HW2 REPORT

Rohith Reddy Kolla | rkolla@buffalo.edu

1.

a) Dividing the data into test with 11 observations and training with 66 observations. Fitting a linear model with rating as the dependent variable and all other variables except name, mfr and type. The MSE for training comes out to be 7.052 * 10^-14 whereas the MSE for test is 1.429 ^ 10^-13.

```
Call:
lm(formula = rating ~ . - name - mfr - type, data = cereal_train)

Residuals:
       Min         1Q     Median         3Q        Max
-4.802e-07 -2.255e-07  6.692e-08  1.921e-07  4.518e-07

Coefficients:
             Estimate Std. Error    t value Pr(>|t|)
(Intercept)  5.493e+01  3.751e-07  1.464e+08   <2e-16 ***
calories    -2.227e-01  5.554e-09 -4.010e+07   <2e-16 ***
protein      3.273e+00  5.859e-08  5.587e+07   <2e-16 ***
fat         -1.691e+00  6.284e-08 -2.692e+07   <2e-16 ***
sodium      -5.449e-02  5.293e-10 -1.029e+08   <2e-16 ***
fiber        3.443e+00  4.413e-08  7.803e+07   <2e-16 ***
carbo        1.092e+00  1.744e-08  6.266e+07   <2e-16 ***
sugars      -7.249e-01  1.881e-08 -3.853e+07   <2e-16 ***
potass      -3.399e-02  1.546e-09 -2.199e+07   <2e-16 ***
vitamins    -5.121e-02  2.473e-09 -2.071e+07   <2e-16 ***
shelf       -1.941e-08  5.596e-08 -3.470e-01    0.730
weight      -3.561e-07  5.254e-07 -6.780e-01    0.501
cups         1.498e-07  2.013e-07  7.440e-01    0.460
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.963e-07 on 53 degrees of freedom
Multiple R-squared:      1,     Adjusted R-squared:      1
F-statistic: 1.313e+16 on 12 and 53 DF,  p-value: < 2.2e-16
```

fit.test.mse	1.42934143844153e-13
fit.train.mse	7.05232502089955e-14

b) Performing backward subset selection –

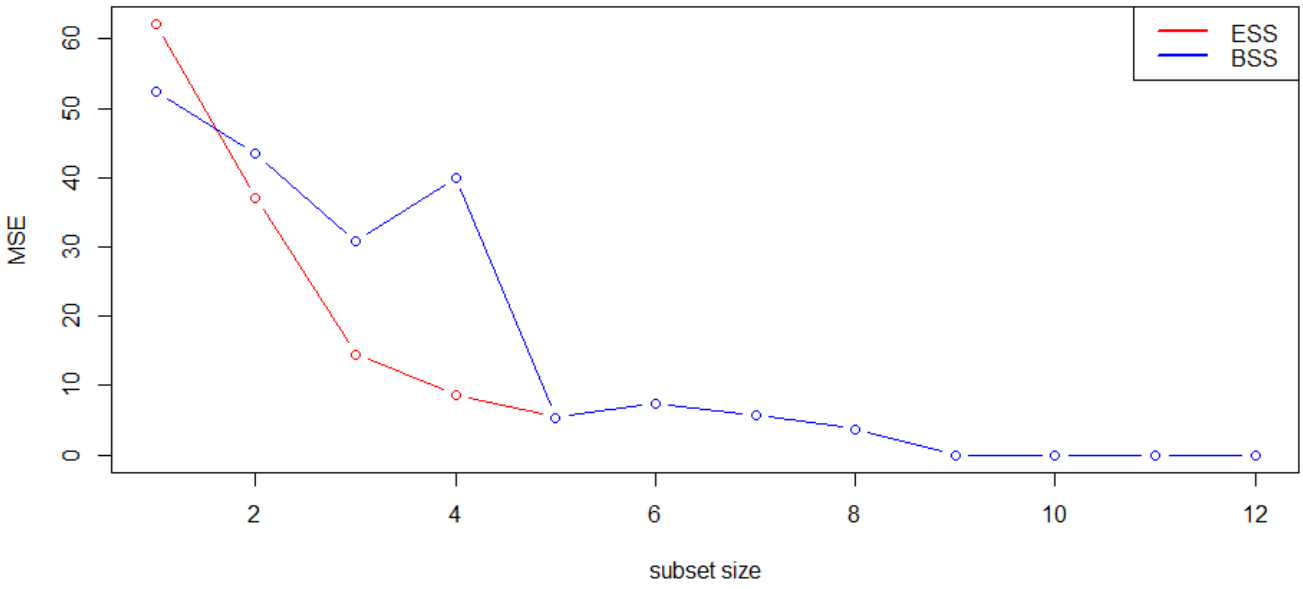
```
> bss.summary$outmat
         calories protein fat sodium fiber carbo sugars potass vitamins shelf weight cups
1  ( 1 ) "*"      " "     " " " "    " "   " "   " "    " "    " "      " "   " "    " "
2  ( 1 ) "*"      " "     " " " "    "*"   " "   " "    " "    " "      " "   " "    " "
3  ( 1 ) "*"      " "     " " " "    "*"   "*"   " "    " "    " "      " "   " "    " "
4  ( 1 ) "*"      " "     " " "*"    "*"   "*"   " "    " "    " "      " "   " "    " "
5  ( 1 ) "*"      "*"     " " "*"    "*"   "*"   " "    " "    " "      " "   " "    " "
6  ( 1 ) "*"      "*"     " " "*"    "*"   "*"   "*"    " "    " "      " "   " "    " "
7  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    " "    " "      " "   " "    " "
8  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    " "      " "   " "    " "
9  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    " "
10  ( 1 ) "*"     "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    "*"
11  ( 1 ) "*"     "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   "*"    "*"
12  ( 1 ) "*"     "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   "*"    "*"
```

