# MSIT 423

# Project Two

*Ray Liu*

## Features and model description

The data set is from a crowd-sourcing website that enables users to submit and discuss ideas to improve the product. After comparing a different kind of classifier, I get the result (AUC) below:

| **Model** | **Contributor + Content** | **All** |
|---|---|---|
| **Lasso** | 0.618 | 0.765 |
| **Ridge** | 0.622 | 0.749 |
| **GAM** | 0.599 | 0.792 |
| **Tree** | 0.719 | 0.911 |
| **Random Forest** | 0.748 | 0.943 |
| **GBM** | 0.744 | 0.946 |

The predictor's analysis

| Predictors | Definition | Category |
|---|---|---|
| Pastaccept | Number of ideas accepted in the past | Contributor |
| commentsC | Number of comments written by the contributor | Contributor |
| X1-X11 | Summary of what is in the text | Content |
| age | How long the idea has been submitted | Content |
| month | The month when it was submitted | Content |
| diversity | How different the idea is from previous ideas | Content |
| comments | Number of comments written about the idea | Crowd |
| votes | The number of people who visited the side and voted for implementing the idea | Crowd |

# Data preview:

## 2.1 Preliminary analysis:

### 2.1.1 The profile of the data

When I view the dimension of the data, I found that they are heavily biased. The mean is only 0.8941, indicating that most of the response is 0. It may produce some problem in prediction, accuracy, for instance, can be extremely high even if I predict all they as 0 without any model. So, AUC may be a better criterion to evaluate the results.

```
       y
Min.   :0.00000
1st Qu.:0.00000
Median :0.00000
Mean   :0.08941
3rd Qu.:0.00000
Max.   :1.00000
```

### 2.1.2 Correlation analysis

According to the correlation matrix, luckily, there is not much correlation between these variables. There is only two combinations of variables have correlation more than 0.6, the age and pastideas (0.84), the pastaccept and pastideas.

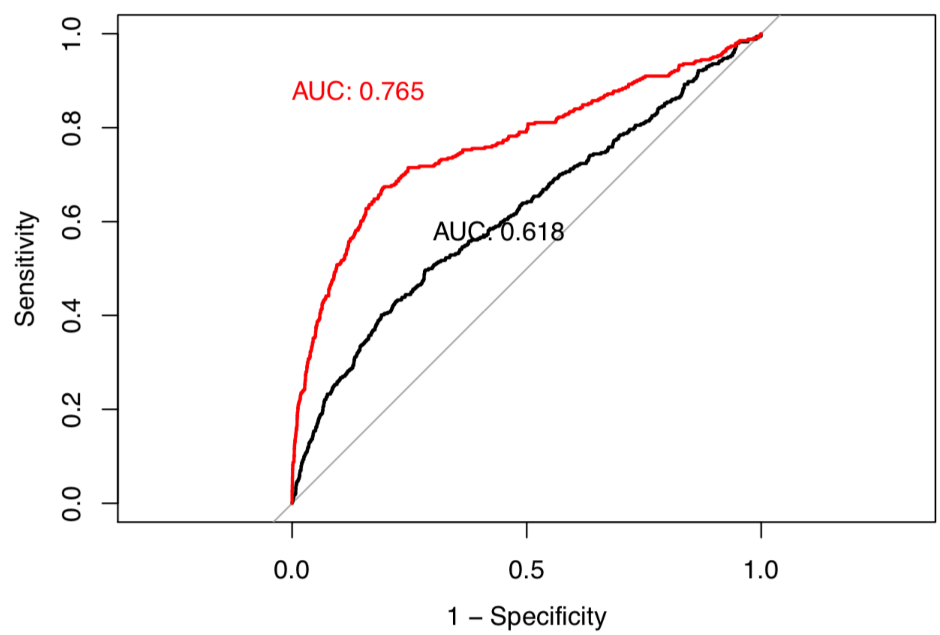
```
##             month diversity pastideas pastaccept commentsC   age votes
## month        1.00      0.02     -0.01       0.01      0.03 -0.01  0.02
## diversity    0.02      1.00     -0.03      -0.02     -0.03 -0.04 -0.01
## pastideas   -0.01     -0.03      1.00       0.64      0.40  0.84  0.01
## pastaccept   0.01     -0.02      0.64       1.00      0.33  0.48  0.04
## commentsC    0.03     -0.03      0.40       0.33      1.00  0.44  0.03
## age         -0.01     -0.04      0.84       0.48      0.44  1.00  0.01
## votes        0.02     -0.01      0.01       0.04      0.03  0.01  1.00
## comments    -0.01     -0.05      0.03       0.05      0.08  0.03  0.42
## X1           0.03      0.42     -0.09      -0.06     -0.04 -0.10 -0.01
## X2          -0.02     -0.16      0.05       0.05      0.03  0.06 -0.04
## X3          -0.01     -0.01     -0.03      -0.03     -0.01 -0.02  0.04
## X4          -0.01     -0.39     -0.01      -0.04      0.00  0.00  0.03
## X5           0.02      0.00      0.00       0.04      0.01  0.00  0.01
## X6           0.00      0.07      0.02       0.03      0.00  0.01  0.00
## X7           0.00     -0.10     -0.02       0.01      0.05  0.00  0.04
## X8           0.02     -0.23     -0.02      -0.02     -0.02 -0.02 -0.05
## X9           0.00     -0.46      0.00       0.02      0.02  0.00 -0.01
## X10         -0.01      0.27      0.01      -0.02     -0.01  0.00  0.01
## X11         -0.01      0.20     -0.03      -0.04     -0.02 -0.03 -0.03
## y           -0.01     -0.02      0.03       0.06      0.09  0.03  0.33
```

Pic2.1.2 Part of the correlation matrix

# Classifier Model

## 3.1 Lasso

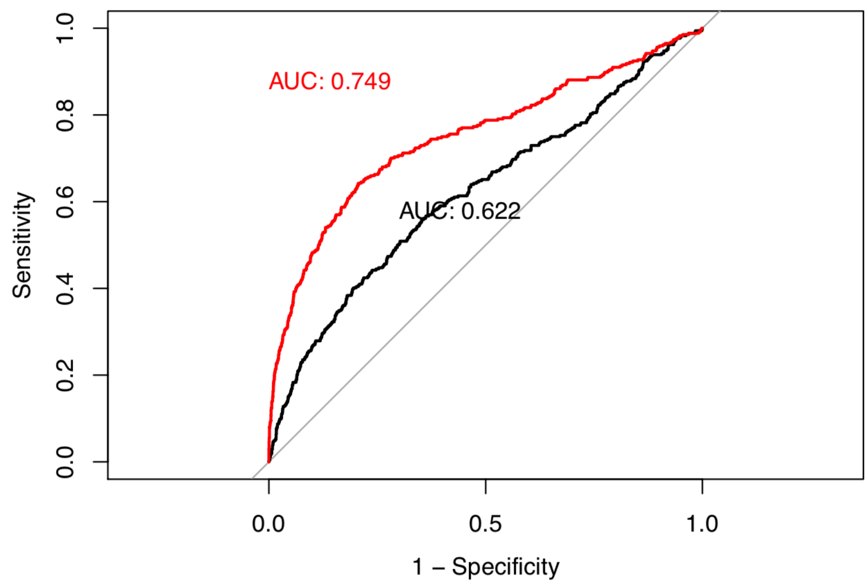
Given the optimal lambda, only X2 has been deleted from the model. We have the MSE of test dataset 0.0864 and 0.0767 for adding the predictors besides contributor and content predictors. When calculating the AUC, we find the crowd predictors increases the AUC by 0.147.




## 3.2 Ridge

Once again, we use Ridge to find the optimal model in the same way. When only consider contributor and content predictors, we get MSE of 0.0862 while 0.0771 for all the predictors. After calculating AUC, the crowd predictors increases the AUC by 0.127.

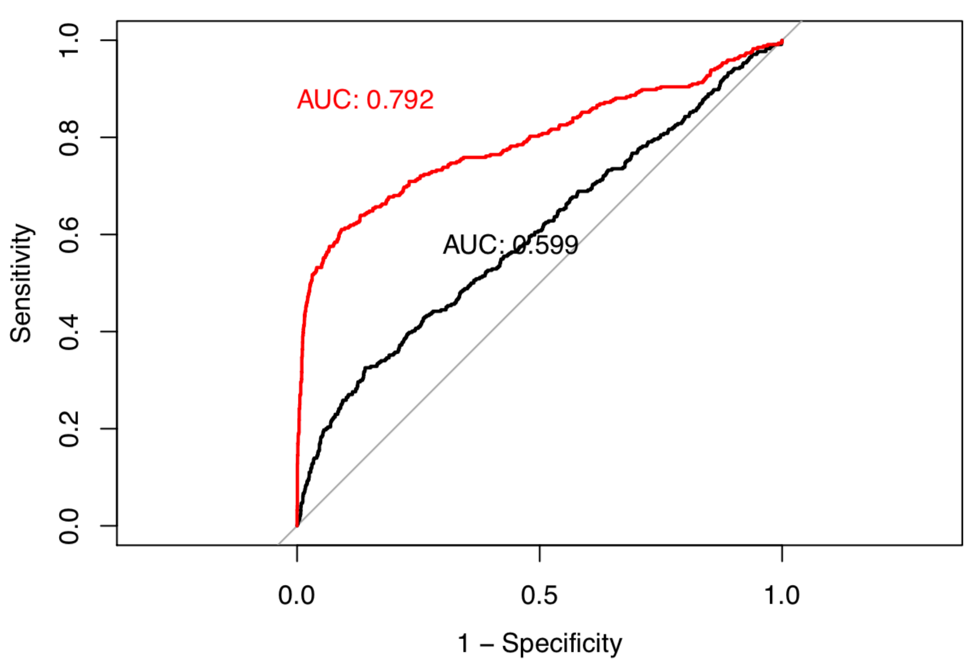
Compare the Ridge and Lasso, we find that Ridge does better when only consider contributor and content predictors. When taking all the predictors into consideration, Lasso does better than Ridge.




## 3.3 GAM

I tried the general additive model, assuming there is no interaction between the predictors. Through the partial depend on digraph, the votes and comments seem to be significant in a certain range. The digraph has been attached in the appendix.

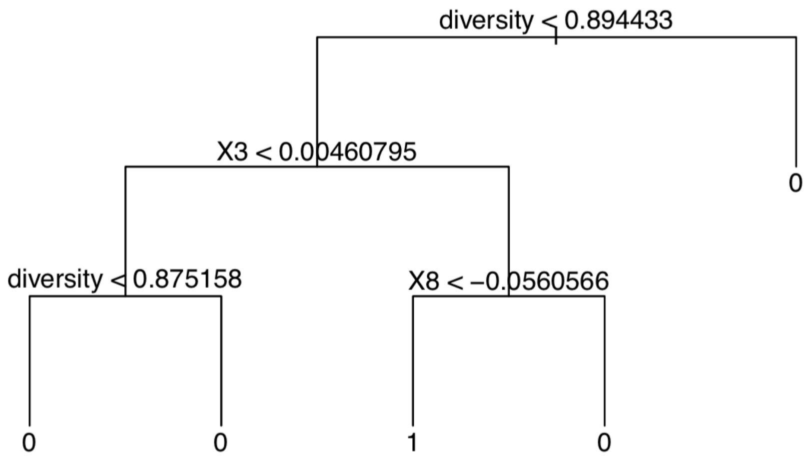
And we find that the AUC of GAM (taking all predictors into consideration) is the best now.



