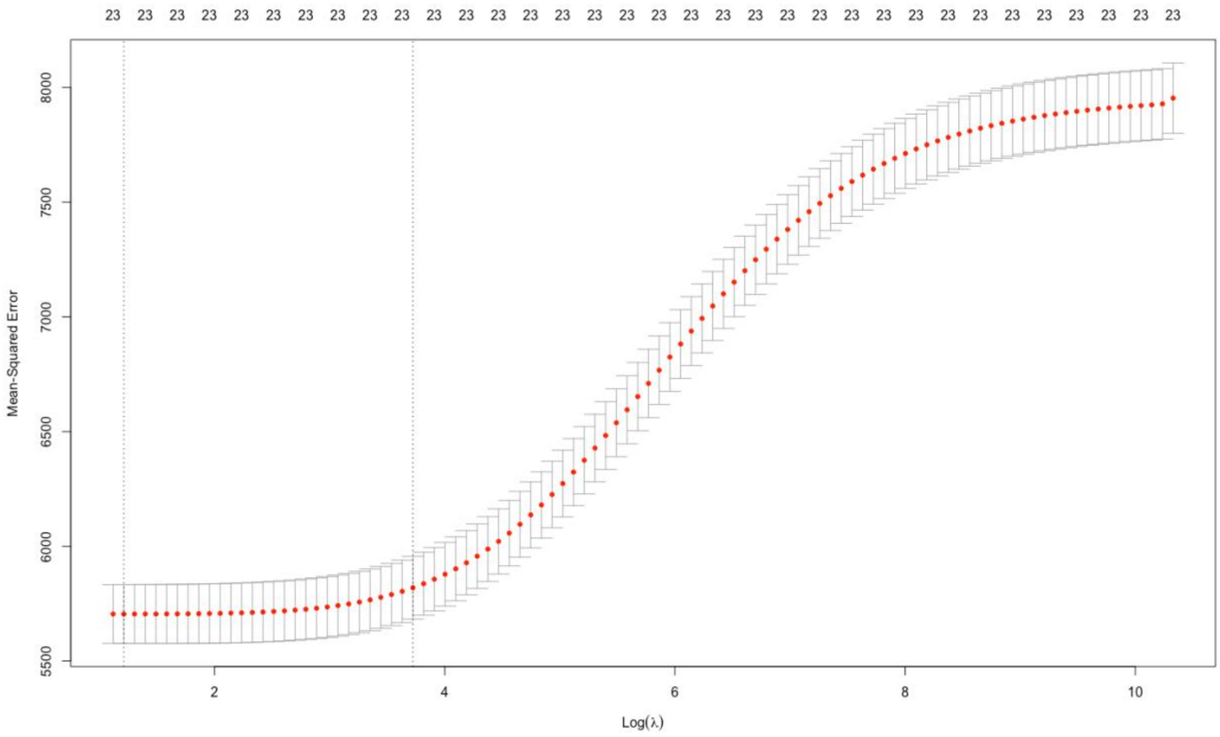
Ximan Liu
1935858
DSC 423
HW9

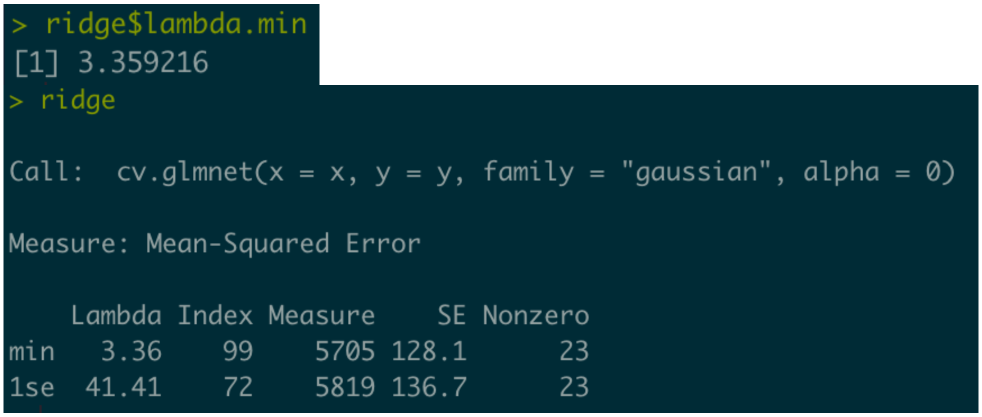
I have completed this work independently. The solutions given are entirely my own work.