c) Performing exhaustive subset selection -

```
> ess.summary$outmat
         calories protein fat sodium fiber carbo sugars potass vitamins shelf weight cups
1  ( 1 )  " "      " "     " " " "    " "   " "   "*"    " "    " "      " "   " "    " "
2  ( 1 )  " "      " "     " " " "    "*"   " "   "*"    " "    " "      " "   " "    " "
3  ( 1 )  " "      " "     " " "*"    "*"   " "   "*"    " "    " "      " "   " "    " "
4  ( 1 )  " "      " "     "*" "*"    "*"   " "   "*"    " "    " "      " "   " "    " "
5  ( 1 )  "*"      "*"     " " "*"    "*"   "*"   " "    " "    " "      " "   " "    " "
6  ( 1 )  "*"      "*"     " " "*"    "*"   "*"   "*"    " "    " "      " "   " "    " "
7  ( 1 )  "*"      "*"     "*" "*"    "*"   "*"   "*"    " "    " "      " "   " "    " "
8  ( 1 )  "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    " "      " "   " "    " "
9  ( 1 )  "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    " "
10  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    "*"
11  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   "*"    "*"
12  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   "*"    "*"
```

From b) and c) the least significant variables for the prediction with both subset selection methods happen to be shelf, weight and cups while the most significant variables differ significantly based on the number of subsets.

d) Comparing the prediction error of the test set with the two methods of subset selection.




```
> test.bss.err
 [1] 5.236577e+01 4.340967e+01 3.076223e+01 4.005824e+01 5.399439e+00 7.351198e+00
 [7] 5.794310e+00 3.714971e+00 1.509865e-13 1.466985e-13 1.456934e-13 1.429341e-13
> test.ess.err
 [1] 6.213739e+01 3.709110e+01 1.453277e+01 8.733913e+00 5.399439e+00 7.351198e+00
 [7] 5.794310e+00 3.714971e+00 1.509865e-13 1.466985e-13 1.456934e-13 1.429341e-13
```

Exhaustive subset selection has always been the more accurate subset selection with the drawback being that it requires more computational power. In the case of 1 subset, the fact that BSS does better than ESS could be attributed to the test set being more suited for BSS prediction. In every other case, ESS performs equally or better as is expected.