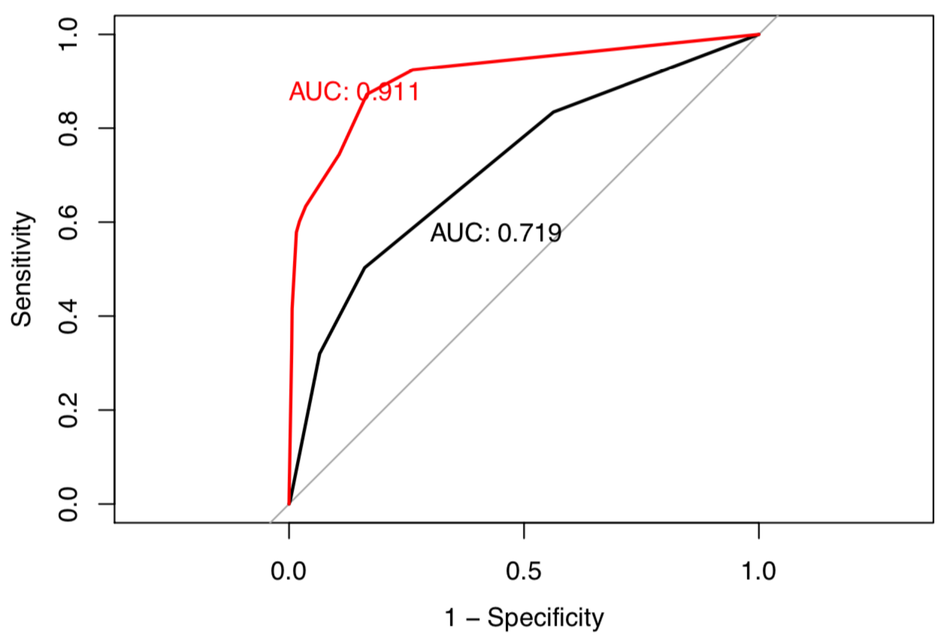
## 3.4 Tree

As a classifier tree, I make y as a factor in the model.

Through the summary of the tree, we find that some of the branches can be pruned.



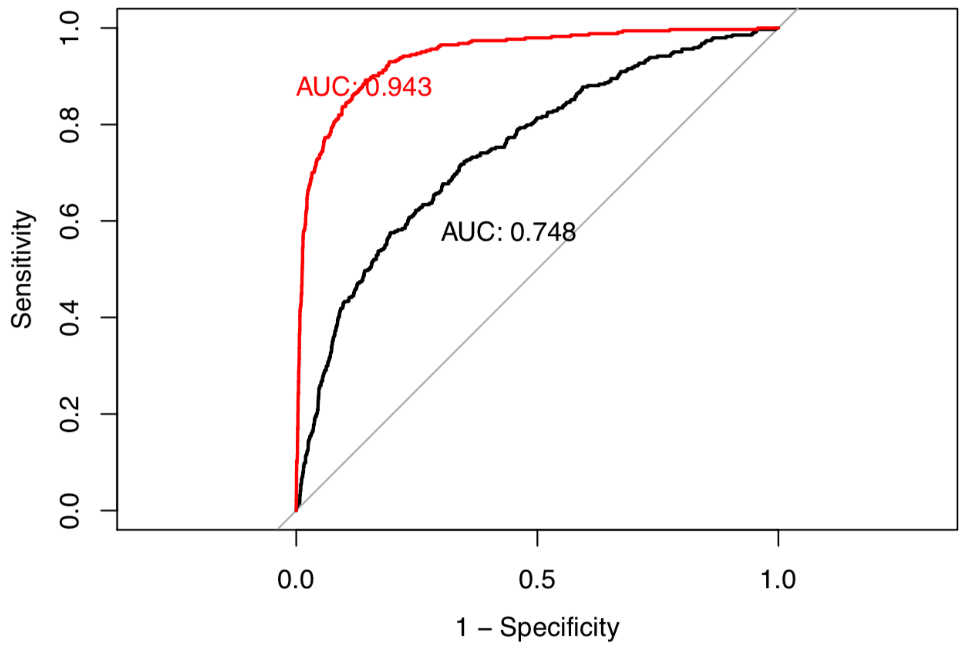

From the AUC plot, we find that the single tree has done a good job.



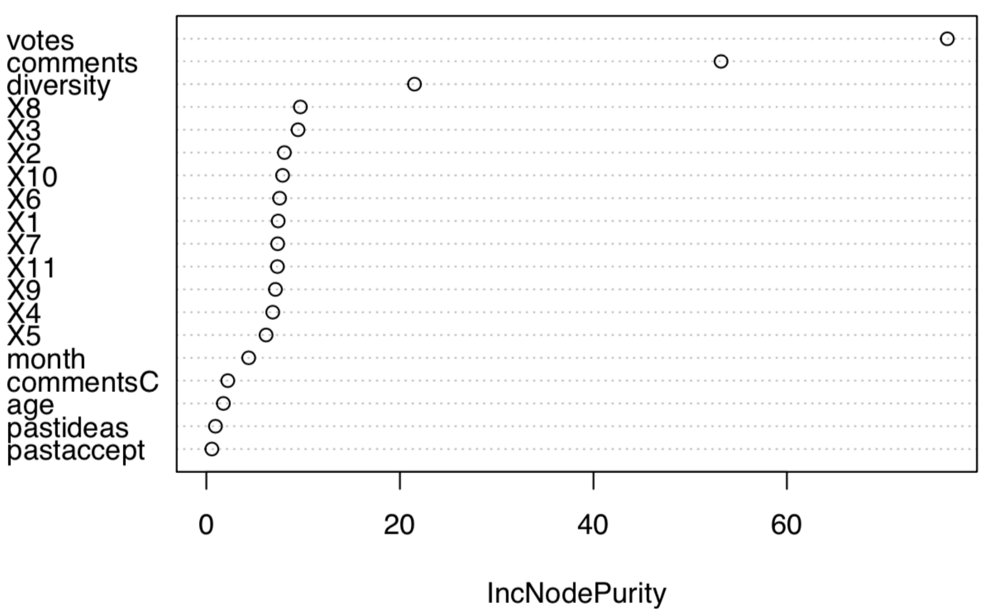
## 3.5 Random Forest

Then I tried the Random Forest. I have fit the Random forest with the number of tree 500, 1000, 5000, and 10000. Finally, find that 1000 has a good result in the shortest time.

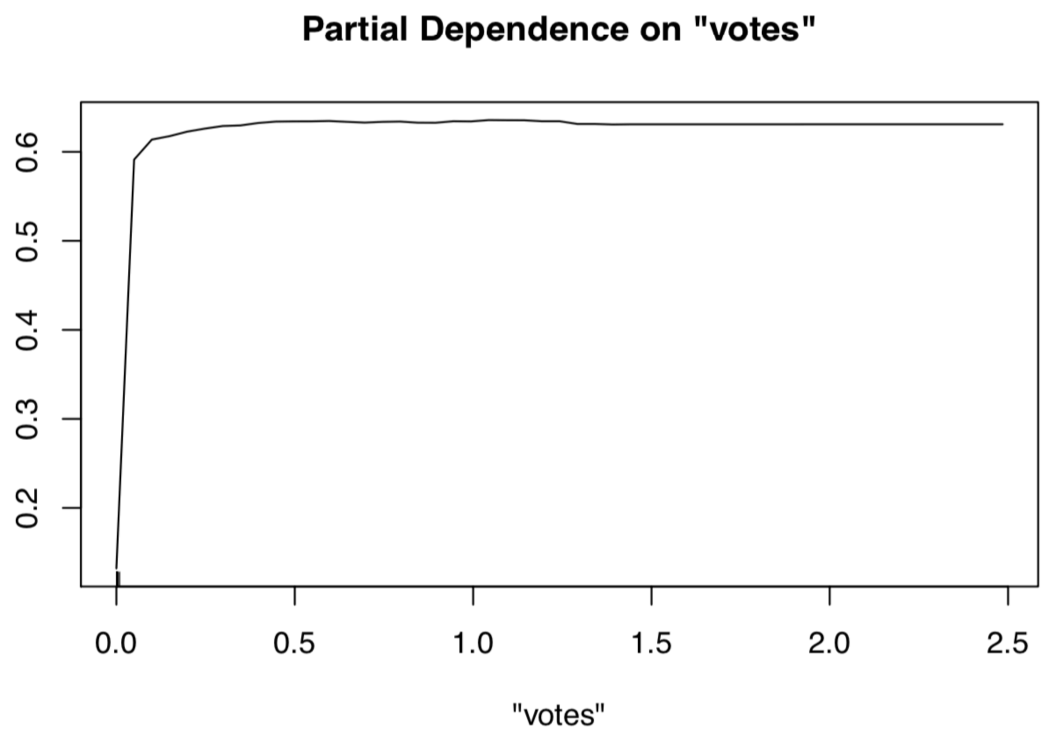
Random Forest has done a pretty good job with an AUC of 0.943.




We can also do former analysis with random forest. The importance of the predictors, for example, can be found from the model.



Then we can also do the partial dependence on certain variables. Take the most important predictors votes for instance.



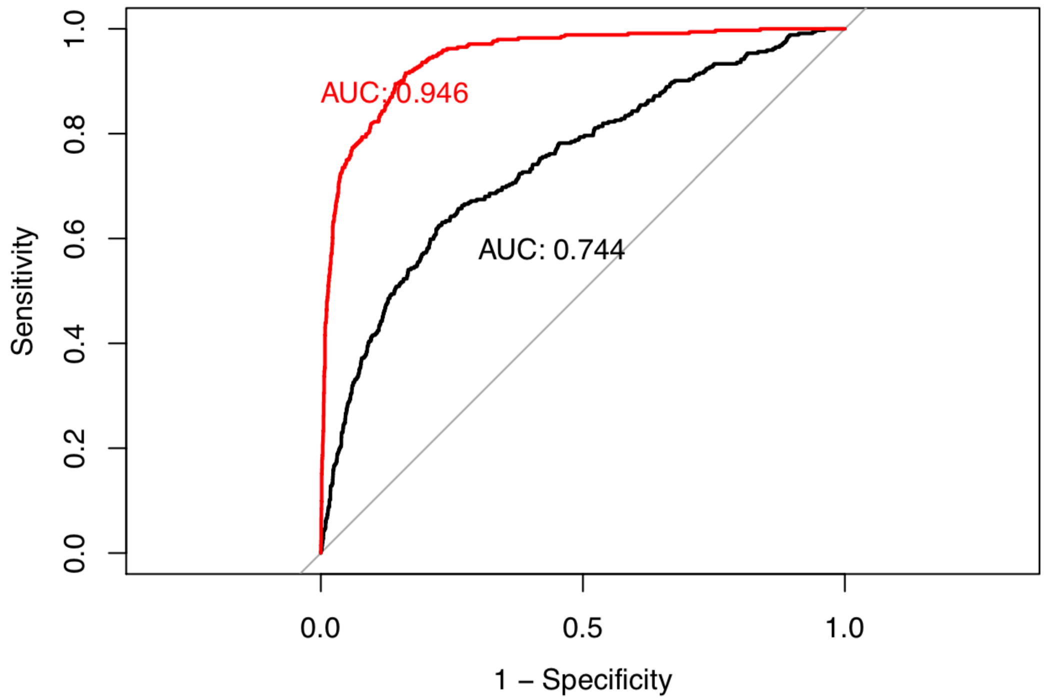

From the graphic, there seems to be a threshold around 0.1, we have to take care of the prediction with a votes number below 0.1, because it can have a huge influence.

Meanwhile, the comments predictor has a threshold at around 0.05, the diversity is stable between 0.1 and 0.88. The graphics are attached in the appendix.

## 3.6 GBM

When I try to model in GBM, I have tried some different number in certain arguments. Finally find that the interaction depth of 1, tree number of 500, with a shrinkage number of 0.02, are good enough for the boosted tree model.

Finally, I got the best AUC up until now, 0.946 when taking all the predictors into consideration.




