Compulsory exercise: Team Supergreat

MA8701 Advanced Statistical Learning V2023

Nora Aasen, Elias Angelsen, Jonas Nordstr?m

10 mars, 2023

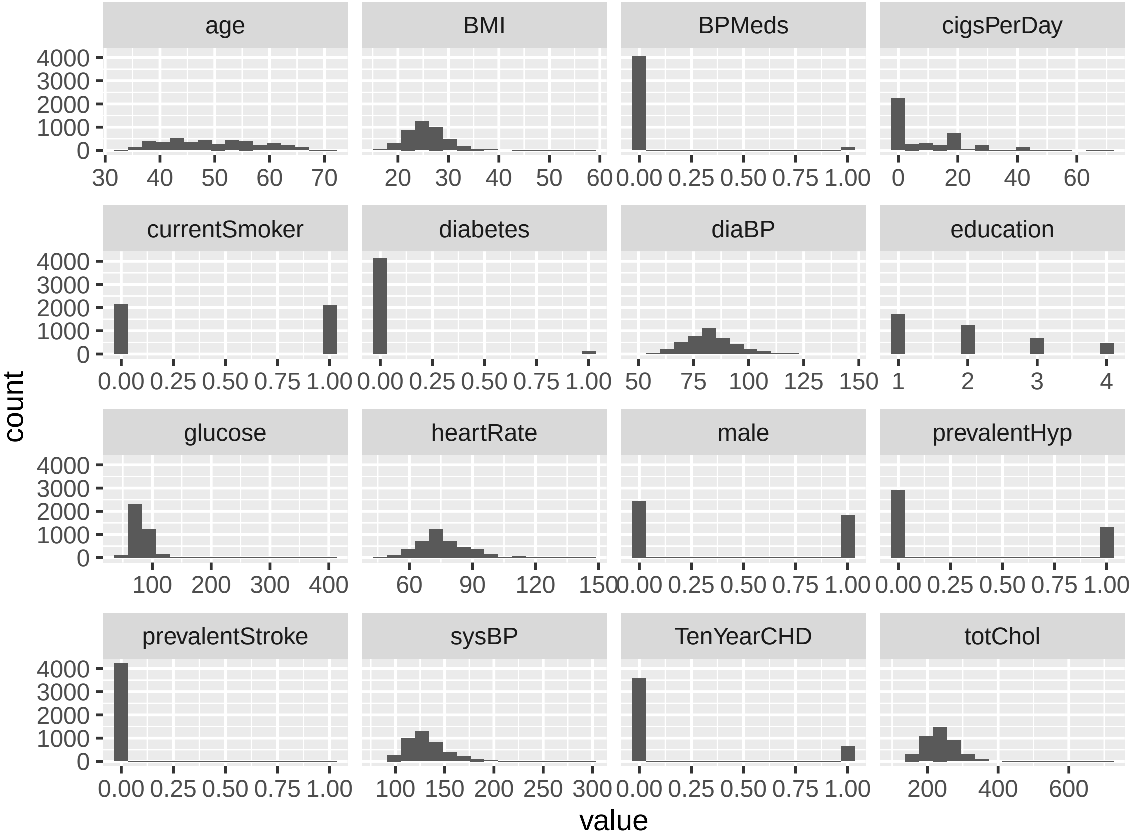
Introduction

In this project we have studied the Framingham Coronary Heart Disease Dataset. This dataset contains patient information for inhabitants in Framingham, Massachusetts, and is typically used to predict the chance of getting coronary heart disease (CHD) within the next 10 years. For this project, however, we intend to use lasso to find the most important risk factors.

We start by examining the dataset.