From the graph and data above, it appears that the best model for prediction has the 9 predictors – Calories, Protein, Fat, Sodium, Fiber, Carbo, Sugars, Potass and Vitamins with a mean squared error of 1.5*10^-13. The addition of the other 3 predictors brings insignificant change to the predictions.

The upward trend in the BSS mean square error at 4 subsets could be explained by the fact that the predictors have to be a subset of the previous prediction (Since it is BSS) and therefore won't necessarily be the best predictors for a given number of subsets.

2) Fitting a linear model for the training data considering only 2's and 3's –

```
Call:
lm(formula = V1 ~ ., data = train_data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.59281 -0.09679 -0.00057  0.09365  0.70122
```

```
Residual standard error: 0.1745 on 1132 degrees of freedom
Multiple R-squared:  0.9005,    Adjusted R-squared:  0.878
F-statistic: 40.01 on 256 and 1132 DF,  p-value: < 2.2e-16
```

The Training and Test mean squared error for the linear fit comes out to be –

test_MSE	0.15166534519127
train_MSE	0.0248117215487067

Performing KNN with k of values 1,3,5,7,9,11,13,15 –

	test_predict		
testing$V1	2	3	Row Total
2	192	6	198
	69.611	80.320	
	0.970	0.030	0.544
	0.985	0.036	
	0.527	0.016	
3	3	163	166
	83.030	95.804	
	0.018	0.982	0.456
	0.015	0.964	
	0.008	0.448	
Column Total	195	169	364
	0.536	0.464	

[1] "KNN Test predictions for k = 1"

	test_predict		
testing$V1	2	3	Row Total
2	731	0	731
	311.709	346.291	
	1.000	0.000	0.526
	1.000	0.000	
	0.526	0.000	
3	0	658	658
	346.291	384.709	
	0.000	1.000	0.474
	0.000	1.000	
	0.000	0.474	
Column Total	731	658	1389
	0.526	0.474	

[1] "KNN Training predictions for k = 1"

Test Error = 02.4% Training Error = 0%

testing$V1	test_predict 2	3	Row Total
2	191	7	198
	68.000	78.462	
	0.965	0.035	0.544
	0.979	0.041	
	0.525	0.019	
3	4	162	166
	81.108	93.587	
	0.024	0.976	0.456
	0.021	0.959	
	0.011	0.445	
Column Total	195	169	364
	0.536	0.464	

[1] "KNN Test predictions for k = 3"

testing$V1	test_predict 2	3	Row Total
2	728	3	731
	304.976	339.791	
	0.996	0.004	0.526
	0.995	0.005	
	0.524	0.002	
3	4	654	658
	338.811	377.488	
	0.006	0.994	0.474
	0.005	0.995	
	0.003	0.471	
Column Total	732	657	1389
	0.527	0.473	

[1] "KNN Training predictions for k = 3"

Test Error = 03.02% Training Error = 0.5%

testing$V1	test_predict 2	3	Row Total
2	191	7	198
	68.000	78.462	
	0.965	0.035	0.544
	0.979	0.041	
	0.525	0.019	
3	4	162	166
	81.108	93.587	
	0.024	0.976	0.456
	0.021	0.959	
	0.011	0.445	
Column Total	195	169	364
	0.536	0.464	

[1] "KNN Test predictions for k = 5"

testing$V1	test_predict 2	3	Row Total
2	726	5	731
	305.479	337.415	
	0.993	0.007	0.526
	0.996	0.008	
	0.523	0.004	
3	3	655	658
	339.369	374.849	
	0.005	0.995	0.474
	0.004	0.992	
	0.002	0.472	
Column Total	729	660	1389
	0.525	0.475	

[1] "KNN Training predictions for k = 5"

Test Error = 03.02% Training Error = 0.57%

testing$V1	test_predict 2	3	Row Total
2	189	9	198
	68.465	76.426	
	0.955	0.045	0.544
	0.984	0.052	
	0.519	0.025	
3	3	163	166
	81.663	91.159	
	0.018	0.982	0.456
	0.016	0.948	
	0.008	0.448	
Column Total	192	172	364
	0.527	0.473	