When summary the GBM model, we find the same results when we analyze the random forest model that, the votes is the most important predictors. The three most important predictors are votes, comments, and diversity. As the summary below, the relevant influence of forth predictor is much smaller than the top 3.

```
##                    var     rel.inf
## votes            votes 55.04241635
## comments      comments 31.65762565
## diversity    diversity 10.51881399
## X8                  X8  0.86415074
## X3                  X3  0.74176020
```

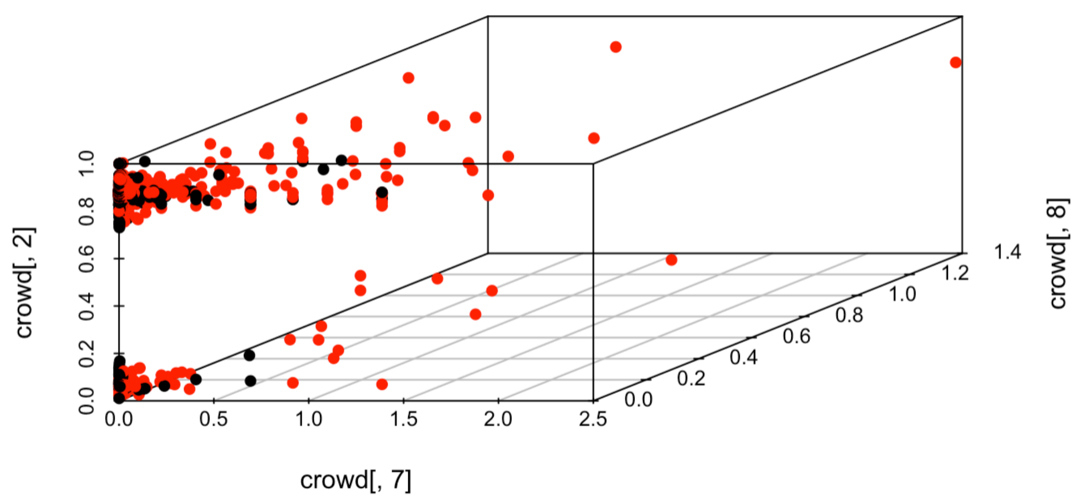
# Classifier Plot

In order to have an intuitive digraph, I tried PCA at first. However, according to the summary of PCA, I found that each eigenvector seems to explain a little proportion of the variance (the top 3 totally explain 0.3272 of variance only).

Calling back the correlation matrix mentioned in Section 2.1, there is little correlation among the predictors, PCA cannot help much.

```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6
Standard deviation     1.6280 1.4248 1.23950 1.14735 1.11179 1.10424
Proportion of Variance 0.1395 0.1069 0.08086 0.06929 0.06506 0.06418
Cumulative Proportion  0.1395 0.2463 0.32720 0.39649 0.46155 0.52572
                           PC7     PC8    PC9    PC10    PC11    PC12
Standard deviation     1.04668 1.00838 0.9853 0.92964 0.91303 0.85768
Proportion of Variance 0.05766 0.05352 0.0511 0.04549 0.04388 0.03872
Cumulative Proportion  0.58338 0.63690 0.6880 0.73348 0.77736 0.81608
                          PC13   PC14    PC15    PC16    PC17    PC18
Standard deviation     0.84562 0.8236 0.78023 0.75003 0.72515 0.52086
Proportion of Variance 0.03764 0.0357 0.03204 0.02961 0.02768 0.01428
Cumulative Proportion  0.85371 0.8894 0.92145 0.95106 0.97874 0.99302
                          PC19
Standard deviation     0.36426
Proportion of Variance 0.00698
Cumulative Proportion  1.00000
```

As mentioned in Section 3.6, the most important predictors, votes, comments, and diversity have high relevant influence. They can visualize the observations.




From the 3d scatterplot, votes and comments are useful predictors.

# Conclusion

1. Random Forest and boosted trees give a pretty good result. We have the best classifier with AUC of 0.946.
2. The most important predictors are votes (rel.inf=55.0424), comments (rel.inf=31.6576), and diversity(10.5188). While the fourth predictor (0.8642) has a long distance from them.
3. Votes and comments are important in classification.

# Mini Project

*Ray Liu*

*5/29/2019*

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
#pre analysis
setwd("~/Desktop/2019/NU/2019-spring/MSIT423/homework")
crowd = read.csv("crowd.csv")
summary(crowd) #seems like most of y is 0.
```

```
##      month          diversity         pastideas       pastaccept
##  Min.   : 1.000   Min.   :0.01042   Min.   :0.000   Min.   :0.00000
##  1st Qu.: 3.000   1st Qu.:0.86371   1st Qu.:0.000   1st Qu.:0.00000
##  Median : 7.000   Median :0.87719   Median :0.000   Median :0.00000
##  Mean   : 6.529   Mean   :0.78775   Mean   :0.149   Mean   :0.03853
##  3rd Qu.:10.000   3rd Qu.:0.88643   3rd Qu.:0.000   3rd Qu.:0.00000
##  Max.   :12.000   Max.   :1.00000   Max.   :2.565   Max.   :1.79176
##    commentsC            age              votes              comments
##  Min.   :0.00000   Min.   :0.0000   Min.   :0.000000   Min.   :0.000000
##  1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.001021   1st Qu.:0.000000
##  Median :0.00000   Median :0.0000   Median :0.002274   Median :0.000000
##  Mean   :0.05067   Mean   :0.6548   Mean   :0.018915   Mean   :0.004939
##  3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.004555   3rd Qu.:0.000746
##  Max.   :3.36730   Max.   :7.4012   Max.   :2.484907   Max.   :1.386294
##        X1                  X2                  X3
##  Min.   :-0.397150   Min.   :-0.891231   Min.   :-0.1789320
##  1st Qu.:-0.026658   1st Qu.: 0.002703   1st Qu.:-0.0092901
##  Median :-0.016004   Median : 0.007402   Median :-0.0027643
##  Mean   :-0.019848   Mean   : 0.008745   Mean   :-0.0043384
##  3rd Qu.:-0.007886   3rd Qu.: 0.012689   3rd Qu.: 0.0000657
##  Max.   : 0.000000   Max.   : 0.109753   Max.   : 0.8358010
##        X4                  X5                   X6
##  Min.   :-0.336066   Min.   :-0.5274983   Min.   :-0.4535512
##  1st Qu.:-0.003608   1st Qu.:-0.0051442   1st Qu.:-0.0065445
##  Median : 0.001401   Median :-0.0003849   Median :-0.0001907
##  Mean   : 0.002623   Mean   :-0.0014760   Mean   :-0.0012704
##  3rd Qu.: 0.009661   3rd Qu.: 0.0047312   3rd Qu.: 0.0043099
##  Max.   : 0.221676   Max.   : 0.2720763   Max.   : 0.4785086
##        X7                   X8                   X9
##  Min.   :-0.1995841   Min.   :-0.267181   Min.   :-0.246135
##  1st Qu.:-0.0074405   1st Qu.:-0.001583   1st Qu.:-0.011851
##  Median :-0.0004099   Median : 0.004073   Median :-0.002682
##  Mean   : 0.0009895   Mean   : 0.005727   Mean   :-0.003309
##  3rd Qu.: 0.0062817   3rd Qu.: 0.014108   3rd Qu.: 0.003402
##  Max.   : 0.3155084   Max.   : 0.315118   Max.   : 0.280809
```

```
##       X10                  X11                   y
##  Min.   :-0.3159175   Min.   :-0.286322   Min.   :0.00000
##  1st Qu.:-0.0078737   1st Qu.:-0.005880   1st Qu.:0.00000
##  Median :-0.0006237   Median : 0.001145   Median :0.00000
##  Mean   :-0.0011428   Mean   : 0.001821   Mean   :0.08941
##  3rd Qu.: 0.0058351   3rd Qu.: 0.009415   3rd Qu.:0.00000
##  Max.   : 0.2983781   Max.   : 0.303909   Max.   :1.00000
```

```
#train set
set.seed(12345)
train = runif(nrow(crowd))<.5
table(train)
```

```
## train
## FALSE  TRUE
##  3540  3506
```

```
addmargins(table(train,crowd$y))
```

```
##
## train      0    1  Sum
##   FALSE 3196  344 3540
##   TRUE  3220  286 3506
##   Sum   6416  630 7046
```

```
round(cor(crowd), 2)
```

```
##             month diversity pastideas pastaccept commentsC   age votes
## month        1.00      0.02     -0.01       0.01      0.03 -0.01  0.02
## diversity    0.02      1.00     -0.03      -0.02     -0.03 -0.04 -0.01
## pastideas   -0.01     -0.03      1.00       0.64      0.40  0.84  0.01
## pastaccept   0.01     -0.02      0.64       1.00      0.33  0.48  0.04
## commentsC    0.03     -0.03      0.40       0.33      1.00  0.44  0.03
## age         -0.01     -0.04      0.84       0.48      0.44  1.00  0.01
## votes        0.02     -0.01      0.01       0.04      0.03  0.01  1.00
## comments    -0.01     -0.05      0.03       0.05      0.08  0.03  0.42
## X1           0.03      0.42     -0.09      -0.06     -0.04 -0.10 -0.01
## X2          -0.02     -0.16      0.05       0.05      0.03  0.06 -0.04
## X3          -0.01     -0.01     -0.03      -0.03     -0.01 -0.02  0.04
## X4          -0.01     -0.39     -0.01      -0.04      0.00  0.00  0.03
## X5           0.02      0.00      0.00       0.04      0.01  0.00  0.01
## X6           0.00      0.07      0.02       0.03      0.00  0.01  0.00
## X7           0.00     -0.10     -0.02       0.01      0.05  0.00  0.04
## X8           0.02     -0.23     -0.02      -0.02     -0.02 -0.02 -0.05
## X9           0.00     -0.46      0.00       0.02      0.02  0.00 -0.01
## X10         -0.01      0.27      0.01      -0.02     -0.01  0.00  0.01
## X11         -0.01      0.20     -0.03      -0.04     -0.02 -0.03 -0.03
## y           -0.01     -0.02      0.03       0.06      0.09  0.03  0.33
##             comments    X1    X2    X3    X4   X5   X6    X7    X8    X9
## month          -0.01  0.03 -0.02 -0.01 -0.01 0.02 0.00  0.00  0.02  0.00
## diversity      -0.05  0.42 -0.16 -0.01 -0.39 0.00 0.07 -0.10 -0.23 -0.46
## pastideas       0.03 -0.09  0.05 -0.03 -0.01 0.00 0.02 -0.02 -0.02  0.00
## pastaccept      0.05 -0.06  0.05 -0.03 -0.04 0.04 0.03  0.01 -0.02  0.02
## commentsC       0.08 -0.04  0.03 -0.01  0.00 0.01 0.00  0.05 -0.02  0.02
## age             0.03 -0.10  0.06 -0.02  0.00 0.00 0.01  0.00 -0.02  0.00
## votes           0.42 -0.01 -0.04  0.04  0.03 0.01 0.00  0.04 -0.05 -0.01
```

```
## comments       1.00 -0.03 -0.02  0.04  0.05  0.02  0.00  0.07 -0.08  0.03
## X1            -0.03  1.00 -0.29  0.06 -0.06  0.03  0.04  0.01 -0.10 -0.08
## X2            -0.02 -0.29  1.00 -0.07  0.02 -0.01 -0.01 -0.04  0.09  0.00
## X3             0.04  0.06 -0.07  1.00  0.25  0.00 -0.03  0.07 -0.14 -0.05
## X4             0.05 -0.06  0.02  0.25  1.00  0.05  0.08 -0.11  0.01  0.11
## X5             0.02  0.03 -0.01  0.00  0.05  1.00 -0.05  0.12  0.10 -0.02
## X6             0.00  0.04 -0.01 -0.03  0.08 -0.05  1.00 -0.10  0.05  0.03
## X7             0.07  0.01 -0.04  0.07 -0.11  0.12 -0.10  1.00  0.08  0.02
## X8            -0.08 -0.10  0.09 -0.14  0.01  0.10  0.05  0.08  1.00  0.01
## X9             0.03 -0.08  0.00 -0.05  0.11 -0.02  0.03  0.02  0.01  1.00
## X10            0.00  0.00  0.00  0.01 -0.09 -0.02 -0.06 -0.01  0.02 -0.19
## X11           -0.04  0.04  0.02 -0.09 -0.10  0.10  0.11 -0.07 -0.01  0.02
## y              0.27 -0.03 -0.04  0.09  0.05  0.03 -0.01  0.06 -0.09  0.01
##              X10   X11     y
## month      -0.01 -0.01 -0.01
## diversity   0.27  0.20 -0.02
## pastideas   0.01 -0.03  0.03
## pastaccept -0.02 -0.04  0.06
## commentsC  -0.01 -0.02  0.09
## age         0.00 -0.03  0.03
## votes       0.01 -0.03  0.33
## comments    0.00 -0.04  0.27
## X1          0.00  0.04 -0.03
## X2          0.00  0.02 -0.04
## X3          0.01 -0.09  0.09
## X4         -0.09 -0.10  0.05
## X5         -0.02  0.10  0.03
## X6         -0.06  0.11 -0.01
## X7         -0.01 -0.07  0.06
## X8          0.02 -0.01 -0.09
## X9         -0.19  0.02  0.01
## X10         1.00 -0.02 -0.02
## X11        -0.02  1.00 -0.05
## y          -0.02 -0.05  1.00
```