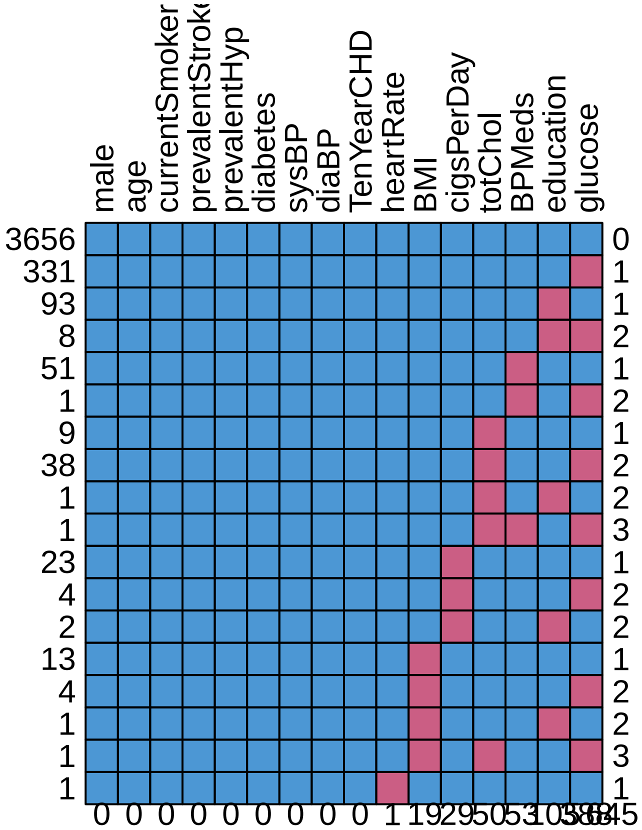
Exploratory Analysis

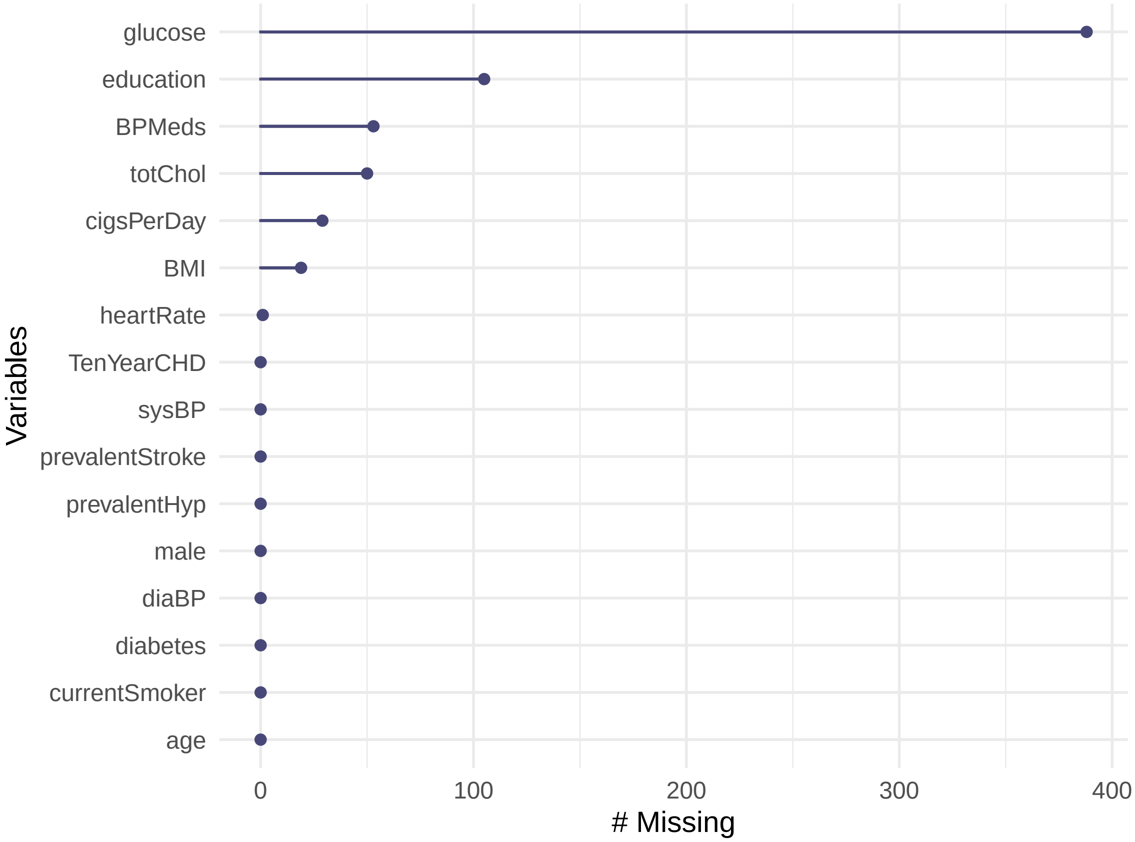
```
## 'data.frame':    4238 obs. of  16 variables:
##  $ male           : int  1 0 1 0 0 0 0 0 1 1 ...
##  $ age            : int  39 46 48 61 46 43 63 45 52 43 ...
##  $ education      : int  4 2 1 3 3 2 1 2 1 1 ...
##  $ currentSmoker  : int  0 0 1 1 1 0 0 1 0 1 ...
##  $ cigsPerDay     : int  0 0 20 30 23 0 0 20 0 30 ...
##  $ BPMeds         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ prevalentStroke: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ prevalentHyp   : int  0 0 0 1 0 1 0 0 1 1 ...
##  $ diabetes       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ totChol        : int  195 250 245 225 285 228 205 313 260 225 ...
##  $ sysBP          : num  106 121 128 150 130 ...
##  $ diaBP          : num  70 81 80 95 84 110 71 71 89 107 ...
##  $ BMI            : num  27 28.7 25.3 28.6 23.1 ...
##  $ heartRate      : int  80 95 75 65 85 77 60 79 76 93 ...
##  $ glucose        : int  77 76 70 103 85 99 85 78 79 88 ...
##  $ TenYearCHD     : int  0 0 0 1 0 0 1 0 0 0 ...
```



This data set contains 4238 observations, 15 covariates and a binary response variable `TenYearCHD`, thus we will try to fit a logistic regression model. The response variable has 644 observations that are 1, which equals about 15.2% of the total observations. Most of our covariates are either binary, or numeric. However, we notice that the variable education is most likely a categorical covariate. We could not find any further elaboration for which four categories the numbers represent, so based on the frequency of each value and naive guessing we changed it to a factor variable and defined the four categories as none, hs, college, post-grad.

The next thing we looked at was the number of missing data in our data set.


male
age
currentSmoker
prevalentStroke
prevalentHyp
diabetes
sysBP
diaBP
TenYearCHD
heartRate
BMI
cigsPerDay
totChol
BPMeds
education
glucose
3656
331
93
8
51
1
9
38
1
1
23
4
2
13
4
1
1
1
0
1
1
2
1
2
1
2
2
3
1
2
2
1
2
2
3
1
0
0
0
0
0
0
0
0
0
1
19
29
50
53
105
388
645



As we can see there are six covariates that has missing data: `glucose`, `education`, `BPMeds`, `totChol`, `cigsPerDay`, and `BMI`. We cannot use the rows that contain missing values as is. The easiest solution is to remove all rows that contains `NA`'s. This is the *complete case* solution.

The complete data set contains 3656 observations and the response variable has 644 observations that are 1, which equals about 15.2% of the total observations. As we can see, the proportion of positive observations in the response is the same, which is a good indicator that our data is missing at random (MAR), and it is therefore possible to do imputation.

From the exploratory analysis we see that there are many missing data points. However, the complete case data set is also quite large. Our main focus for this project will therefore be to compare the results from doing lasso on the complete case with the results from doing lasso on an imputed data set.

Missing Data

In our data set, we have some missing values, and we are going to handle these missing values in a slightly more refined manner than just considering the complete case (although our approach will be focused on being exploratory/explicit for the sake of learning instead of "optimal").

First, as a theoretical warm-up, recall that there are several types of mechanisms for missing data. Let $Z = (X, y)$ denote the full collection of covariates and responses, respectively, and we let a subscript mis/obs indicate restricting Z or X to the missing or observed parts, respectively. We may form an indicator (0-1) matrix R indicating missing (0) covariates and observed (1) covariates. Assume ψ is short notation for the parameters in the distribution of R.

The missing data may be characterized by the conditioning in the distribution of R. We define the data to be:

- missing completely at random (MCAR) if $P(R|Z, \psi) = P(R|\psi)$,
- missing at random (MAR) if $P(R|Z, \psi) = P(R|Z_{obs}, \psi)$,
- missing not at random (MNAR) if $P(R|Z, \psi) = P(R|Z, \psi)$ (we don't have MCAR or MAR).