[1] "KNN Test predictions for k = 7"

testing$V1	test_predict 2	3	Row Total
2	725	6	731
	305.053	335.973	
	0.992	0.008	0.526
	0.996	0.009	
	0.522	0.004	
3	3	655	658
	338.896	373.247	
	0.005	0.995	0.474
	0.004	0.991	
	0.002	0.472	
Column Total	728	661	1389
	0.524	0.476	

[1] "KNN Training predictions for k = 7"

Test Error = 3.29% Training Error = 0.64%

testing$V1	test_predict 2	3	Row Total
2	188	10	198
	68.083	75.167	
	0.949	0.051	0.544
	0.984	0.058	
	0.516	0.027	
3	3	163	166
	81.208	89.657	
	0.018	0.982	0.456
	0.016	0.942	
	0.008	0.448	
Column Total	191	173	364
	0.525	0.475	

[1] "KNN Test predictions for k = 9"

testing$V1	test_predict 2	3	Row Total
2	722	9	731
	302.418	331.154	
	0.988	0.012	0.526
	0.994	0.014	
	0.520	0.006	
3	4	654	658
	335.969	367.893	
	0.006	0.994	0.474
	0.006	0.986	
	0.003	0.471	
Column Total	726	663	1389
	0.523	0.477	

[1] "KNN Training predictions for k = 9"

Test Error = 3.57%

Training Error = 0.93%

testing$V1	test_predict 2	3	Row Total
2	188	10	198
	68.083	75.167	
	0.949	0.051	0.544
	0.984	0.058	
	0.516	0.027	
3	3	163	166
	81.208	89.657	
	0.018	0.982	0.456
	0.016	0.942	
	0.008	0.448	
Column Total	191	173	364
	0.525	0.475	

[1] "KNN Test predictions for k = 11"

testing$V1	test_predict 2	3	Row Total
2	723	8	731
	302.844	332.580	
	0.989	0.011	0.526
	0.994	0.012	
	0.521	0.006	
3	4	654	658
	336.442	369.477	
	0.006	0.994	0.474
	0.006	0.988	
	0.003	0.471	
Column Total	727	662	1389
	0.523	0.477	

[1] "KNN Training predictions for k = 11"

Test Error = 3.57%

Training Error = 0.86%

testing$V1	test_predict 2	3	Row Total
2	187	11	198
	67.701	73.927	
	0.944	0.056	0.544
	0.984	0.063	
	0.514	0.030	
3	3	163	166
	80.752	88.178	
	0.018	0.982	0.456
	0.016	0.937	
	0.008	0.448	
Column Total	190	174	364
	0.522	0.478	

[1] "KNN Test predictions for k = 13"

testing$V1	test_predict 2	3	Row Total
2	723	8	731
	302.844	332.580	
	0.989	0.011	0.526
	0.994	0.012	
	0.521	0.006	
3	4	654	658
	336.442	369.477	
	0.006	0.994	0.474
	0.006	0.988	
	0.003	0.471	
Column Total	727	662	1389
	0.523	0.477	

[1] "KNN Training predictions for k = 13"

Test Error = 3.84%

Training Error = 0.86%

testing$V1	test_predict 2	3	Row Total
2	187	11	198
	67.701	73.927	
	0.944	0.056	0.544
	0.984	0.063	
	0.514	0.030	
3	3	163	166
	80.752	88.178	
	0.018	0.982	0.456
	0.016	0.937	
	0.008	0.448	
Column Total	190	174	364
	0.522	0.478	

[1] "KNN Test predictions for k = 15"

Test Error = 3.84%

testing$V1	test_predict 2	3	Row Total
2	721	10	731
	303.347	330.261	
	0.986	0.014	0.526
	0.996	0.015	
	0.519	0.007	
3	3	655	658
	337.001	366.900	
	0.005	0.995	0.474
	0.004	0.985	
	0.002	0.472	
Column Total	724	665	1389
	0.521	0.479	

[1] "KNN Training predictions for k = 15"

Training Error = 0.93%

From the above results it is evident that the best test results for KNN from the given values is when k = 1 with 355 correct predictions out of 364 with a success rate of around 97.5%. The linear regression classification however seems to have only made 349 correct predictions as shown below which is worse than the results of KNN with any of the considered k values.