```r
#strong correlation between age and pastideas, while other variables is not highly related with each ot
```

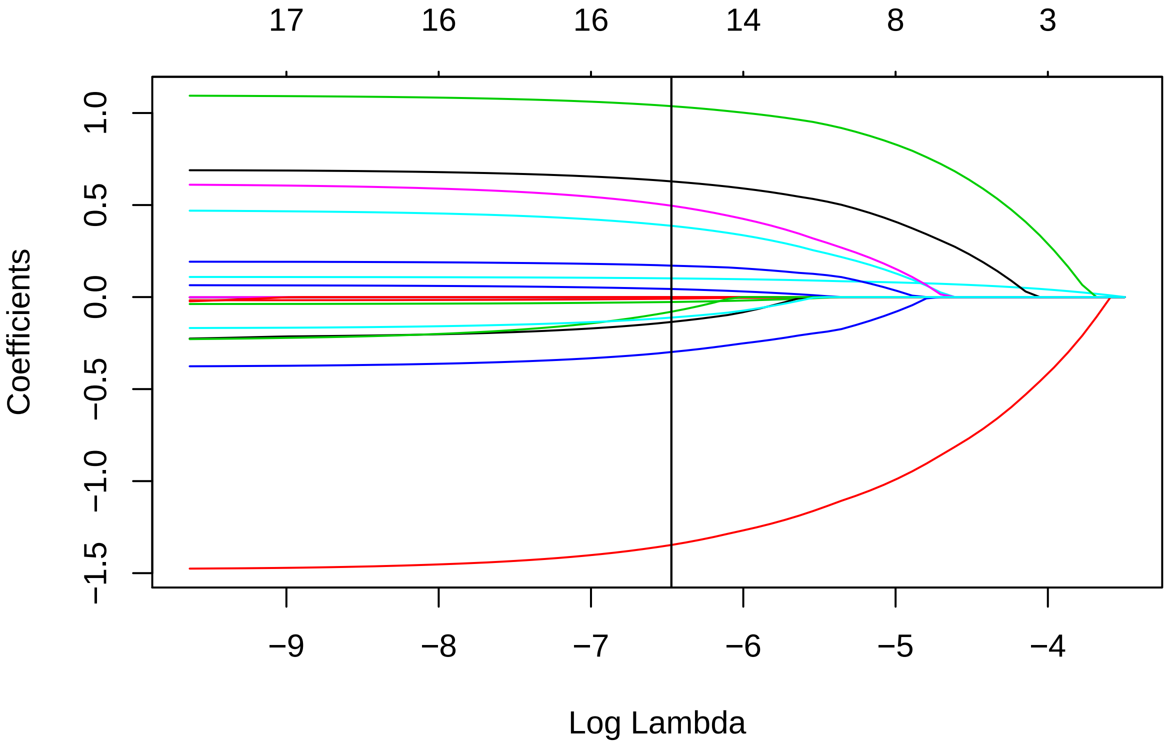
```r
#Try Lasso
#lasso(conributer+content)/all variables
library(glmnet)
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-18
```

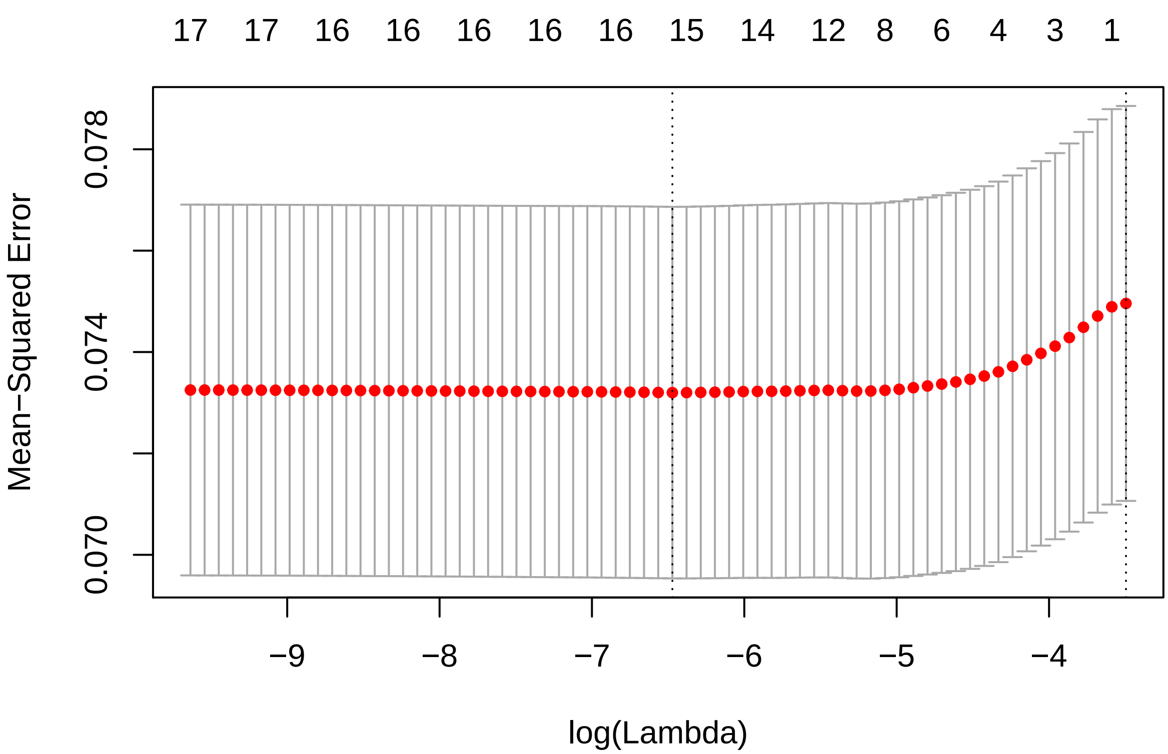
```r
x = model.matrix(y ~ month+diversity+pastideas+pastaccept+commentsC+age+X1+X2
                 +X3+X4+X5+X6+X7+X8+X9+X10+X11, crowd)
fit.lasso = glmnet(x[train,], crowd$y[train], alpha=1)
plot(fit.lasso, xvar="lambda")
fit.cv = cv.glmnet(x[train,], crowd$y[train], alpha=1) # find optimal lambda
abline(v=log(fit.cv$lambda.min))
fit.cv$lambda.min        # optimal value of lambda
```

```
## [1] 0.00154603
```

```r
abline(v=log(fit.cv$lambda.min))
```




```r
plot(fit.cv)             # plot MSE vs. log(lambda)
```




```r
predict(fit.lasso, s=fit.cv$lambda.min, newx=x[!train,], type = "coef")
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
```

```
##                          1
## (Intercept)  1.014665e-01
## (Intercept)  .
## month       -1.046356e-03
## diversity   -8.873627e-03
## pastideas   -2.703135e-02
## pastaccept   4.383091e-02
## commentsC    1.015161e-01
## age         -1.214399e-05
## X1          -1.351786e-01
## X2           .
## X3           1.037145e+00
## X4           1.706700e-01
## X5           3.871876e-01
## X6           4.964316e-01
## X7           6.284911e-01
## X8          -1.346778e+00
## X9          -7.973079e-02
## X10         -2.984569e-01
## X11         -1.114583e-01
```

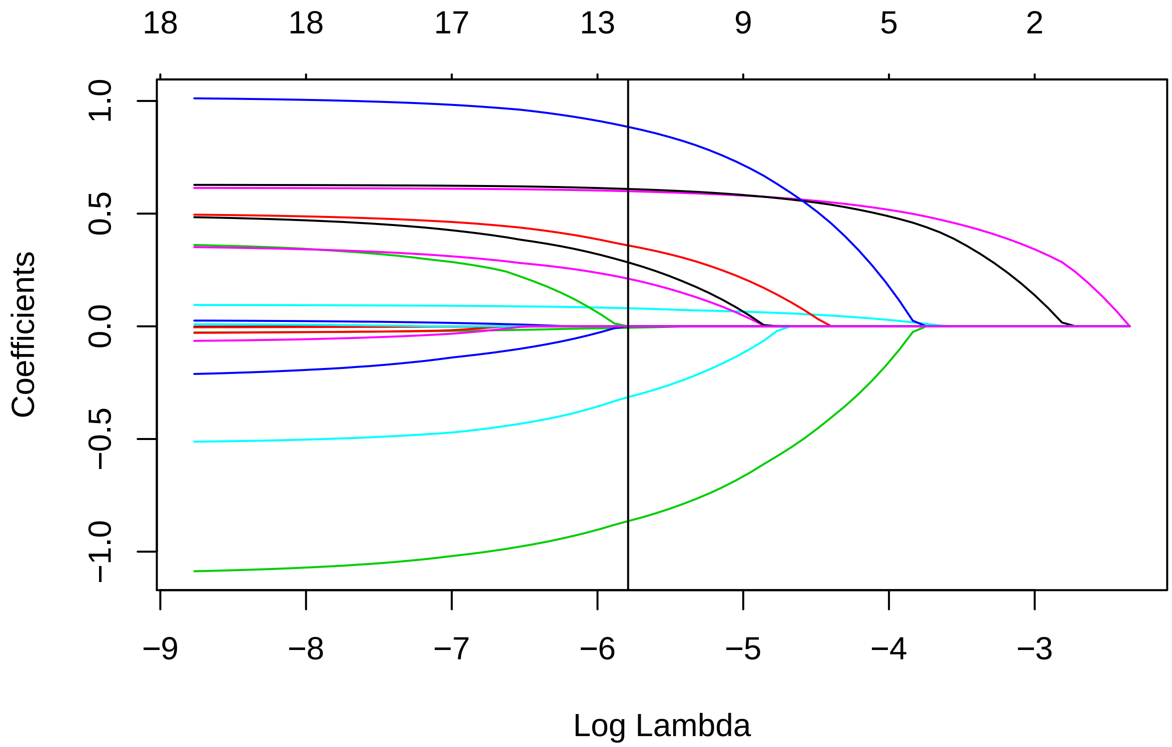
```
yhat = predict(fit.lasso, s=fit.cv$lambda.min, newx=x[!train,])  # find yhat for best model
mean((crowd$y[!train] - yhat)^2)      # compute test set MSE
```