By exploring for example the missing pattern of the variable cigsPerDay, we obtain a clear indication that our missing mechanism is not MCAR. No non-smoker has failed to answer the question "how may cigarettes do you smoke a day?", which is a question only aimed at smokers. The simple explanation may be that the survey they answered automatically fills in 0 for cigsPerDay if you claim to be a non-smoker. In more mathematical terms, cigsPerDay depends on the observed answer to "do you smoke?" (found in variable currentSmoker), indicating that we do not work with MCAR data. Luckily, most methods are applicable if our missingness is at least MAR.

We will assume that the missing mechanism is MAR, as there is no clear reason to suspect it to be MNAR. A pressing obstruction to being MAR can again be found in cigsPerDay. In the real world, if smokers have failed to report cigsPerDay, it may for example be because they smoke so much that they are ashamed to answer the question (and skips it), but we will simply assume that such a thing is not happening, as we trust people to answer truthfully (often, at least) if they volunteer for medical studies.

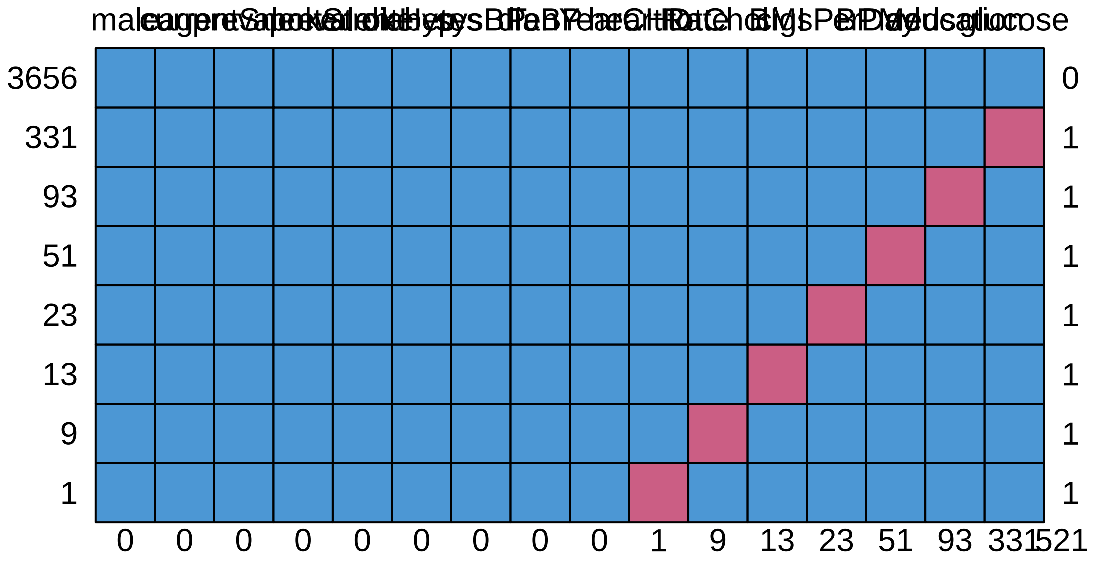
To treat the missing data, we will use single imputation, as multiple imputation may cause difficulties with the resulting inference, as Rubins rules need to be combined with the Lasso, bootstrap and concluding inference.

The single imputation technique we will use is simply regression imputation, where we adapt our regression technique depending on the type of variable imputed. For continuous variables, we simply use a linear regression model. For binary variables, we use logistic regression to classify their values, and for the variable "education", which is a four-class variable, we have utilized kNN for multiclass imputation.

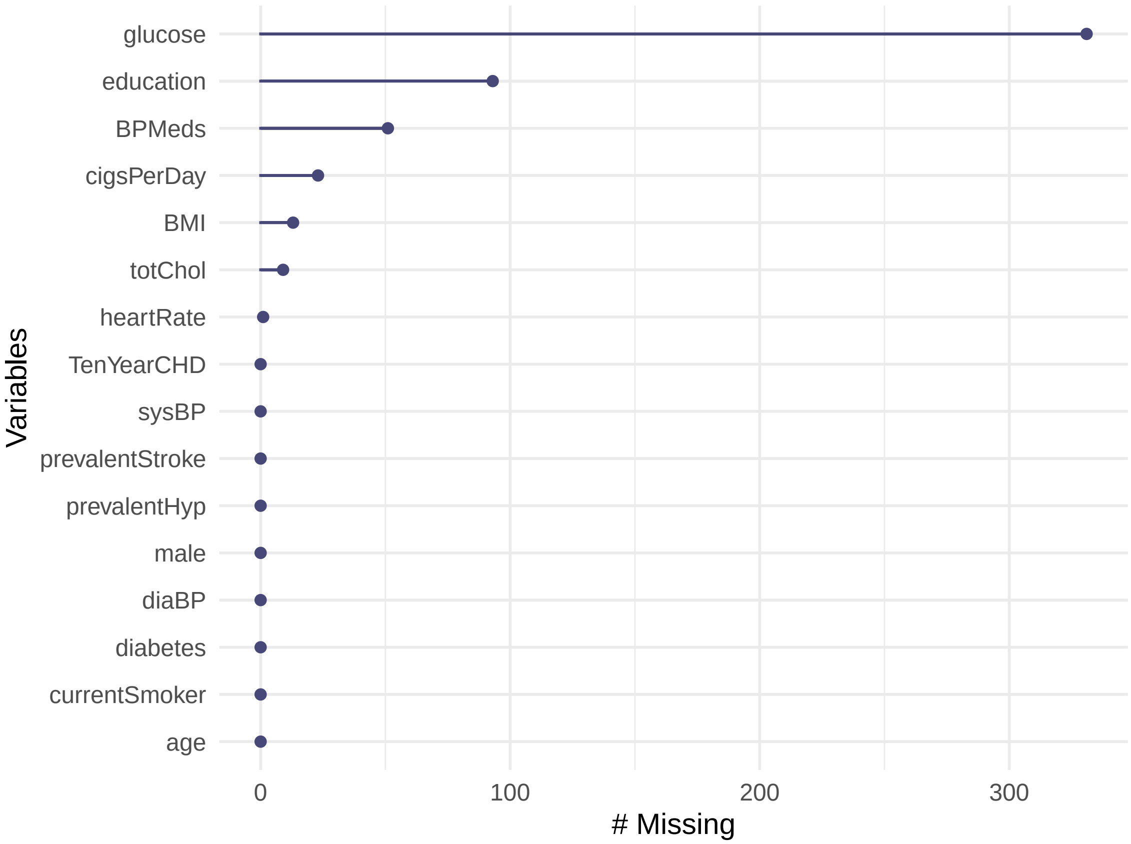
To avoid encountering observations with more than one missing value, and hence problems with regressing, we remove all samples with more than one NA.