Linear Regression Model Predictions -

test_data$V1	pred 2	3	Row Total
2	191	7	198
	63.263	76.299	
	0.965	0.035	0.544
	0.960	0.042	
	0.525	0.019	
3	8	158	166
	75.458	91.007	
	0.048	0.952	0.456
	0.040	0.958	
	0.022	0.434	
Column Total	199	165	364
	0.547	0.453	

Test Error = 4.1%

3)

a) The data has been split into a training set containing 700 of the observations and a test set containing the remaining 77 observations. (Private variable is not considred).

Fitting a linear model to the training set -

```
> fit <- lm(Apps~., data = college_train[,-1])
> summary(fit)

Call:
lm(formula = Apps ~ ., data = college_train[, -1])

Residuals:
    Min      1Q  Median      3Q     Max
-5174.2  -445.9   -26.2   321.3  7320.2

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -9.708e+02  4.171e+02  -2.327  0.02025 *
Accept       1.608e+00  4.235e-02  37.970  < 2e-16 ***
Enroll      -9.402e-01  1.932e-01  -4.867 1.41e-06 ***
Top10perc    4.847e+01  6.028e+00   8.040 3.96e-15 ***
Top25perc   -1.331e+01  4.865e+00  -2.735  0.00640 **
F.Undergrad  7.635e-02  3.389e-02   2.253  0.02460 *
P.Undergrad  4.726e-02  3.393e-02   1.393  0.16414
Outstate    -1.179e-01  1.930e-02  -6.106 1.71e-09 ***
Room.Board   1.200e-01  5.127e-02   2.340  0.01958 *
Books        1.444e-02  2.550e-01   0.057  0.95486
Personal     9.986e-03  6.830e-02   0.146  0.88381
PhD         -8.571e+00  5.013e+00  -1.710  0.08776 .
Terminal    -1.713e-01  5.457e+00  -0.031  0.97497
S.F.Ratio    2.704e+01  1.384e+01   1.954  0.05105 .
perc.alumni -1.248e+00  4.409e+00  -0.283  0.77725
Expend       9.900e-02  1.455e-02   6.806 2.20e-11 ***
Grad.Rate    8.751e+00  3.205e+00   2.731  0.00648 **
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1070 on 683 degrees of freedom
Multiple R-squared:  0.9284,    Adjusted R-squared:  0.9267
F-statistic: 553.2 on 16 and 683 DF,  p-value: < 2.2e-16
```

Considering the fact that the residuals are enormous, the mean squared error will be significantly greater as shown below.

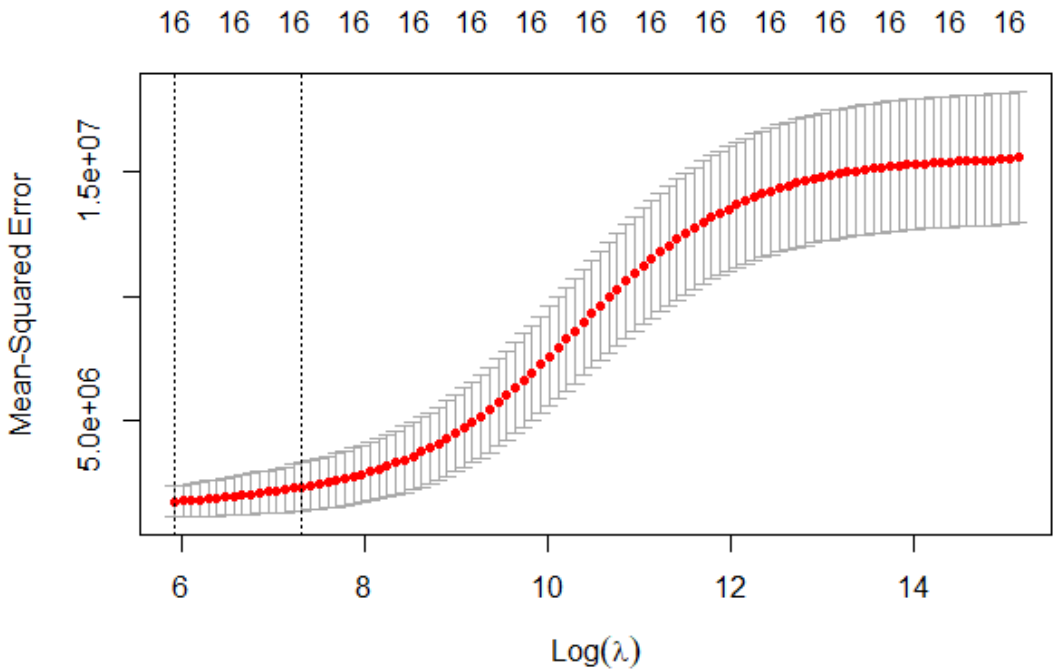
fit.test.mse	753299.815497666
fit.train.mse	1117847.53086034