```
## [1] 0.08638104
```

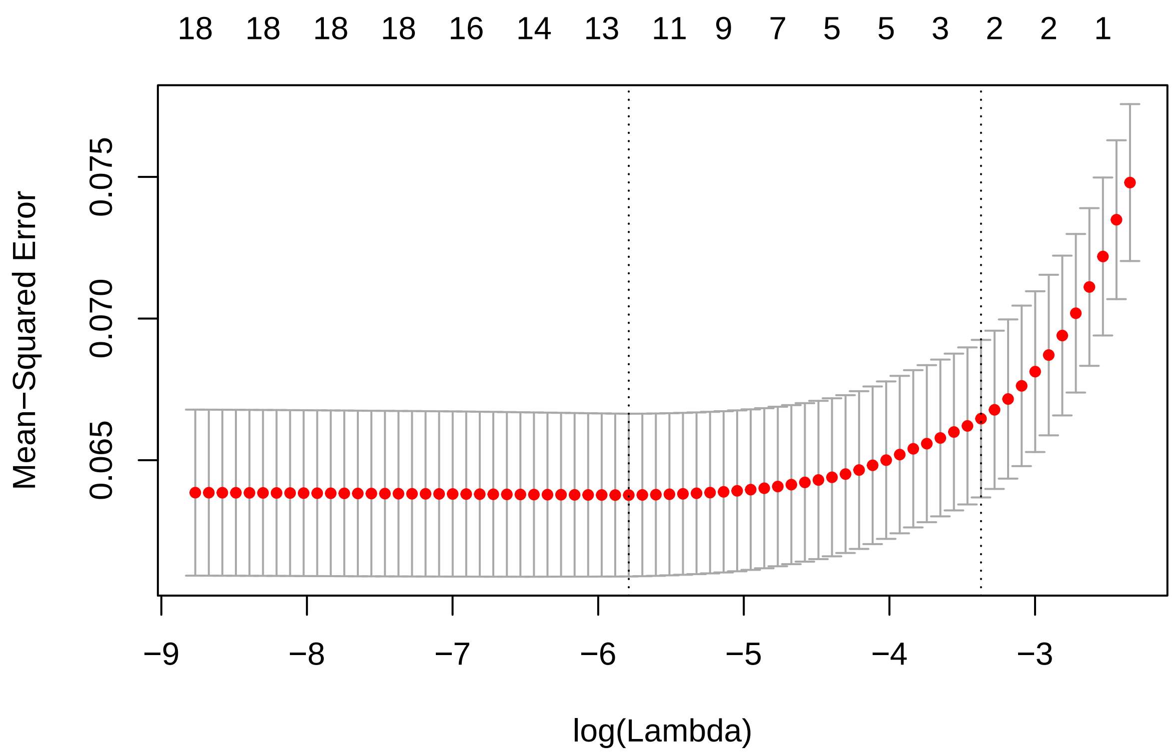
```
#all variables
xa = model.matrix(y ~ ., crowd)
fit.lassoa = glmnet(xa[train,], crowd$y[train], alpha=1)
plot(fit.lassoa, xvar="lambda")
fit.cva = cv.glmnet(xa[train,], crowd$y[train], alpha=1) # find optimal lambda
fit.cva$lambda.min        # optimal value of lambda
```

```
## [1] 0.003058272
```

```
abline(v=log(fit.cva$lambda.min))
```




```r
plot(fit.cva)           # plot MSE vs. log(lambda)
```



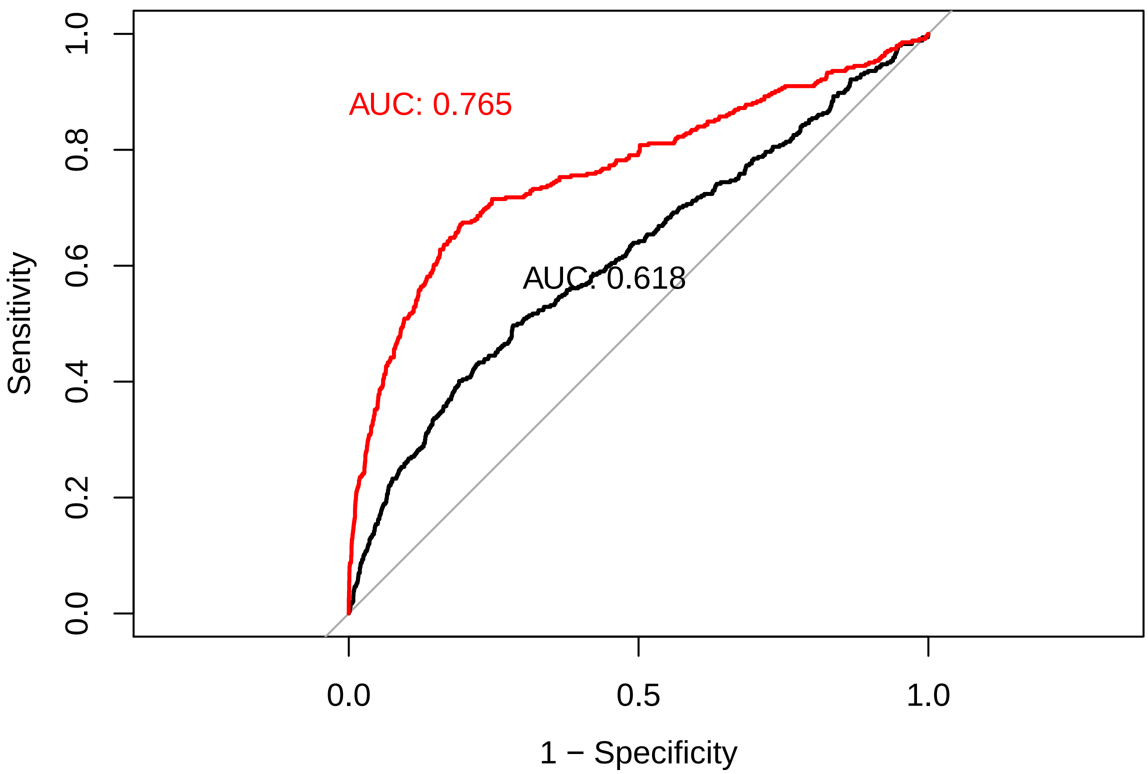

```r
yhata = predict(fit.lassoa, s=fit.cva$lambda.min, newx=xa[!train,])  # find yhat for best model
mean((crowd$y[!train] - yhata)^2)       # compute test set MSE
```

```
## [1] 0.07666033
```

```r
#AUC
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following object is masked from 'package:glmnet':
##
##     auc

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

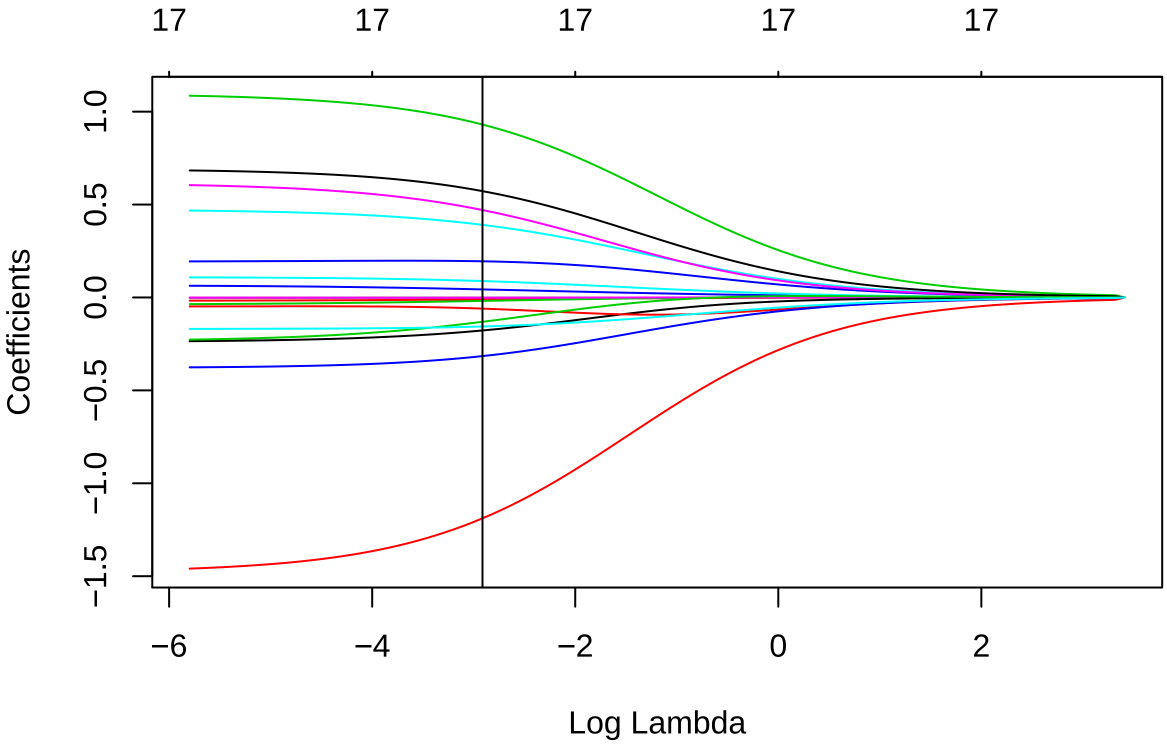
```r
plot.roc(crowd$y[!train], as.vector(yhat), legacy.axes=T,
         print.auc=T, print.auc.x=.7, print.auc.y=.6)
plot.roc(crowd$y[!train], as.vector(yhata), add=T, col=2,
         print.auc=T, print.auc.x=1, print.auc.y=.9, print.auc.col=2)
```



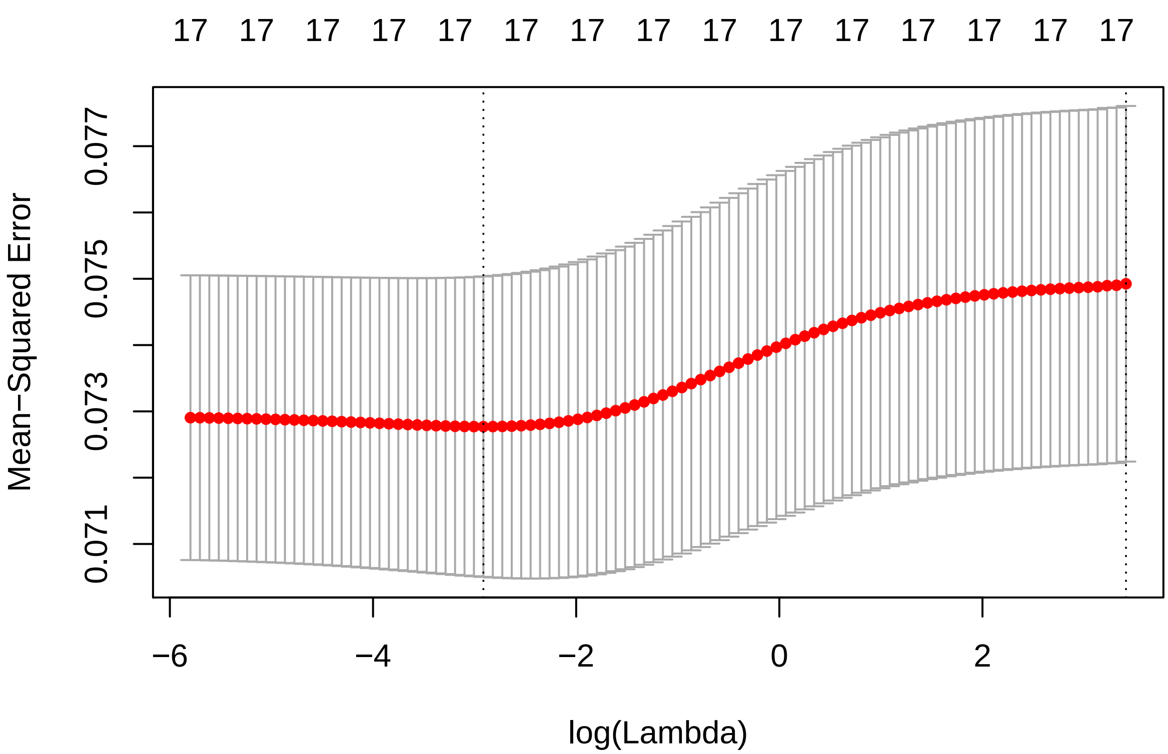

```r
#ridge(conributer+content)/all variables
library(glmnet)
x = model.matrix(y ~ month+diversity+pastideas+pastaccept+commentsC+age+X1+X2
                 +X3+X4+X5+X6+X7+X8+X9+X10+X11, crowd)
fit.lasso = glmnet(x[train,], crowd$y[train], alpha=0)
plot(fit.lasso, xvar="lambda")
fit.cv = cv.glmnet(x[train,], crowd$y[train], alpha=0) # find optimal lambda
fit.cv$lambda.min         # optimal value of lambda
```

```
## [1] 0.05428407
```

```r
abline(v=log(fit.cv$lambda.min))
```




```r
plot(fit.cv)           # plot MSE vs. log(lambda)
```