1)
a)
i)
Ridge regression can be used to mitigate multicollinearity. And it estimates tend to be stable in the sense that they are usually little affected by small changes in the data on which the fitted regression is based. In contrast, ordinary least squares estimates may be highly unstable under certain conditions, for example when the independent variables are highly multicollinear.

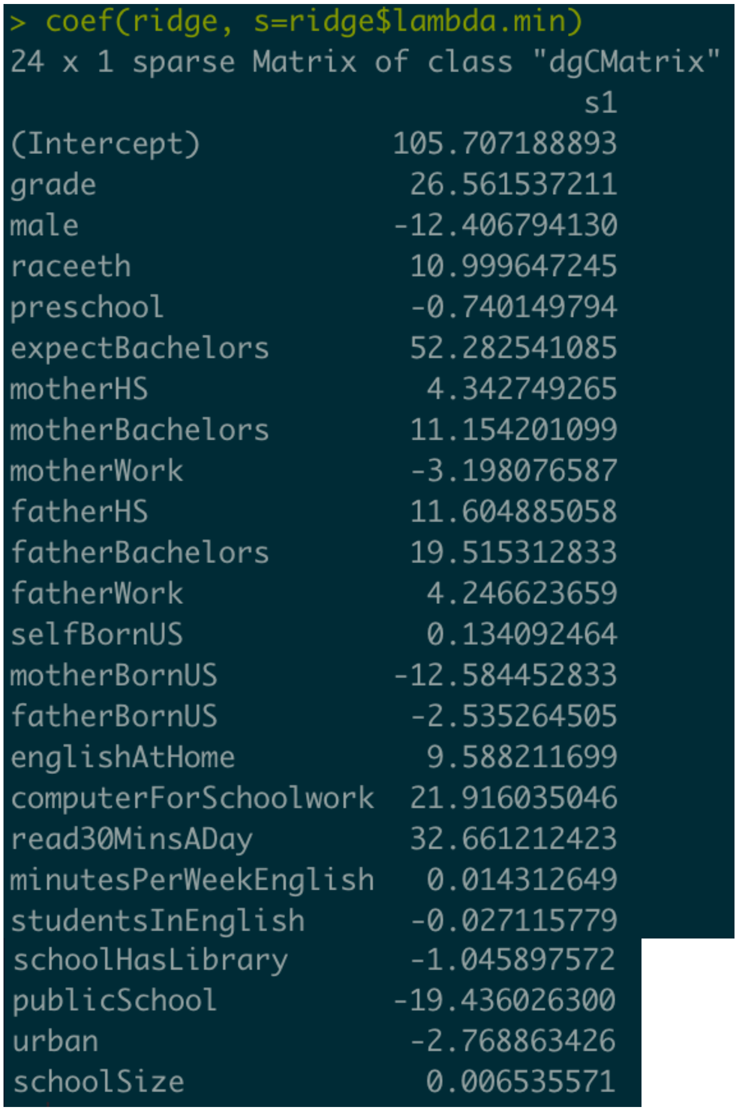
The plot shows the lambda versus the mean-squared error. The lambda shows a value of 3.359216, which minimizes the mean-squared error. Also we shows the coefficients below, which are the betas.

```
# R code:
install.packages("glmnet")
library(glmnet)

# Ridge
# alpha=0
# Multicollinearity
Pisa2009 <- Pisa2009[complete.cases(Pisa2009),]

raceeth <- c("White", "Black", "Hispanic", "More than one race", "Asian", "American Indian/Alaska Native", "Native Hawaiian/Other Pacific Islander")
raceeth.factor <- factor(raceeth)
as.numeric(raceeth.factor)

x <- data.matrix(Pisa2009[,2:24])
y <- as.double(Pisa2009[,25])

set.seed(123)
ridge <- cv.glmnet(x, y, family="gaussian", alpha=0)

plot(ridge)
ridge$lambda.min
coef(ridge, s=ridge$lambda.min)
```





```
> ridge$lambda.min
[1] 3.359216
> ridge

Call:  cv.glmnet(x = x, y = y, family = "gaussian", alpha = 0)

Measure: Mean-Squared Error

    Lambda Index Measure    SE Nonzero
min   3.36    99    5705 128.1      23
1se  41.41    72    5819 136.7      23
```