b) Fitting a ridge regression model on the training set using the glmnet package –

```
# Fitting a Ridge Regression Model
ridge.mod = glmnet(as.matrix(college_train[,3:18]), college_train[,2], alpha = 0)
```

Upon cross-examination, the best lambda value turns out to be –

best_lamda	372.904453168218



The test mean squared error using the best lambda is

test_err	599864.680747206

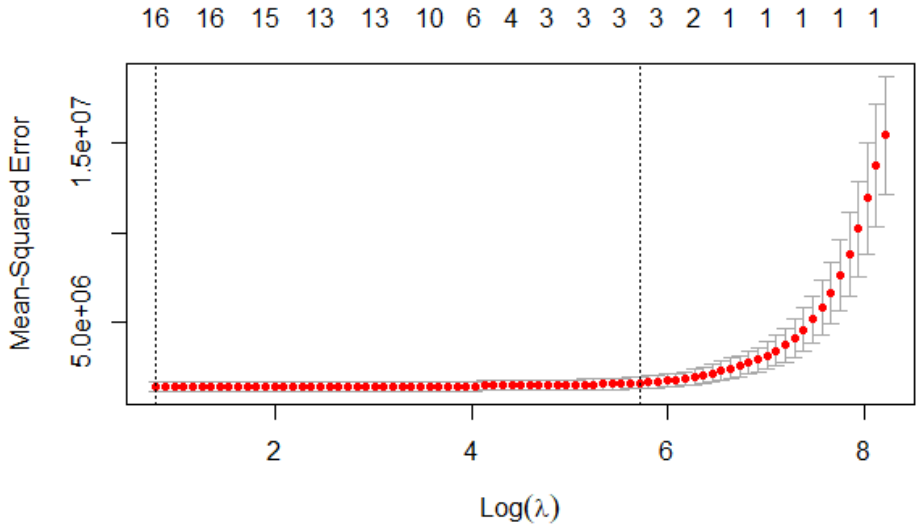
which is considerably lower than the test mean squared error using linear regression.

c) Fitting a LASSO model on the training set using the glmnet package

```
# Fitting a Lasso Model
lasso.mod <- glmnet(as.matrix(college_train[,3:18]), college_train[,2], alpha = 1)
```

Upon Cross-Examination, the best lambda turns out to be –

best_lamda_2	2.18410914175671



The test mean squared error is higher than that from Ridge at `test_err_2  743676.823592791`.

Furthermore, there are no zero coefficient estimates.

```
17 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept) -9.883371e+02
Accept       1.593464e+00
Enroll      -8.259286e-01
Top10perc    4.670414e+01
Top25perc   -1.190245e+01
F.Undergrad  6.144553e-02
P.Undergrad  4.667938e-02
Outstate    -1.150868e-01
Room.Board   1.188840e-01
Books        7.780168e-03
Personal     7.825325e-03
PhD         -8.359176e+00
Terminal    -5.578330e-02
S.F.Ratio    2.587183e+01
perc.alumni -1.412061e+00
Expend       9.779808e-02
Grad.Rate    8.339086e+00
```