Our data is split into training and test sets, of sizes approximately 80 and 20 of the original dataset, respectively. In order to avoid data leakage in our imputation if the test set, we fit the imputation models on the training set. Note that we do not include the response (TenYearCHD) in the regression, and in order to avoid too much correlation between the imputed samples, we always base the regression models on the complete case data, instead of letting the imputed values for variable n regress to impute variable $n + 1$.

First, we are removing all samples with more than one covariate missing, as this is only 61 samples.



```
##      male age currentSmoker prevalentStroke prevalentHyp diabetes sysBP diaBP
## 3656    1   1             1               1            1        1     1     1
## 331     1   1             1               1            1        1     1     1
## 93      1   1             1               1            1        1     1     1
## 51      1   1             1               1            1        1     1     1
## 23      1   1             1               1            1        1     1     1
## 13      1   1             1               1            1        1     1     1
## 9       1   1             1               1            1        1     1     1
## 1       1   1             1               1            1        1     1     1
##         0   0             0               0            0        0     0     0
##      TenYearCHD heartRate totChol BMI cigsPerDay BPMeds education glucose
## 3656          1         1       1   1          1      1         1       1   0
## 331           1         1       1   1          1      1         1       0   1
## 93            1         1       1   1          1      1         0       1   1
## 51            1         1       1   1          1      0         1       1   1
## 23            1         1       1   1          0      1         1       1   1
## 13            1         1       1   0          1      1         1       1   1
## 9             1         1       0   1          1      1         1       1   1
## 1             1         0       1   1          1      1         1       1   1
##               0         1       9  13         23     51        93     331 521
```



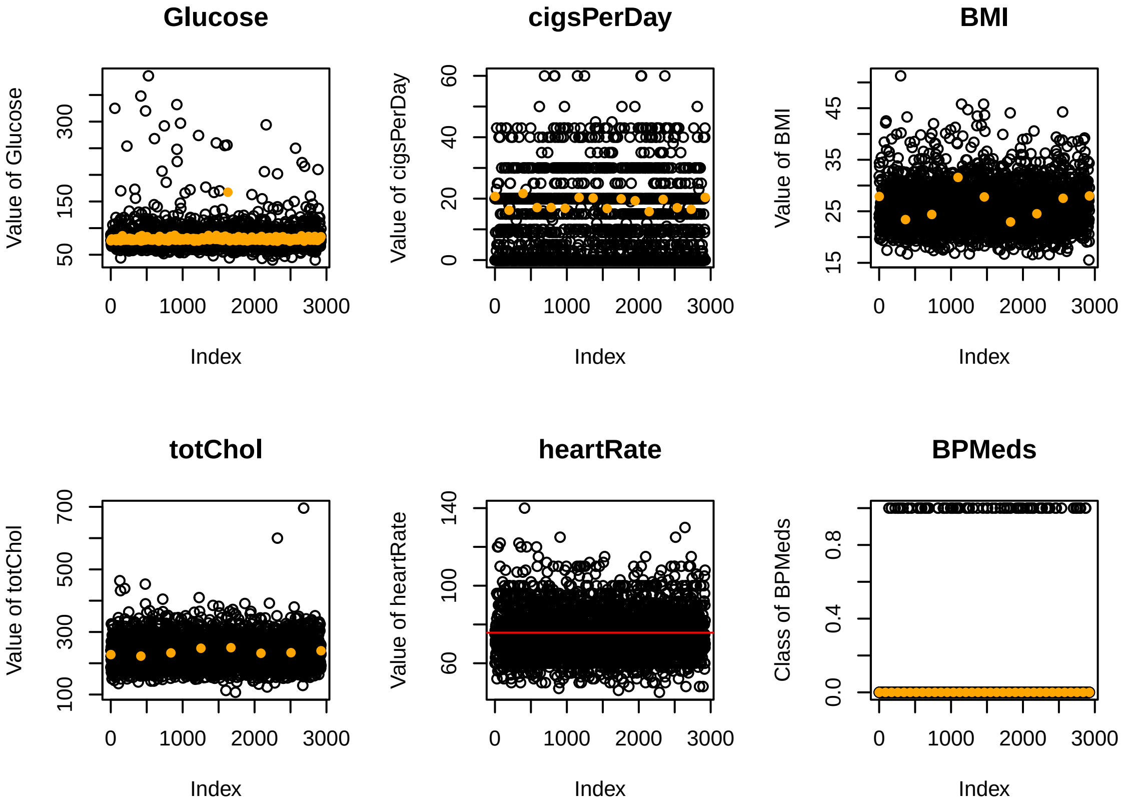
We further split the data into a training and test set, following the training-test-ratio that we have previously set.