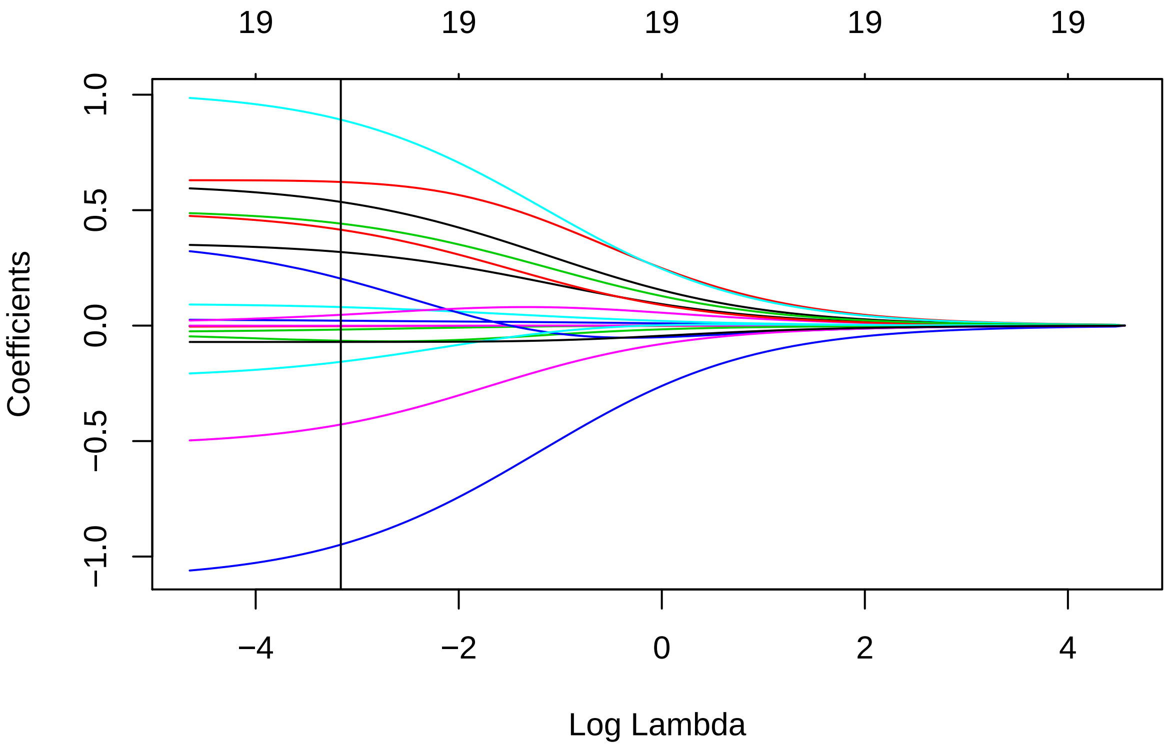

```r
yhat = predict(fit.lasso, s=fit.cv$lambda.min, newx=x[!train,])  # find yhat for best model
mean((crowd$y[!train] - yhat)^2)      # compute test set MSE
```

```
## [1] 0.08618138
```

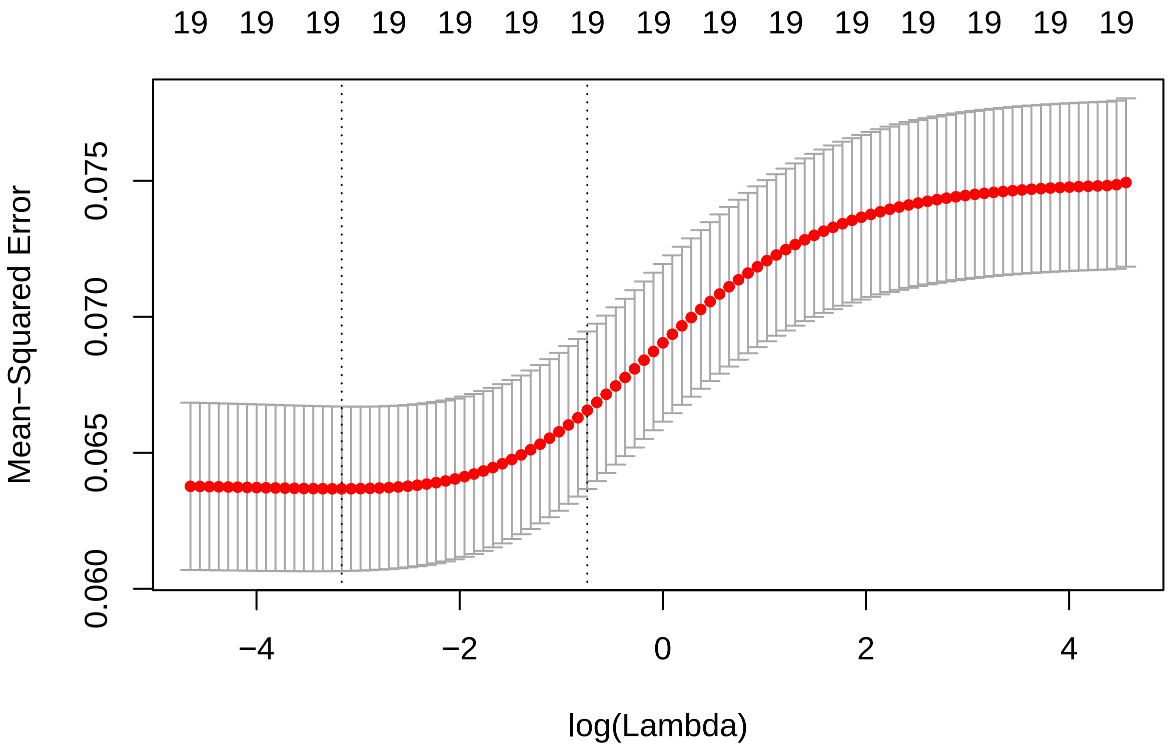
```r
#all variables
xa = model.matrix(y ~ ., crowd)
fit.lassoa = glmnet(xa[train,], crowd$y[train], alpha=0)
plot(fit.lassoa, xvar="lambda")
fit.cva = cv.glmnet(xa[train,], crowd$y[train], alpha=0) # find optimal lambda
fit.cva$lambda.min          # optimal value of lambda
```

```
## [1] 0.04235359
```

```r
abline(v=log(fit.cva$lambda.min))
```



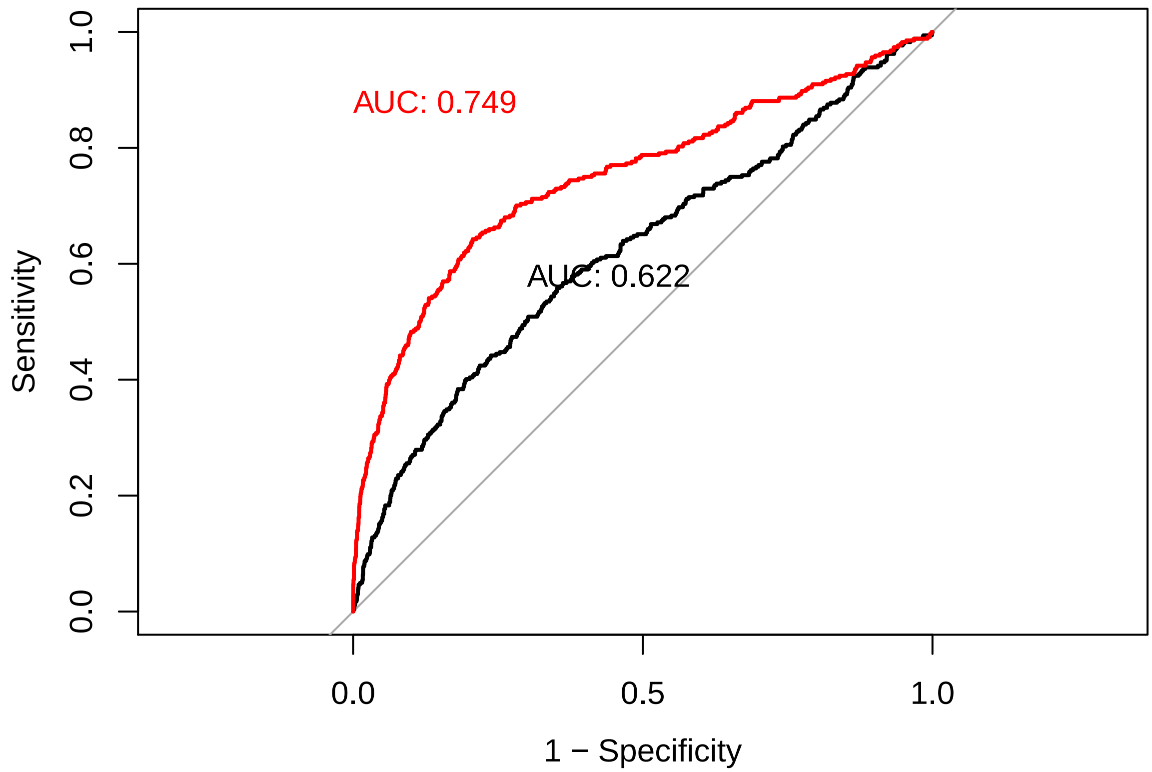

```r
plot(fit.cva)          # plot MSE vs. log(lambda)
```




```
yhata = predict(fit.lassoa, s=fit.cva$lambda.min, newx=xa[!train,])  # find yhat for best model
mean((crowd$y[!train] - yhata)^2)       # compute test set MSE
```

```
## [1] 0.07706735
```

```
#AUC
library(pROC)
plot.roc(crowd$y[!train], as.vector(yhat), legacy.axes=T,
         print.auc=T, print.auc.x=.7, print.auc.y=.6)
plot.roc(crowd$y[!train], as.vector(yhata), add=T, col=2,
         print.auc=T, print.auc.x=1, print.auc.y=.9, print.auc.col=2)
```




```r
#GAM / all
library(gam)
```

```
## Loading required package: splines
```

```
## Loaded gam 1.16
```

```r
fit.gam=gam(y ~ month+diversity+pastideas+pastaccept+commentsC+age+X1+X2
            +X3+X4+X5+X6+X7+X8+X9+X10+X11, binomial, data=crowd[train,])
```