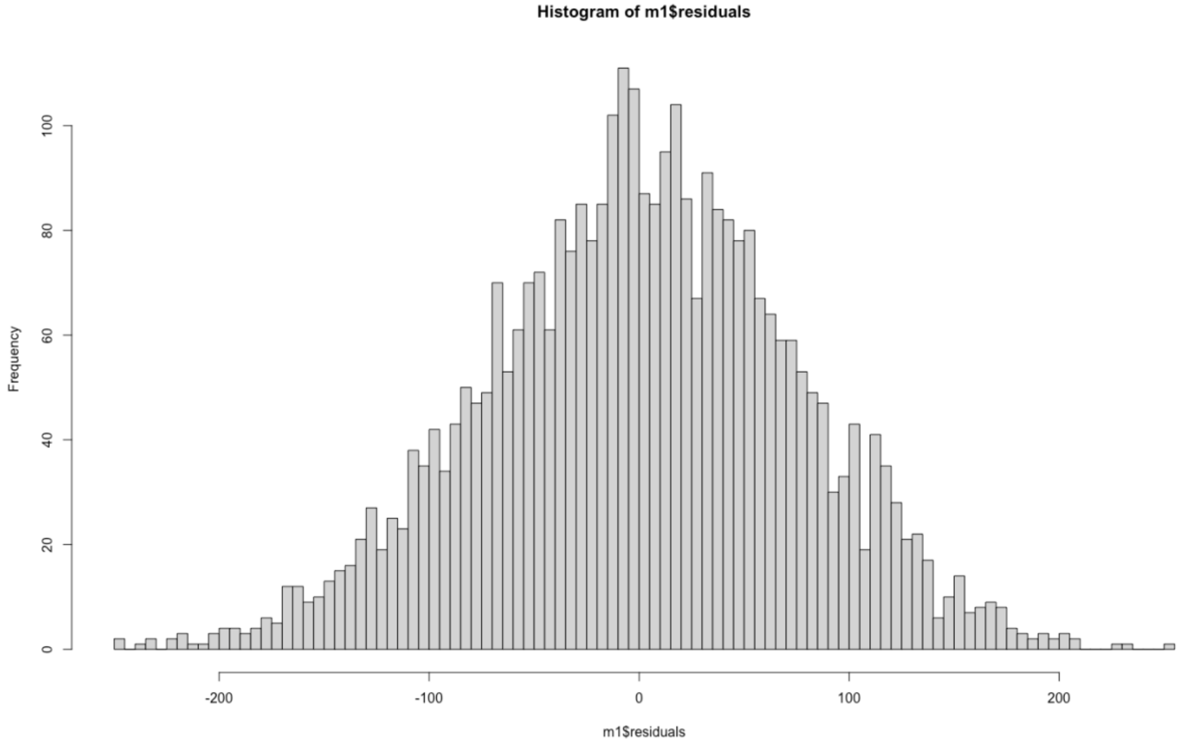
```
> coef(ridge, s=ridge$lambda.min)
24 x 1 sparse Matrix of class "dgCMatrix"
                                 s1
(Intercept)           105.707188893
grade                  26.561537211
male                  -12.406794130
raceeth                10.999647245
preschool              -0.740149794
expectBachelors        52.282541085
motherHS                4.342749265
motherBachelors        11.154201099
motherWork             -3.198076587
fatherHS               11.604885058
fatherBachelors        19.515312833
fatherWork              4.246623659
selfBornUS              0.134092464
motherBornUS          -12.584452833
fatherBornUS           -2.535264505
englishAtHome           9.588211699
computerForSchoolwork  21.916035046
read30MinsADay         32.661212423
minutesPerWeekEnglish   0.014312649
studentsInEnglish      -0.027115779
schoolHasLibrary       -1.045897572
publicSchool          -19.436026300
urban                  -2.768863426
schoolSize              0.006535571
```

ii)

After selecting variables with ridge regression, we build a model using lm and plot the model as showed below. As the histogram shows the residuals, we find that the graph is normal distributed, relatively symmetrical, and unbiased. And there are no relatively extreme outliers.