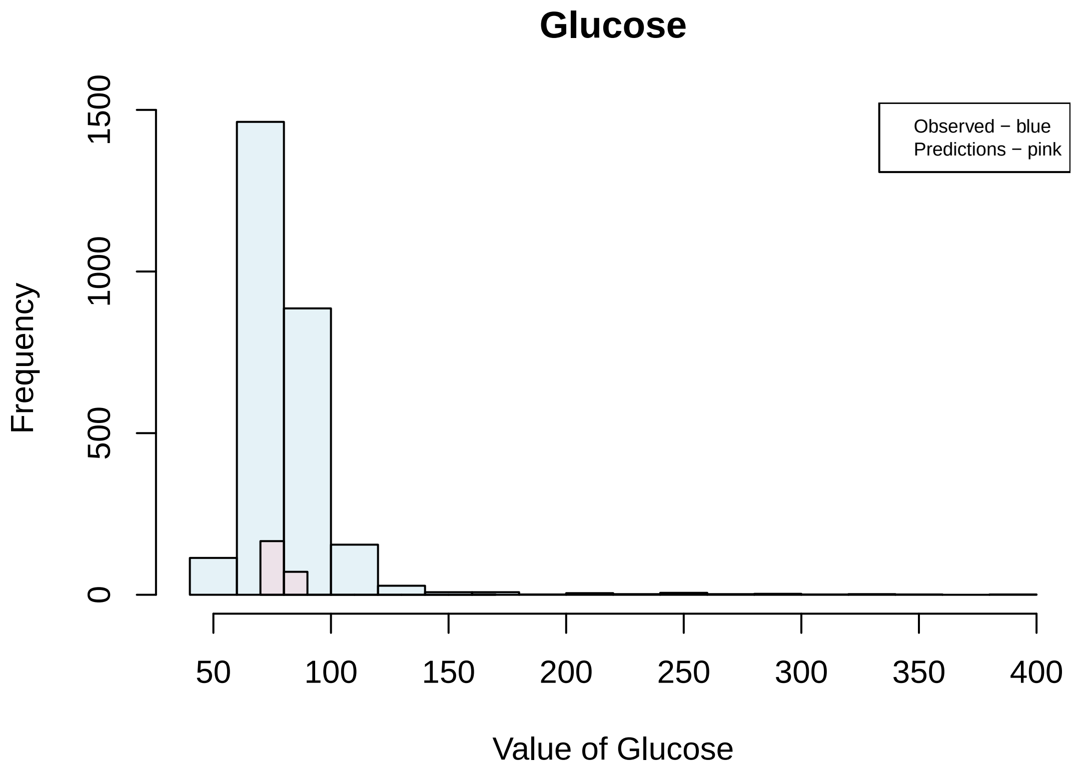
Using the above data sets, we make regression models for each missing variable based on the data we have. These automatically neglects NA's.

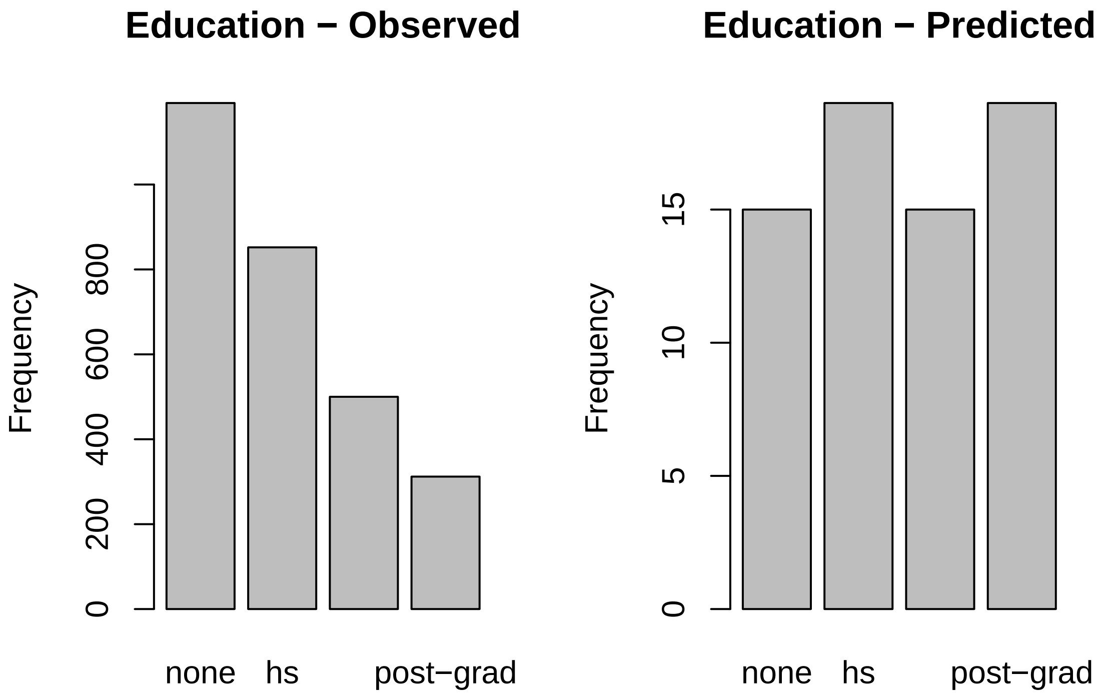
To predict, we pick out those samples with missing values of each variable from the training and test set. Note that we are imputing for both the training and test set, even though the sampled (therefore "random") split into training and test set may have sorted all NA's of a variable (e.g. heartRate) into a single set (i.e. into either training or test set). This is not a problem, as we are only picking out the incomplete cases. Therefore, predicting and filling in the missing values is a vacuous operation, not yielding any problems.

We predict new samples for the imputation.

To see how our predictions (on the training data) compare to the complete case, we plot (in different ways) the predicted values/classes. For illustrating different types of plots, we plot glucose as a transparent histogram and as a point plot.


Glucose
cigsPerDay
BMI
Value of Glucose
50
150
300
0
1000
2000
3000
Index
Value of cigsPerDay
0
20
40
60
0
1000
2000
3000
Index
Value of BMI
15
25
35
45
0
1000
2000
3000
Index
totChol
heartRate
BPMeds
Value of totChol
100
300
500
700
0
1000
2000
3000
Index
Value of heartRate
60
100
140
0
1000
2000
3000
Index
Class of BPMeds
0.0
0.4
0.8
0
1000
2000
3000
Index

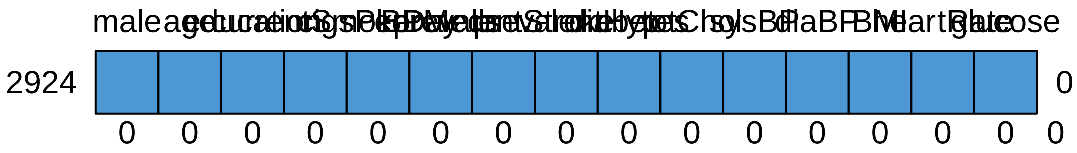

Glucose
Observed − blue
Predictions − pink
Frequency
0
500
1000
1500
50
100
150
200
250
300
350
400
Value of Glucose

