c)^2}{E_c}$$

The Fisher exact test uses a different approach that is more appropriate for sparse cells. The variable `observed.tables.list`, computed above, is a list of contingency tables for each SNP. Use the code below to `sapply` the `fisher.test` to each table. **5.** Show the table of the results (fish.results).

```
# Fisher exact test (chi-square test) for all SNPs
fish_fn <- function(i){
      cbind(snp.ids[i], fisher.test(observed.tables.list[[i]])$p.value)
}

# apply fisher exact test to all SNPs
fish.df <- data.frame(t(sapply(1:ncol(genotypes.df), fish_fn)))
colnames(fish.df) <- c("rs", "p_value")

# sort SNPs by Fisher exact p-value
library(dplyr)
fish.results <- fish.df                              %>%
                mutate_at("p_value", as.character) %>%
                mutate_at("p_value", as.numeric)   %>%
                arrange(p_value)
print(fish.results)
```

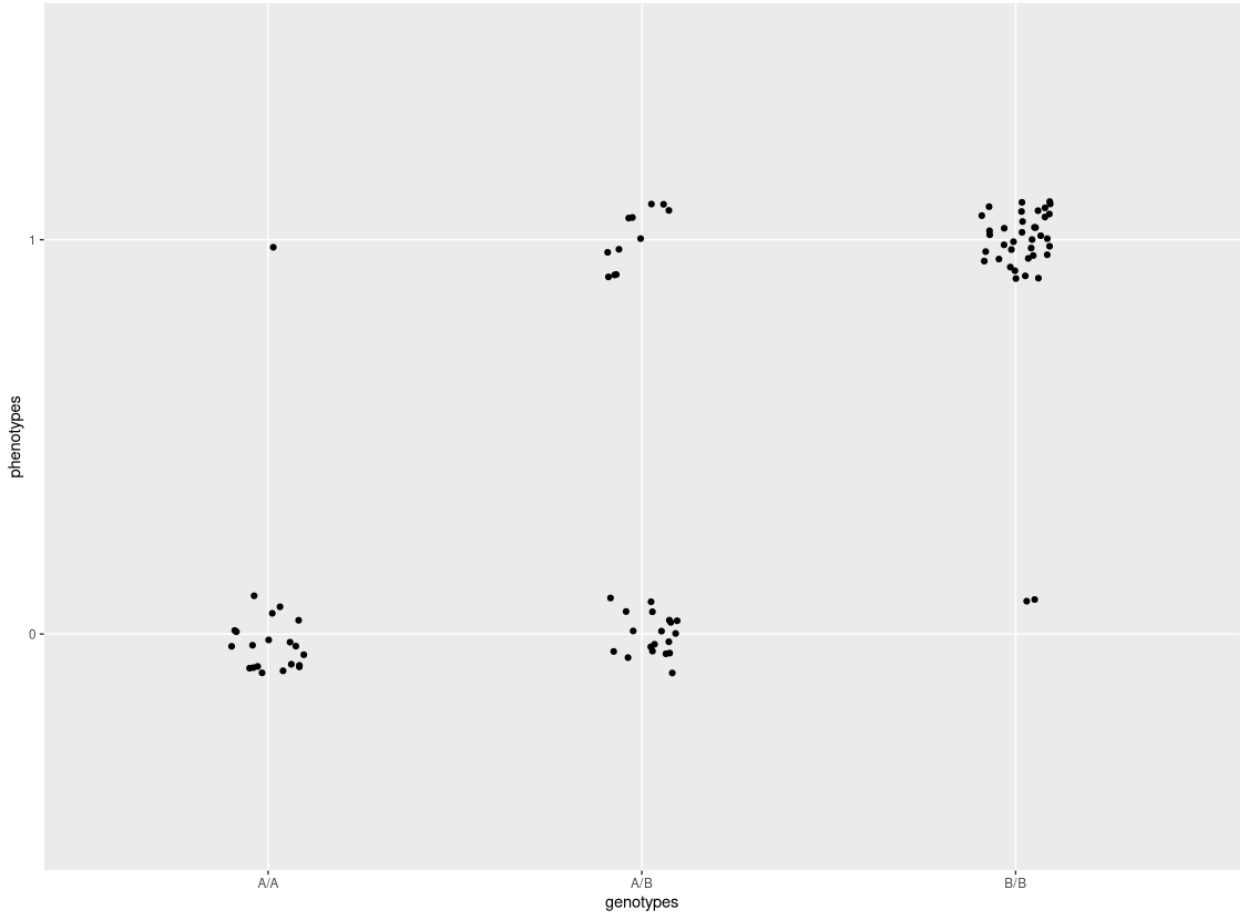
```
> print(fish.results)
       rs          p_value
1   rs7835221    1.578661e-13
2   rs2460911    8.032123e-04
3   rs2460915    3.925298e-03
4   rs6999231    2.096815e-01
5  rs17786052    2.375773e-01
6    rs529983    2.377522e-01
7  rs12156420    2.919794e-01
8  rs17121574    4.033435e-01
9  rs10105623    4.193435e-01
10   rs634228    4.609209e-01
11  rs2460914    7.209905e-01
12   rs607499    7.237858e-01
13 rs17178729    7.271907e-01
14   rs556531    7.476514e-01
15   rs754238    7.830830e-01
16   rs630969    7.961562e-01
17 rs11203962    8.332052e-01
```