# R code:
m1 <- lm(readingScore ~ grade + male + raceeth + preschool + expectBachelors + motherHS + motherBachelors + motherWork + fatherHS + fatherBachelors + fatherWork + selfBornUS + motherBornUS + fatherBornUS + englishAtHome + computerForSchoolwork + read30MinsADay + minutesPerWeekEnglish + studentsInEnglish + schoolHasLibrary + publicSchool + urban + schoolSize, data = Pisa2009)
summary(m1)

```
m1$residuals
sum(m1$residuals)
hist(m1$residuals, breaks = 100)
```



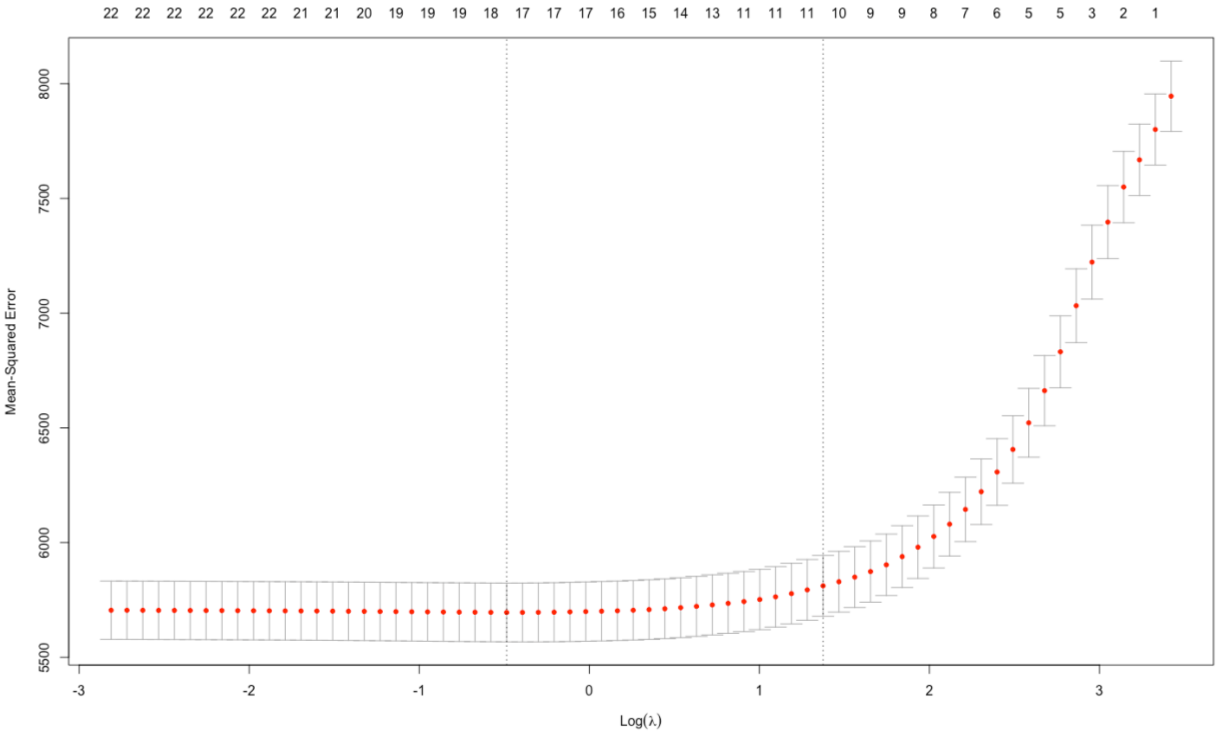
b)

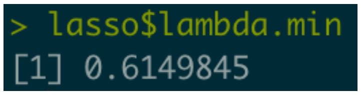
LASSO regression can be used to select features, it is a form of continuous feature selection. To run LASSO, we need to create separate structures for the dependent variable and the independent variables. The penalty factor in LASSO affects how many features are kept; choosing the penalty factor via cross-validation ensures that the model will generalize well to subsequent data samples.

Looking at the coefficients, it appears that preschool, selfBornUS, fatherBornUS, studentsInEnglish, schoolHasLibrary and urban were removed from the model.