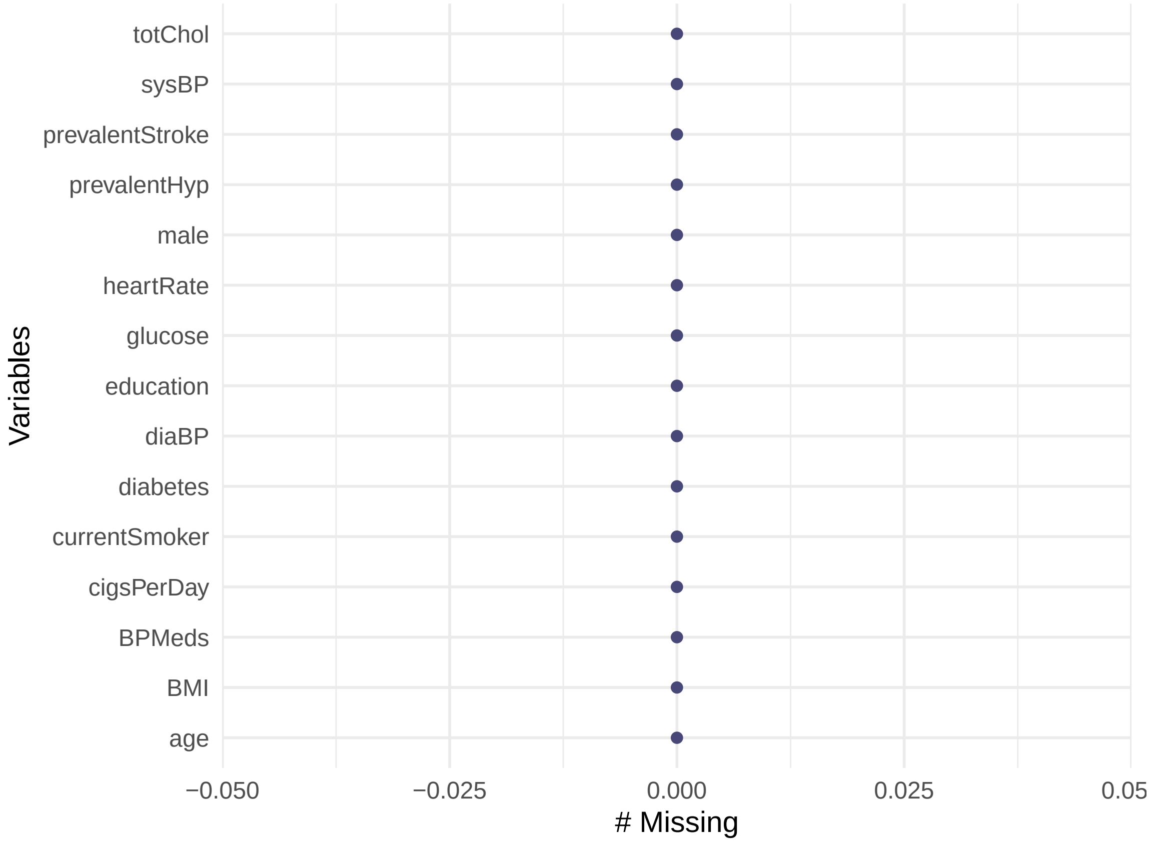


Lastly, we update our data with the newly imputed values.

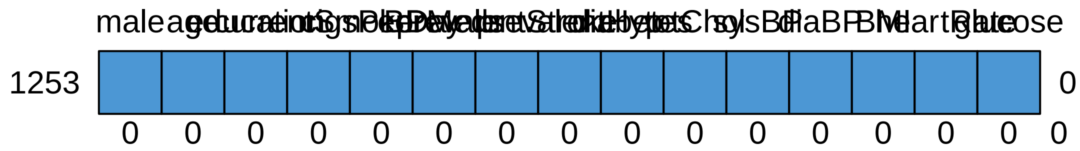
We should by now have obtained completely imputed data sets. To see this, we consider the missing data patterns of the newly constructed data sets.

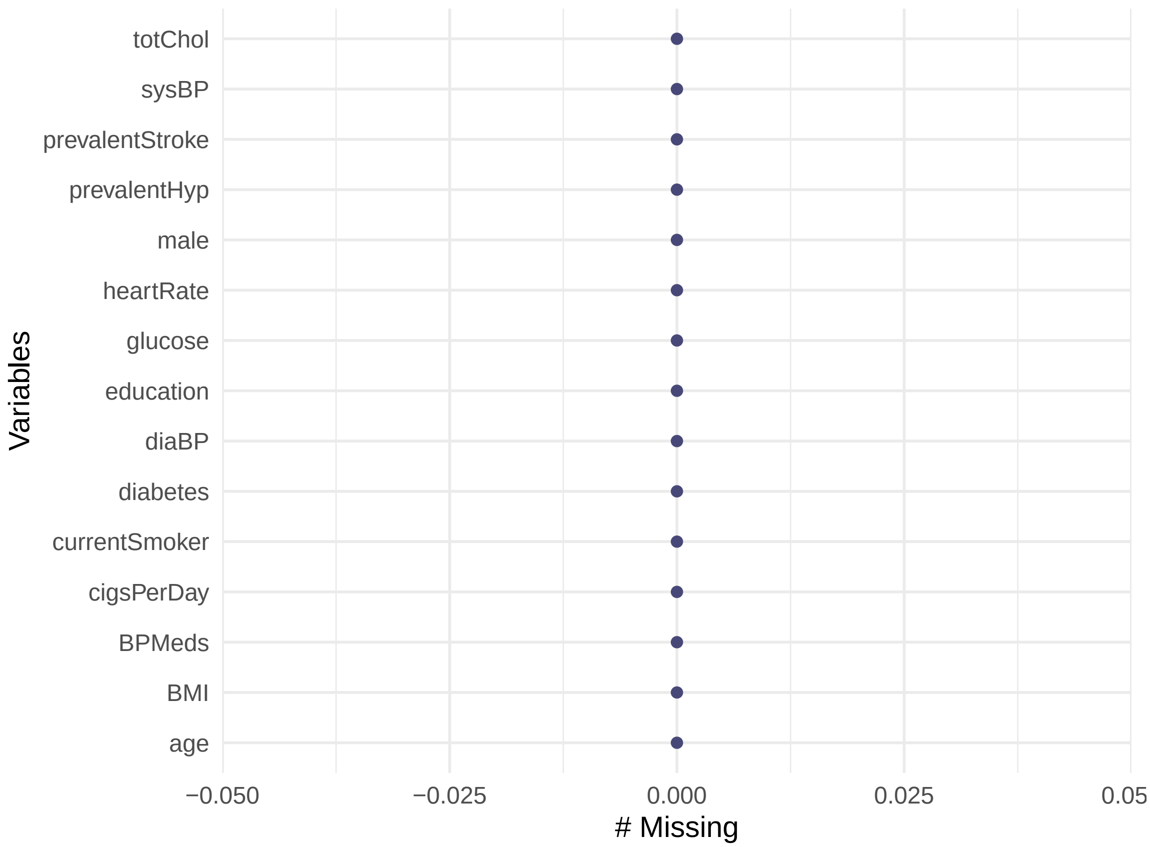
```
##  /\     /\
## {  ‘---’  }
## {  O   O  }
## ==>  V <==  No need for mice. This data set is completely observed.
##  \  \|/  /
##   ‘-----’
```