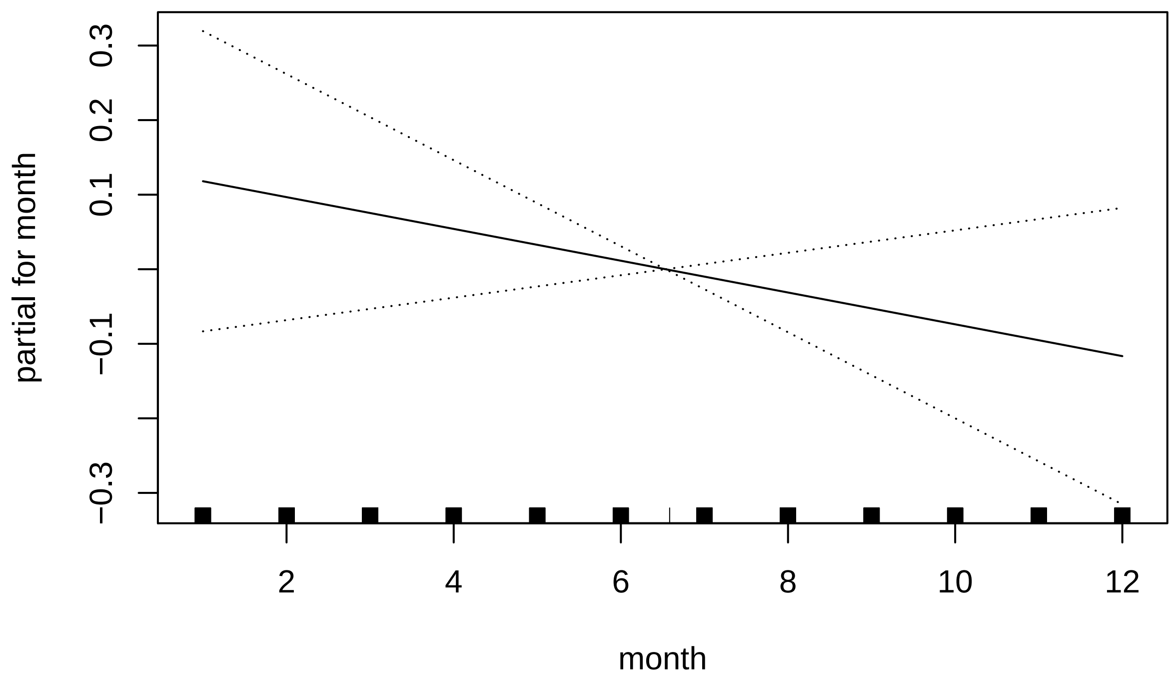
```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

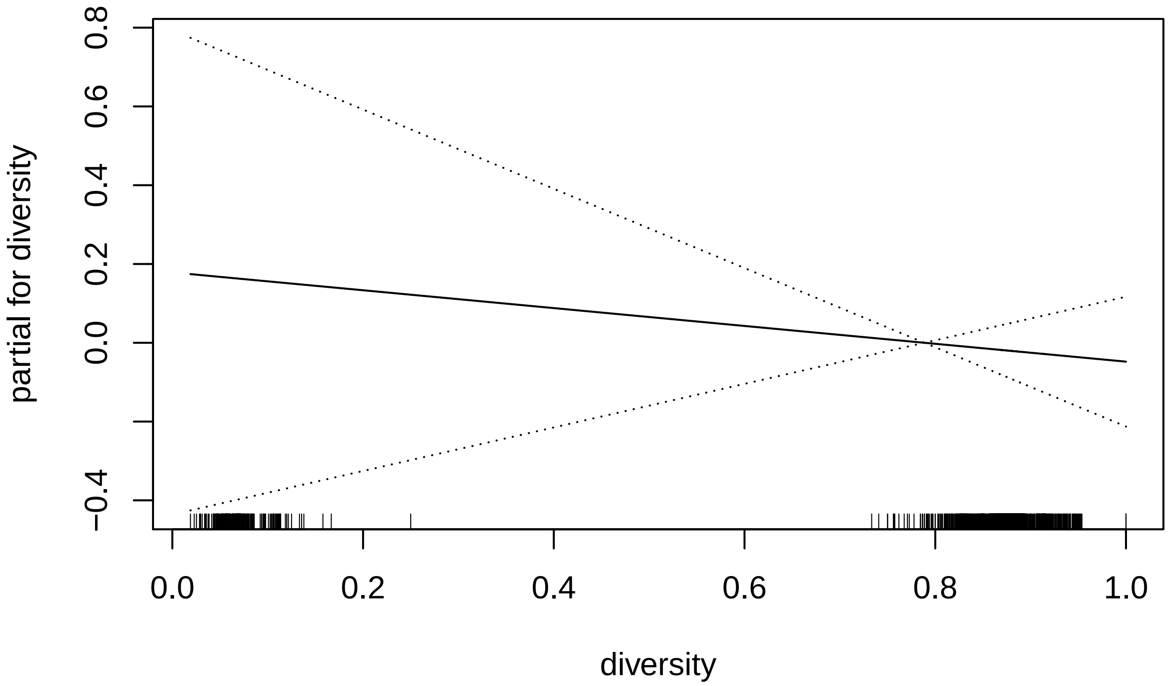
```r
summary(fit.gam)
```

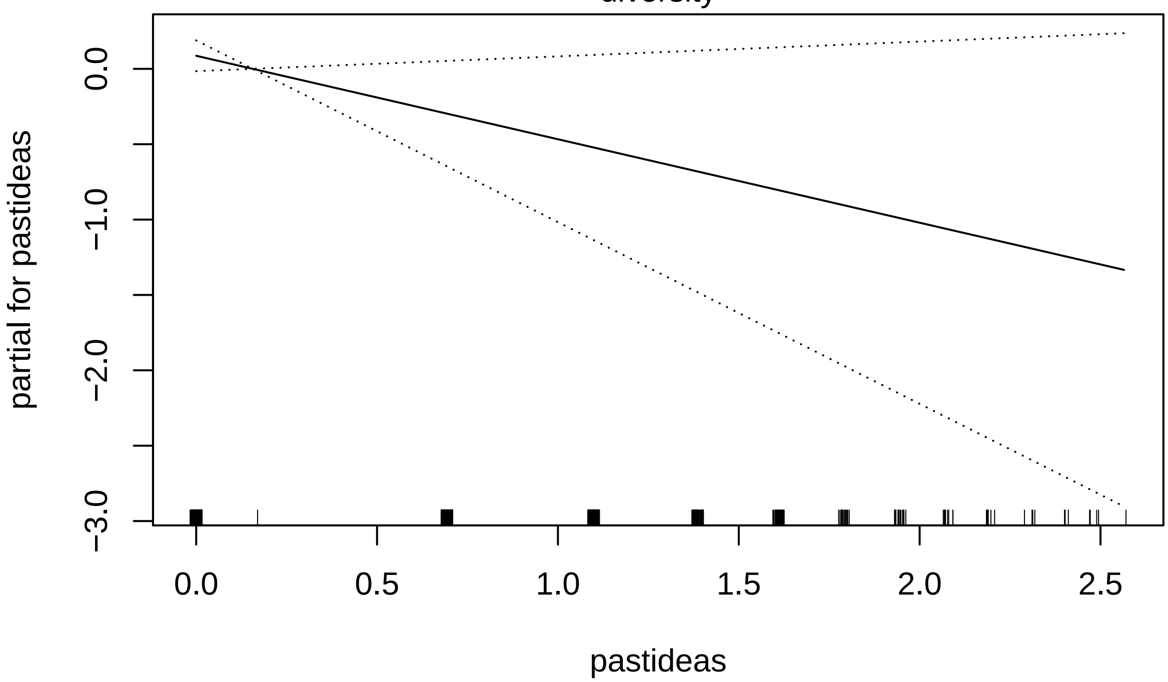
```
##
## Call: gam(formula = y ~ month + diversity + pastideas + pastaccept +
##     commentsC + age + X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 +
##     X9 + X10 + X11, family = binomial, data = crowd[train, ])
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1365 -0.4236 -0.3847 -0.3260  2.8197
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 1981.577 on 3505 degrees of freedom
## Residual Deviance: 1877.084 on 3488 degrees of freedom
## AIC: 1913.084
##
## Number of Local Scoring Iterations: 5
##
```

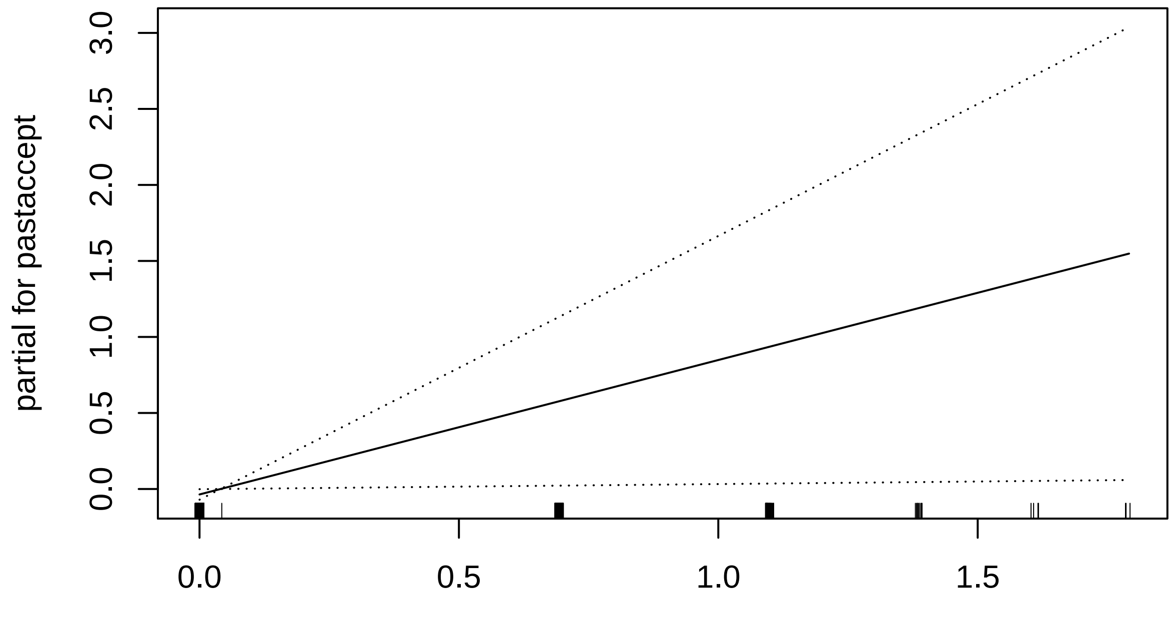
```
## Anova for Parametric Effects
##                Df Sum Sq Mean Sq F value    Pr(>F)    
## month           1    0.8  0.8320  0.8208 0.3650114    
## diversity       1    1.5  1.5383  1.5176 0.2180601    
## pastideas       1    4.7  4.7073  4.6440 0.0312313 *  
## pastaccept      1    4.8  4.8285  4.7635 0.0291353 *  
## commentsC       1   28.3 28.2684 27.8881 1.364e-07 ***
## age             1    0.1  0.0695  0.0685 0.7934824    
## X1              1    0.3  0.2737  0.2700 0.6033823    
## X2              1    3.3  3.2805  3.2363 0.0721085 .  
## X3              1   13.5 13.4867 13.3053 0.0002685 ***
## X4              1    5.7  5.7362  5.6590 0.0174194 *  
## X5              1    6.6  6.6022  6.5134 0.0107486 *  
## X6              1    2.4  2.4140  2.3815 0.1228683    
## X7              1    3.4  3.3690  3.3237 0.0683736 .  
## X8              1   23.8 23.7660 23.4463 1.340e-06 ***
## X9              1    0.4  0.3978  0.3925 0.5310447    
## X10             1    0.1  0.0922  0.0910 0.7629592    
## X11             1    0.7  0.6917  0.6824 0.4088032    
## Residuals    3488 3535.6  1.0136                      
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