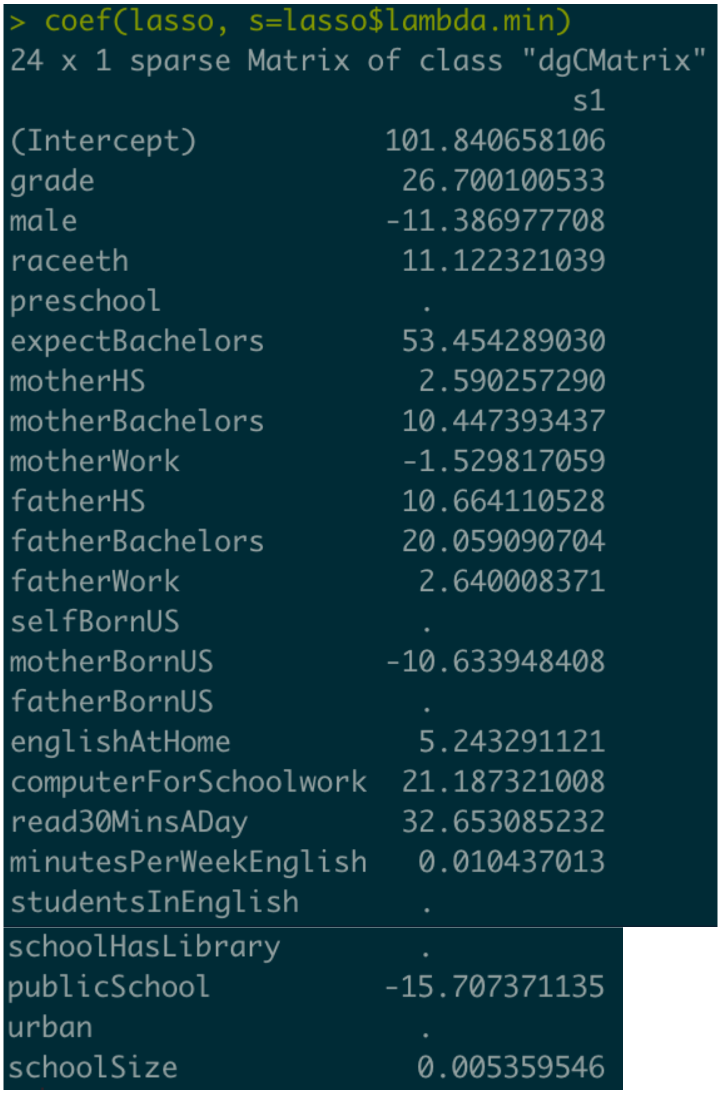
```
# R code:
# Lasso
# Feature selection
# alpha=1
Pisa2009 <- Pisa2009[complete.cases(Pisa2009),]

raceeth <- c("White", "Black", "Hispanic", "More than one race", "Asian", "American Indian/Alaska Native", "Native Hawaiian/Other Pacific Islander")
```

```
raceeth.factor <- factor(raceeth)
as.numeric(raceeth.factor)

x <- data.matrix(Pisa2009[,2:24])
y <- as.double(Pisa2009[,25])

set.seed(123)
lasso <- cv.glmnet(x, y, family="gaussian", alpha=1)

plot(lasso)
lasso$lambda.min
coef(lasso, s=lasso$lambda.min)
```





```
> lasso$lambda.min
[1] 0.6149845
```



```
> coef(lasso, s=lasso$lambda.min)
24 x 1 sparse Matrix of class "dgCMatrix"
                                 s1
(Intercept)           101.840658106
grade                  26.700100533
male                  -11.386977708
raceeth                11.122321039
preschool               .
expectBachelors        53.454289030
motherHS                2.590257290
motherBachelors        10.447393437
motherWork             -1.529817059
fatherHS               10.664110528
fatherBachelors        20.059090704
fatherWork              2.640008371
selfBornUS              .
motherBornUS          -10.633948408
fatherBornUS            .
englishAtHome           5.243291121
computerForSchoolwork  21.187321008
read30MinsADay         32.653085232
minutesPerWeekEnglish   0.010437013
studentsInEnglish       .
schoolHasLibrary        .
publicSchool          -15.707371135
urban                   .
schoolSize              0.005359546
```

c)
No. The two models are not the same.

Looking at the coefficients of LASSO, it appears that preschool, selfBornUS, fatherBornUS, studentsInEnglish, schoolHasLibrary and urban were removed from the LASSO model.