male
age
education
currentSmoker
cigsPerDay
BPMeds
prevalentStroke
prevalentHyp
diabetes
totChol
sysBP
diaBP
BMI
heartRate
glucose
2924
0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0



```
##  /\     /\
## {  ‘---’  }
## {  O   O  }
## ==>  V <==  No need for mice. This data set is completely observed.
##  \  \|/  /
##   ‘-----’
```


male
glucose
1253
0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0



There are several steps in this procedure that could be improved.

First of all, we could have used more flexible models for imputation than linear imputation, and we could have fit several different models on the training set and done evaluation procedures within the training set (e.g. cross validated optimization of ROC-AUC) to pick the models before fixing a model to impute each variable.

Some variables, such as cigsPerDay, are in some sense discrete, although we have treated them as continuous (as it is possible to smoke e.g. $2,73$ cigaretts per day). These could have been rounded off to integers, but we didn't see why this would be necessary. We did not include the response (TenYearCHD) in the regressions. It is not clear to us why it would be a better choice to include it, as we don't want the imputed data to be overfitted towards the response. We could might as well have included the response, if we wanted. More reading and testing would be needed to figure out the optimal solution, if there is any canonical choice.

The imputation was not done using the functions from the MICE package. This is partially because we wanted to get our hands dirty and do it ourselves (for practice) and partially because we couldn't figure out how to use MICE to train a regression model on the training set and predict values on the test set using the same regression model.

Model

In the model section we will consider two data sets: the complete case and imputed case. Both data sets are further divided into a train and test set.

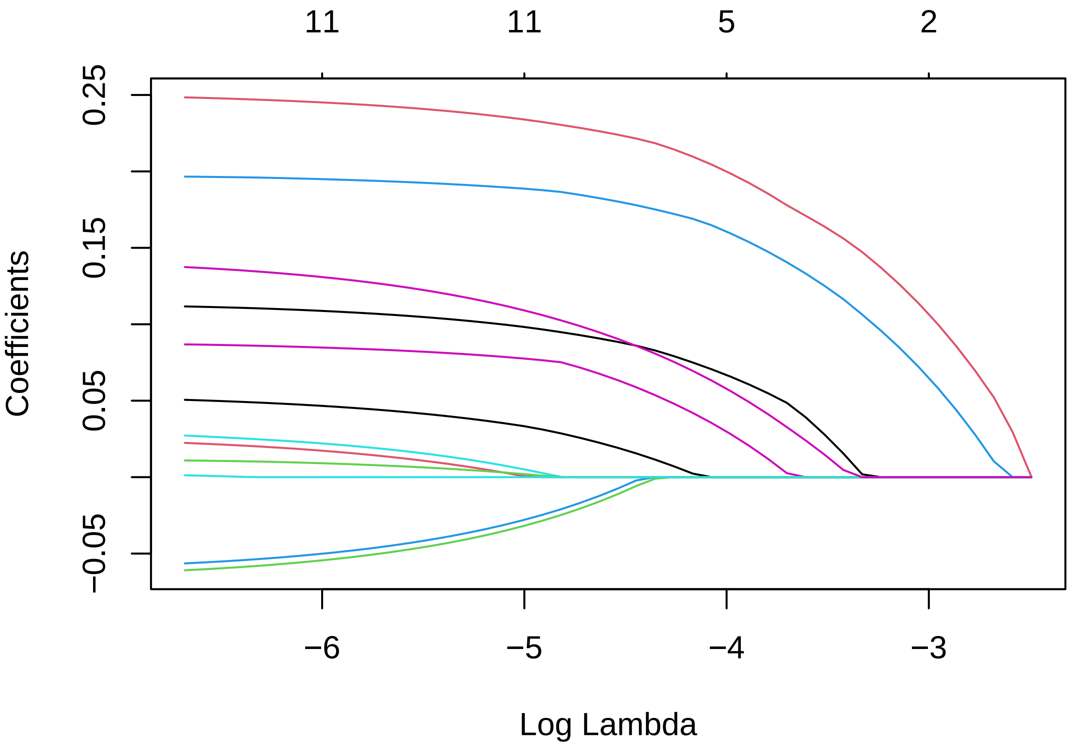
Given the binary response it is natural to consider fitting a logistic regression model to our data. Although we intend to use lasso, it is nice to start by fitting a regular logistic regression model to get an indication of which covariates that are most present, and for later comparison.