B. Logistic regression with genotypes. In previous labs, we applied logistic regression to a categorical (e.g., case/control) outcome with a numeric (e.g.,

expression) predictor. Logistic regression also works for categorical (e.g., genotype) predictors. **6.** Show the plot of the data and logistic model (code below). What genotype has the highest susceptibility for the simulated disease?

```
library(ggplot2) # try installing ggplot2 with BiocManager if plot fails
i<-8
A1<-ex.data$map$allele.1[i]
A2<-ex.data$map$allele.2[i]
geno.labels <- c(paste(A1,A1,sep=""),paste(A1,A2,sep=""),paste(A2,A2,sep=""))

# data from the one SNP
oneSNP.df <- data.frame(cbind(genotypes.df[[i]],as.numeric(phenotype)))
colnames(oneSNP.df) <- c("genotypes","phenotypes")

lr.plot <- ggplot(oneSNP.df, aes(x=genotypes, y=phenotypes)) +
  geom_point(position = position_jitter(w = 0.1, h = 0.1)) +
  # stat_smooth plots the probability based on the model
  stat_smooth(method="glm", method.args = list(family = "binomial")) +
  xlim(geno.labels) + ggtitle(snp.ids[i])
print(lr.plot)
```

6.



B/B has the highest susceptibility.

7. Fill in the blanks in the code below to fit a logistic regression model of the phenotype with the SNP in the *i=8* column. (hint: look at the output of the variable `tdlr`). A SNP has three genotype states (AA, AB, BB) and hence has three regression beta coefficients. **8.** What is the rs number of this SNP, what are the genotypes (`geno.labels`), and what are the regression coefficients for each genotype?

```
library(broom) # for tidy function. make sure installed
pheno.factor <- factor(phenotype,labels=c(0,1))
i<-8
lr <- glm(pheno.factor~genotypes.df[[i]],family=binomial)
td.lr <- tidy(lr)

pval_vec <- td.lr$p.value # vector of $p.value from td.lr

coef_vec <- td.lr$estimate # vector of $estimate

cbind(snp.ids[i], coef_vec[1], coef_vec[2], coef_vec[3], pval_vec[1],
pval_vec[2], pval_vec[3])
```

8. RS number: rs7835221
Genotypes: GG, GA, AA
Regression Coefficients:
"-2.9957322607839" "2.44918855441583" "5.88610401617461"

9. Using the code above, fill in the blanks below in the `LR.fn` function, which will be used in `sapply` to create a table of logistic regression results for each SNP in the data set. Show the results sorted by the BB homozygous beta p-value (code given). **10**. How similar are the SNP rankings between logistic regression and the chi-square Fisher test?

```
LR.fn <- function(i){

      lr <- glm(pheno.factor~genotypes.df[[i]],family=binomial)
      td.lr <- tidy(lr)


      pval_vec <- td.lr$p.value # vector of $p.value from td.lr
      coef_vec <- td.lr$estimate # vector of $estimate
      cbind(snp.ids[i], coef_vec[1], coef_vec[2], coef_vec[3], pval_vec[1],
            pval_vec[2], pval_vec[3])
}

# apply Logistic Regression model to all SNPs
LRresults.df <- data.frame(t(sapply(1:ncol(genotypes.df), LR.fn)))

# add column names to results data frame
colnames(LRresults.df) <- c("rs", "AAintercept", "ABcoef", "BBcoef",
```