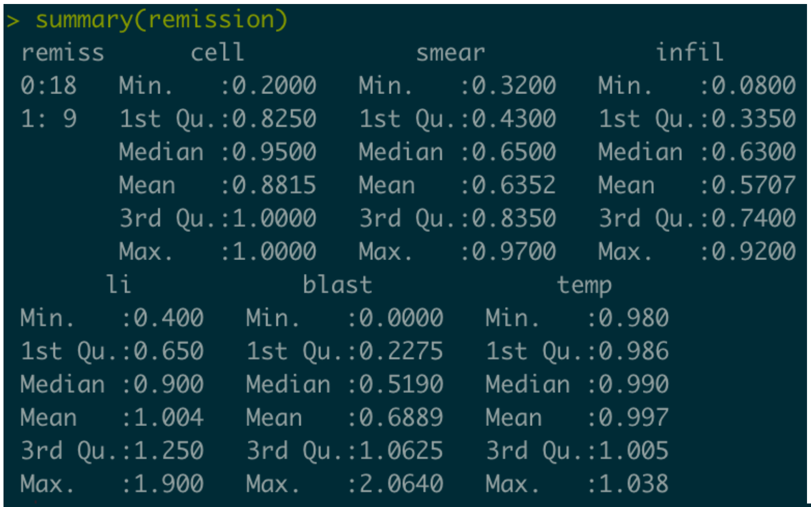
LASSO is a method for reducing the amount of features in a model that is based on sound principles. If our primary goal is prediction and gathering information about all the features isn't prohibitively costly, we may not need to utilize feature selection at all and instead rely on ridge regression to keep track of all the predictors in the model. LASSO is an excellent choice if we need to reduce the number of predictors for practical reasons. However, all it does is provide us with a helpful collection of selected predictors, which aren't always the most essential in a broad sense.

2)

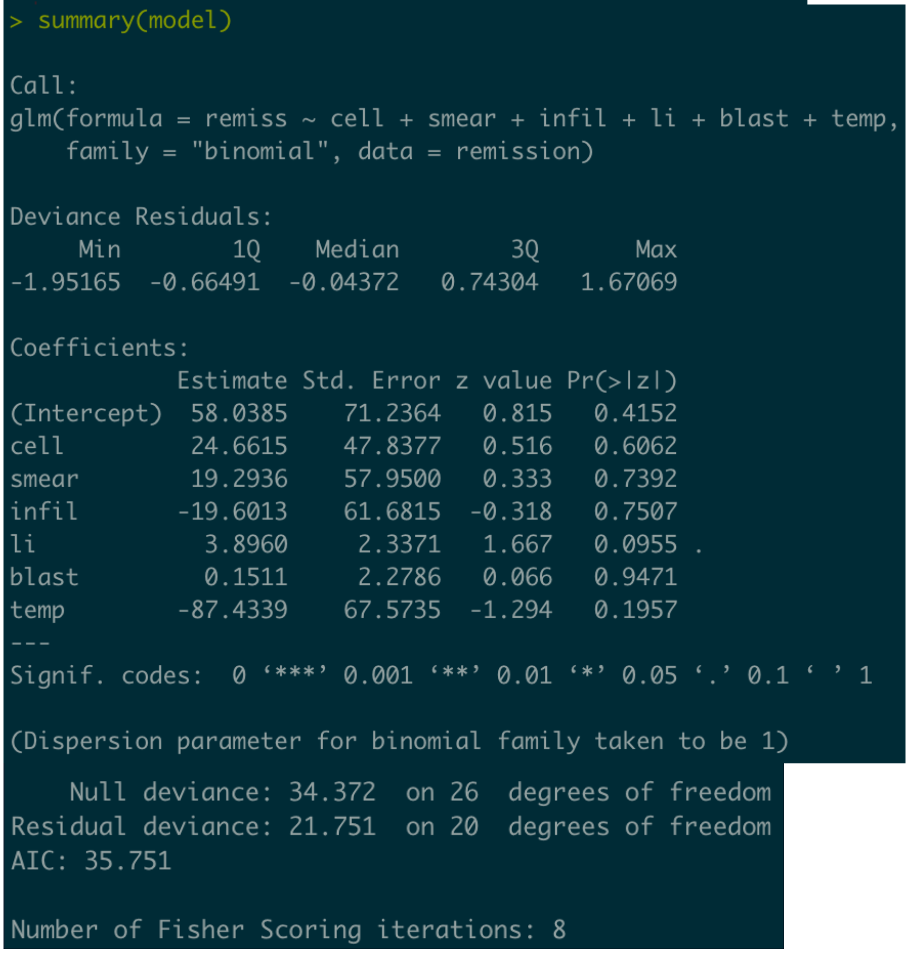
a)