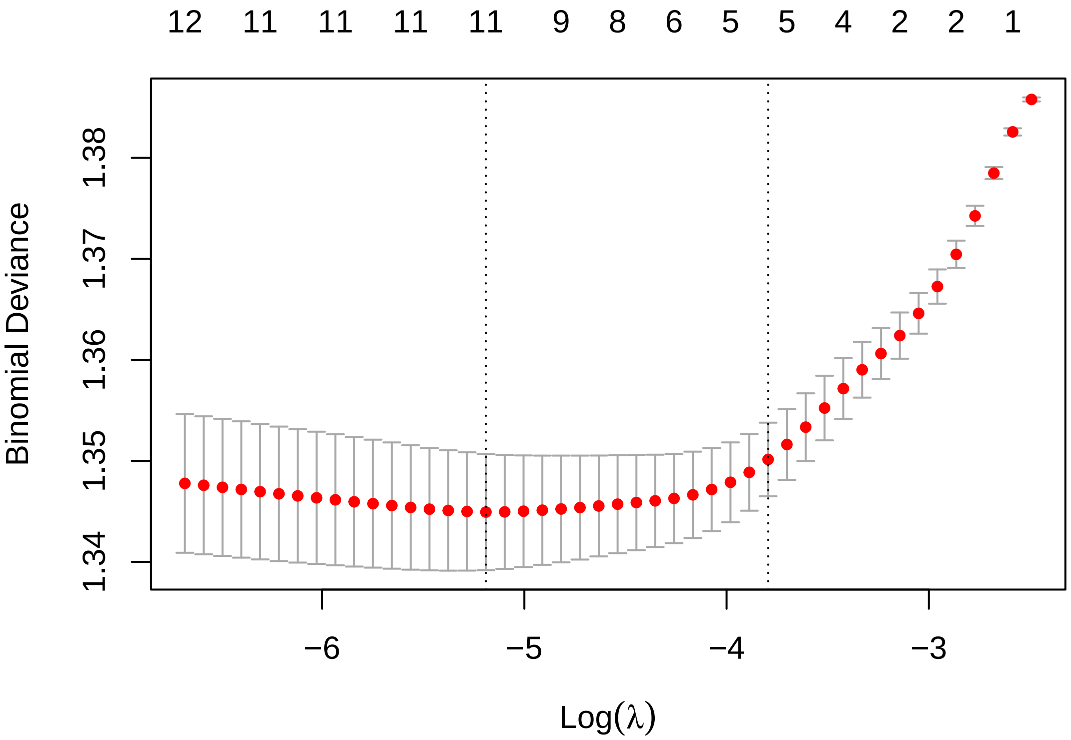
```
##
## Call:
## glm(formula = TenYearCHD ~ ., family = binomial(), data = df_complete[train,
##     ])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9021  -0.5735  -0.4083  -0.2680   2.8946
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -8.860692   0.869952 -10.185  < 2e-16 ***
## male                0.558840   0.134562   4.153 3.28e-05 ***
## age                 0.066711   0.008331   8.007 1.17e-15 ***
## educationhs        -0.326767   0.149827  -2.181  0.02919 *
## educationcollege   -0.420397   0.190222  -2.210  0.02710 *
## educationpost-grad -0.023615   0.200902  -0.118  0.90643
## currentSmoker       0.134862   0.192225   0.702  0.48294
## cigsPerDay          0.016158   0.007647   2.113  0.03461 *
## BPMeds              0.224340   0.271503   0.826  0.40864
## prevalentStroke     0.464241   0.680863   0.682  0.49534
## prevalentHyp        0.125499   0.170554   0.736  0.46183
## diabetes           -0.119936   0.392288  -0.306  0.75981
## totChol             0.001129   0.001411   0.800  0.42347
## sysBP               0.014198   0.004646   3.056  0.00225 **
## diaBP               0.001928   0.007985   0.241  0.80919
## BMI                 0.009351   0.016065   0.582  0.56051
## heartRate          -0.001216   0.005177  -0.235  0.81433
## glucose             0.008353   0.002713   3.078  0.00208 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2104.1  on 2558  degrees of freedom
## Residual deviance: 1844.3  on 2541  degrees of freedom
## AIC: 1880.3
##
## Number of Fisher Scoring iterations: 5
```

The logistic regression model chooses (Intercept), male, age, educationcollege, cigsPerDay, totChol, sysBP, glucose as the significant covariates.

Lasso on Complete Case

Next we wish to do.


11
11
5
2
0.25
0.15
0.05
−0.05
Coefficients
−6
−5
−4
−3
Log Lambda



```
## [1] 0.005570533
```

```
## [1] 0.02248834
```

```
##       (Intercept)               male               age        educationhs
##        0.00000000         0.05483208        0.18542157         0.00000000
##  educationcollege educationpost-grad     currentSmoker         cigsPerDay
##        0.00000000         0.00000000        0.00000000         0.01191032
##            BPMeds    prevalentStroke      prevalentHyp           diabetes
##        0.00000000         0.00000000        0.00000000         0.00000000
##           totChol              sysBP             diaBP                BMI
##        0.00000000         0.14738924        0.00000000         0.00000000
##         heartRate            glucose
##        0.00000000         0.04115236
```

Lasso on Imputed Data

Inference

In order to do inference we simply fit a logistic regression model using the `glm` function in `R`, and extract the inference from there. However, we will keep the coefficients chosen by the lasso-bootstrapping iteration in last section, and now use the test data to fit a logistic model to avoid overfitting.

```
##                   2.5 %      97.5 %
## (Intercept) -8.74057476 -6.12204318
## age          0.03634944  0.07736405
## male         0.21979360  0.88581576
## sysBP        0.01217782  0.02675833
```

```
##                            2.5 %       97.5 %
## (Intercept)        -10.585724659 -7.173760790
## male                 0.295723943  0.823546573
## age                  0.050471829  0.083150126
## educationhs         -0.623648808 -0.035703754
## educationcollege    -0.802829782 -0.055637061
## educationpost-grad  -0.427173917  0.362091394
## currentSmoker       -0.245039783  0.509072837
## cigsPerDay           0.001087953  0.031105481
## BPMeds              -0.318983449  0.748233142
## prevalentStroke     -0.969669245  1.759068083
## prevalentHyp        -0.210389520  0.458581614
## diabetes            -0.922972419  0.621878848
## totChol             -0.001650766  0.003883311
## sysBP                0.005095294  0.023326557
## diaBP               -0.013675064  0.017646470
## BMI                 -0.022338730  0.040685595
## heartRate           -0.011439283  0.008868168
## glucose              0.003132594  0.013820453
```

Discussion

STILL MISSING TO DO:

Is lasso implemented correctly? Why is the bootstrap so slow? Final run should be with seed(8701) and B = 1000. Copy paste the lasso code for imputed data. Finish the inference and discussion. Mainly focus on comparison of complete case vs. imputed?