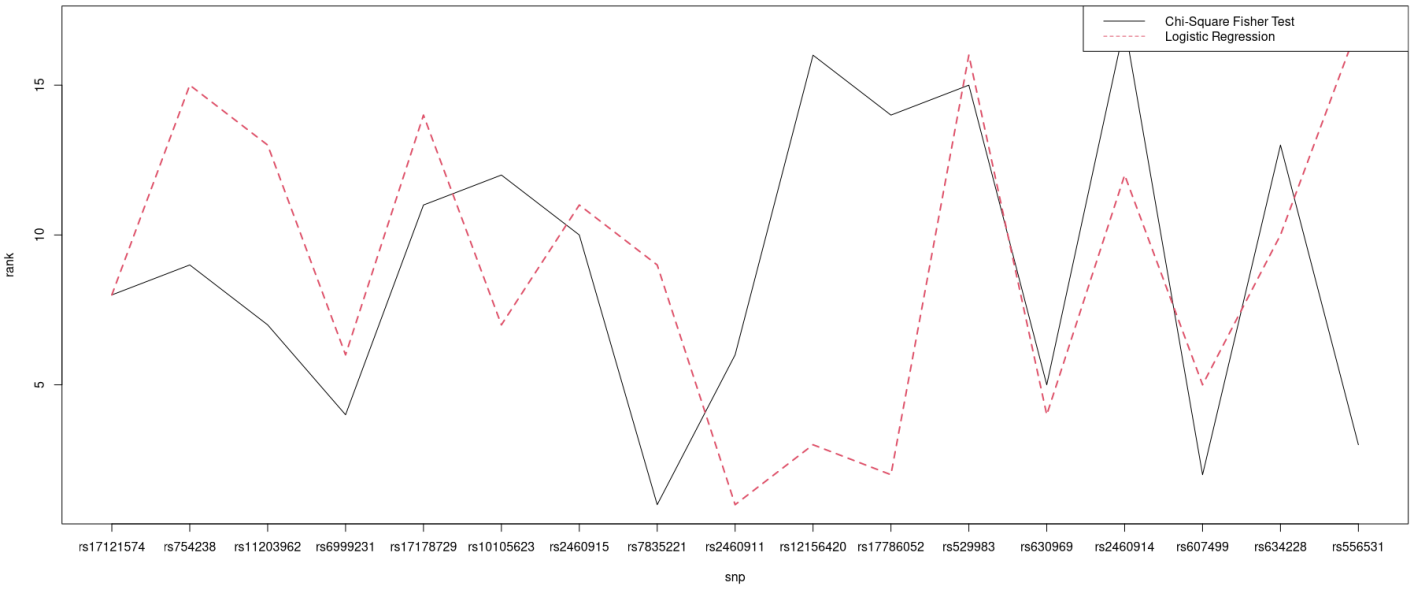
```
"AA.pval", "AB.pval", "BB.pval")

# The following sorts LR results by the p-value of the BB
# homozygous coefficient. tidy made $p_value a factor and when you try to
# convert directly to numeric (as.numeric) turns factors into integer and
# this messes up sorting especially with scientific notation

lr.results.sorted <- LRresults.df    %>%
  mutate_at("BB.pval", as.character) %>%  # convert to char before numeric
  mutate_at("BB.pval", as.numeric)   %>%  # convert to numeric for arrange
  arrange(BB.pval)                        # sort

as.matrix(lr.results.sorted %>% pull(rs,BB.pval))
```

```
> as.matrix(lr.results.sorted %>% pull(rs,BB.pval))
                      [,1]
2.78162616732067e-06 "rs7835221"
0.366736032162567    "rs607499"
0.524686965956501    "rs630969"
0.736455224784033    "rs10105623"
0.94427390326845     "rs2460914"
0.98850203886007     "rs2460915"
0.990318120879495    "rs17786052"
0.990949929337306    "rs2460911"
0.991349938432284    "rs17121574"
0.991351495823794    "rs11203962"
0.991358016447638    "rs754238"
0.99160001515836     "rs634228"
0.99206309049722     "rs6999231"
0.994401158056053    "rs529983"
<NA>                 "rs17178729"
<NA>                 "rs12156420"
<NA>                 "rs556531"
```

10.

	fish.results.rs	lr.results.sorted.rs
1	rs7835221	rs7835221
2	rs2460911	rs607499
3	rs2460915	rs630969
4	rs6999231	rs10105623
5	rs17786052	rs2460914
6	rs529983	rs2460915
7	rs12156420	rs17786052
8	rs17121574	rs2460911
9	rs10105623	rs17121574
10	rs634228	rs11203962
11	rs2460914	rs754238

12	rs607499	rs634228
13	rs17178729	rs6999231
14	rs556531	rs529983
15	rs754238	rs17178729
16	rs630969	rs12156420
17	rs11203962	rs556531



Comparisons are displayed in table form and figure form. Both demonstrate that the highest ranked SNP matches in both the chi-square fisher test and in the logistic regression. While the rankings follow similar trends in some areas (rs24600914 trends to the lower ranking in both, rs607499 trends higher in both), they are also inverted in others (rs2156420 is ranked very highly in logistic regression, but very low in the chi-square fisher test, and the inverse is true for rs556531).

Optional 1. Install and run `qqman` to create a Manhattan plot from their example GWAS results stored in `gwasResults`.

```
#install.packages("qqman")
library(qqman)
manhattan(gwasResults, chr="CHR", bp="BP", snp="SNP", p="P" )
```

```
manhattan(gwasResults, annotatePval = 0.01)
```

Use the SNP information from `ex.data$map` and the P-values from `fish.df` to create a Manhattan plot for our toy data. It will look a unManhattan-like because it just has a few SNPs.

Optional 2. Modify the data `genotypes.df` and `pheno.factor` to create input that works with glmnet to do LASSO penalized regression. Show what SNP(s) are selected.