Initial full model:

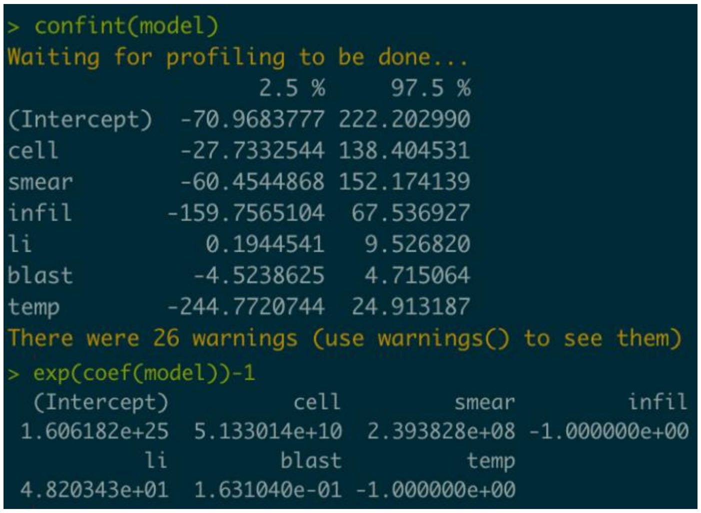
remiss = 58.0385 + 24.6615 * cell + 19.2936 * smear - 19.6013 * infil + 3.8960 * li + 0.1511 * blast -87.4339 * temp + e



```
> summary(remission)
 remiss      cell             smear             infil       
 0:18   Min.   :0.2000   Min.   :0.3200   Min.   :0.0800  
 1: 9   1st Qu.:0.8250   1st Qu.:0.4300   1st Qu.:0.3350  
        Median :0.9500   Median :0.6500   Median :0.6300  
        Mean   :0.8815   Mean   :0.6352   Mean   :0.5707  
        3rd Qu.:1.0000   3rd Qu.:0.8350   3rd Qu.:0.7400  
        Max.   :1.0000   Max.   :0.9700   Max.   :0.9200  
       li            blast             temp      
 Min.   :0.400   Min.   :0.0000   Min.   :0.980  
 1st Qu.:0.650   1st Qu.:0.2275   1st Qu.:0.986  
 Median :0.900   Median :0.5190   Median :0.990  
 Mean   :1.004   Mean   :0.6889   Mean   :0.997  
 3rd Qu.:1.250   3rd Qu.:1.0625   3rd Qu.:1.005  
 Max.   :1.900   Max.   :2.0640   Max.   :1.038  
```

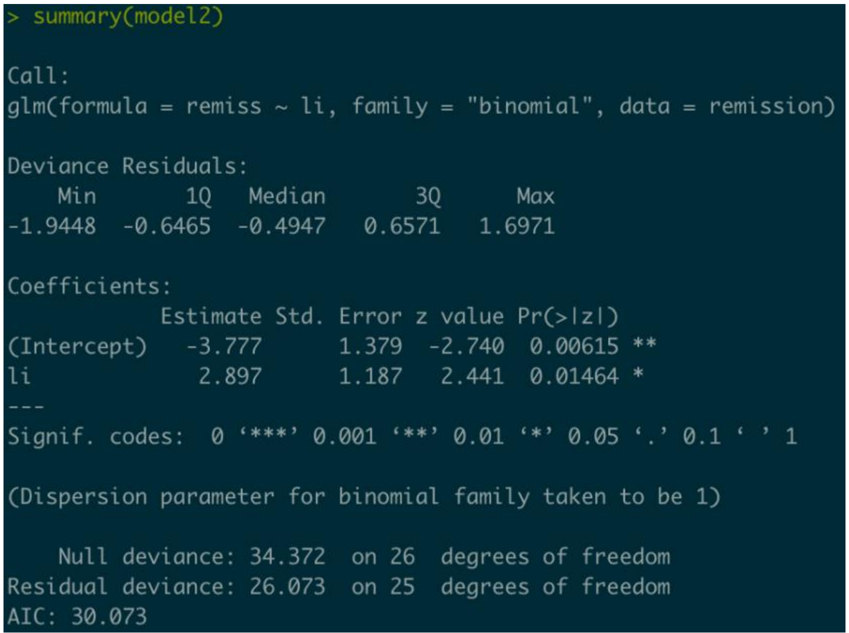


```
> summary(model)

Call:
glm(formula = remiss ~ cell + smear + infil + li + blast + temp, 
    family = "binomial", data = remission)

Deviance Residuals: 
     Min        1Q    Median        3Q       Max  
-1.95165  -0.66491  -0.04372   0.74304   1.67069  

Coefficients:
            Estimate Std. Error z value Pr(>|z|)  
(Intercept)  58.0385    71.2364   0.815   0.4152  
cell         24.6615    47.8377   0.516   0.6062  
smear        19.2936    57.9500   0.333   0.7392  
infil       -19.6013    61.6815  -0.318   0.7507  
li            3.8960     2.3371   1.667   0.0955 .
blast         0.1511     2.2786   0.066   0.9471  
temp        -87.4339    67.5735  -1.294   0.1957  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 34.372  on 26  degrees of freedom
Residual deviance: 21.751  on 20  degrees of freedom
AIC: 35.751

Number of Fisher Scoring iterations: 8
```

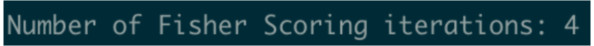


```
> confint(model)
Waiting for profiling to be done...
                   2.5 %     97.5 %
(Intercept)  -70.9683777 222.202990
cell         -27.7332544 138.404531
smear        -60.4544868 152.174139
infil       -159.7565104  67.536927
li             0.1944541   9.526820
blast         -4.5238625   4.715064
temp        -244.7720744  24.913187
There were 26 warnings (use warnings() to see them)
> exp(coef(model))-1
  (Intercept)          cell         smear         infil 
 1.606182e+25  5.133014e+10  2.393828e+08 -1.000000e+00 
           li         blast          temp 
 4.820343e+01  1.631040e-01 -1.000000e+00 
```

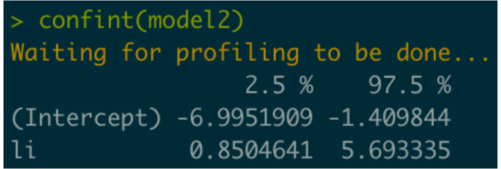
After drop the irrelative variables:
remiss = -3.777 + 2.897 * li + e



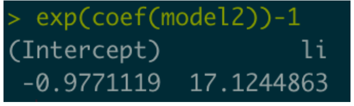
```
> summary(model2)

Call:
glm(formula = remiss ~ li, family = "binomial", data = remission)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-1.9448  -0.6465  -0.4947   0.6571   1.6971  

Coefficients:
            Estimate Std. Error z value Pr(>|z|)   
(Intercept)   -3.777      1.379  -2.740  0.00615 **
li             2.897      1.187   2.441  0.01464 * 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 34.372  on 26  degrees of freedom
Residual deviance: 26.073  on 25  degrees of freedom
AIC: 30.073
```



```
Number of Fisher Scoring iterations: 4
```



```
> confint(model2)
Waiting for profiling to be done...
                 2.5 %    97.5 %
(Intercept) -6.9951909 -1.409844
li           0.8504641  5.693335
```



```
> exp(coef(model2))-1
(Intercept)          li 
 -0.9771119  17.1244863 
```

```
# R code:
install.packages("glmnet")
library(glmnet)

# Logistic model
summary(remission)

remission$remiss <- factor(remission$remiss)

model <- glm(remiss ~ cell + smear + infil + li + blast + temp, data = remission, family =
"binomial")
summary(model)
confint(model)
exp(coef(model))-1

model2 <- glm(remiss ~ li, data = remission, family = "binomial")
summary(model2)
confint(model2)
exp(coef(model2))-1
```

b)
lm is used to fit linear regression models.

glm is used to fit generalized linear models.
It can also be used to fit more complex models like:
Poisson regression model (family=poisson)
logistic regression model (family=binomial)

In logistic regression, the dependent variable is the log odds of an event occurring. Recall, precision, specificity, and accuracy can be used to evaluate a logistic model.

c)
Initial full model:
remiss = 58.0385 + 24.6615 * cell + 19.2936 * smear - 19.6013 * infil + 3.8960 * li + 0.1511 * blast -87.4339 * temp + e

After drop the irrelative variables, the final model:
remiss = -3.777 + 2.897 * li + e

For the final model, we can de-log the coefficients, exp(coef(model2))-1 for every unit change in li, the probability of remiss changes by 17.1244863.