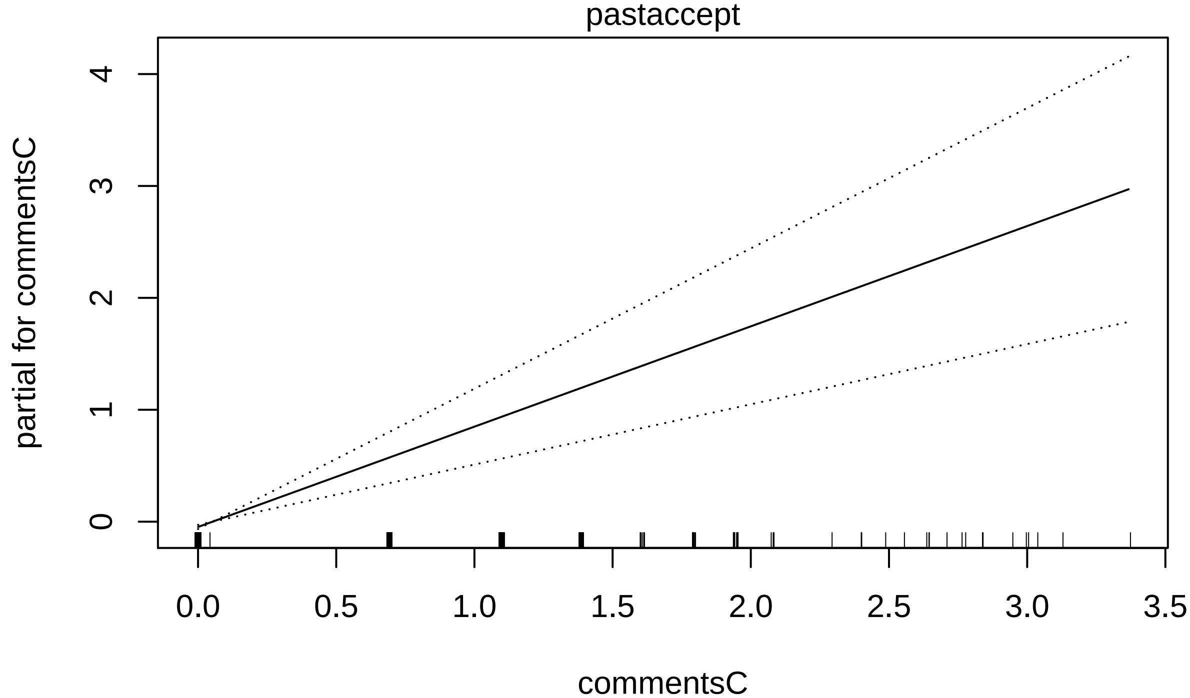
```r
plot(fit.gam, se=T)
```

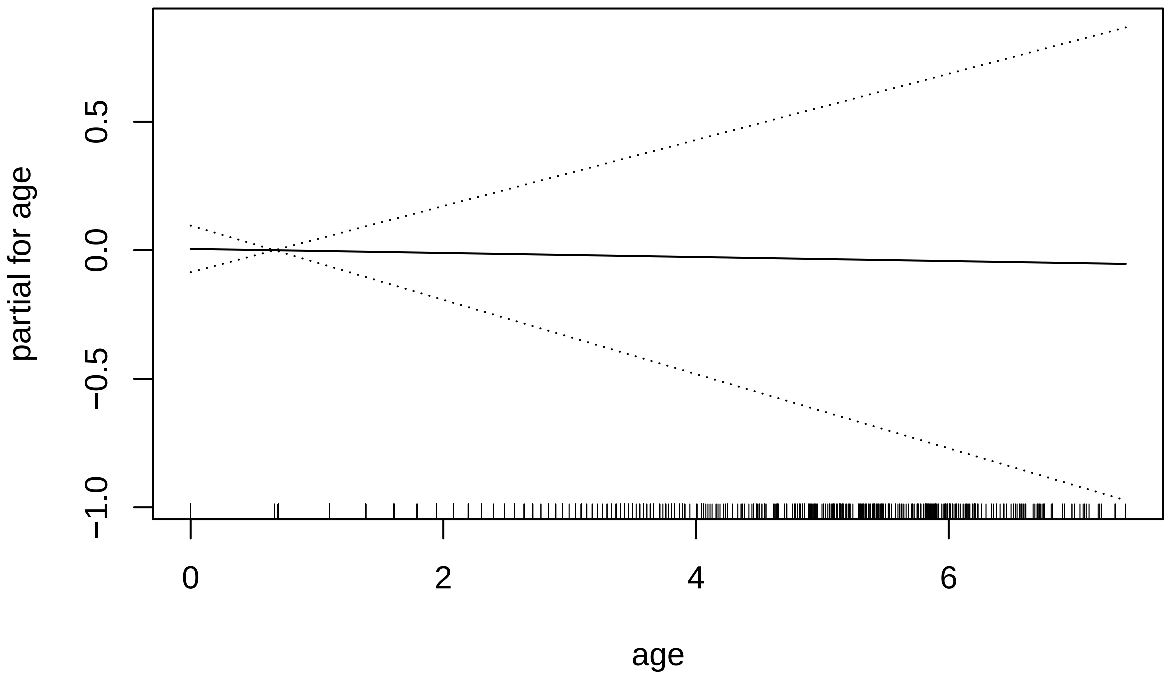


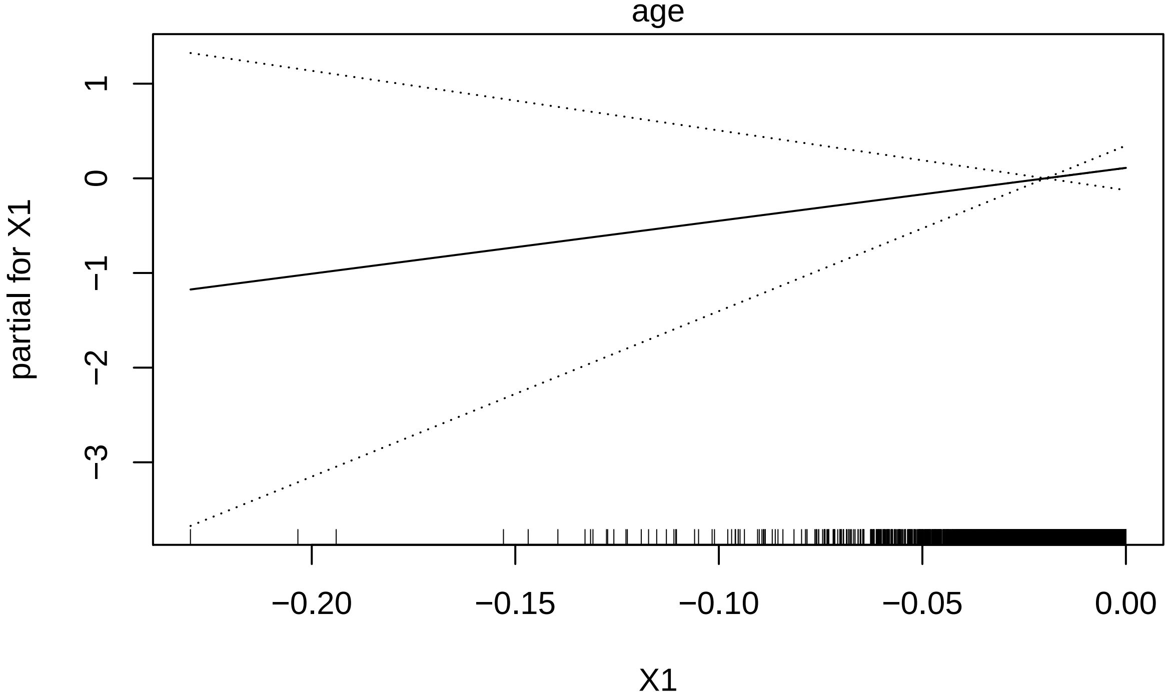

0.8
0.6
0.4
0.2
0.0
−0.4
partial for diversity
0.0
0.2
0.4
0.6
0.8
1.0
diversity

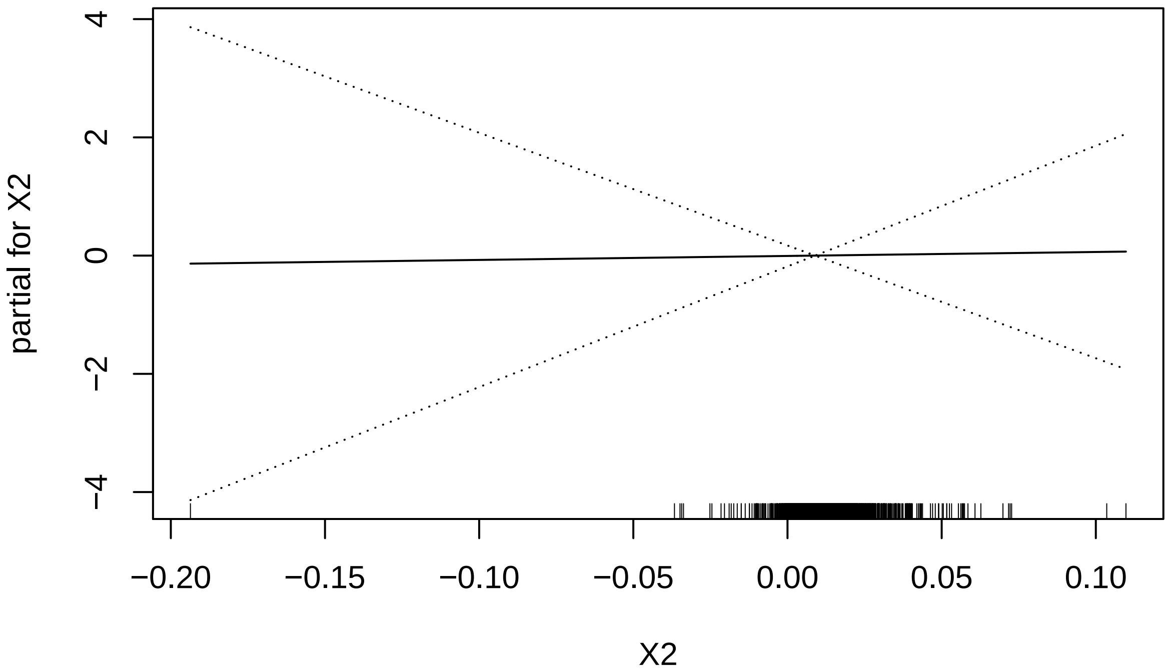

0.0
−1.0
−2.0
−3.0
partial for pastideas
0.0
0.5
1.0
1.5
2.0
2.5
pastideas


partial for pastaccept
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.0
0.5
1.0
1.5
pastaccept

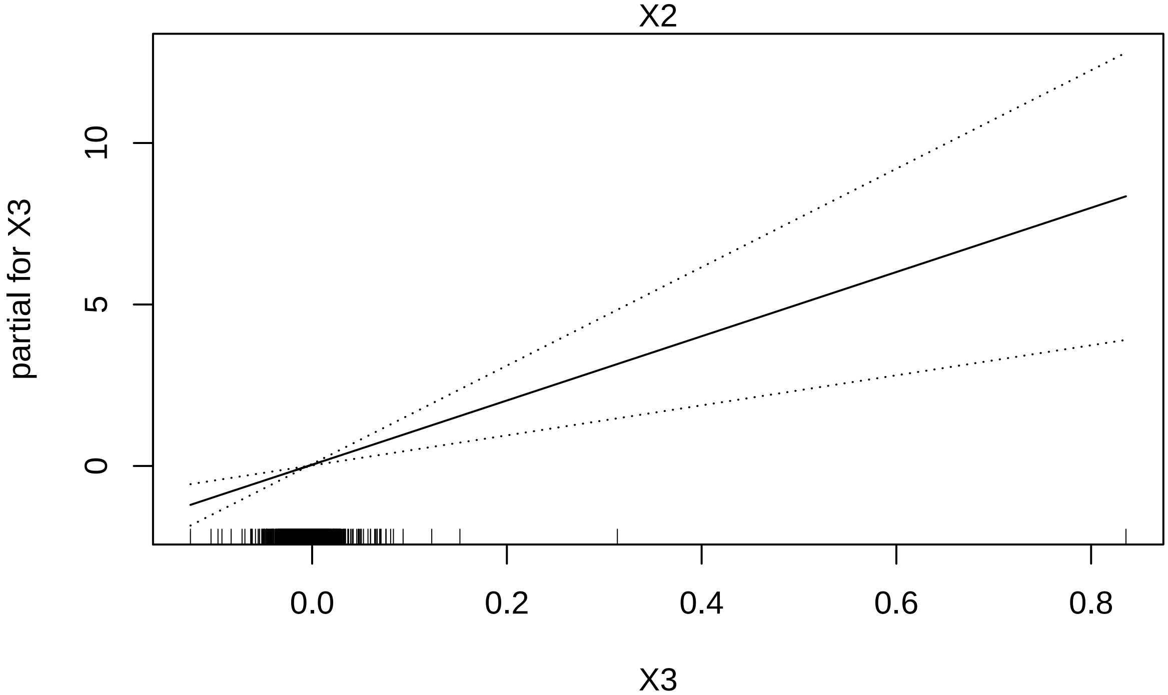

partial for commentsC
0
1
2
3
4
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
commentsC


partial for age
0.5
0.0
−0.5
−1.0
0
2
4
6
age

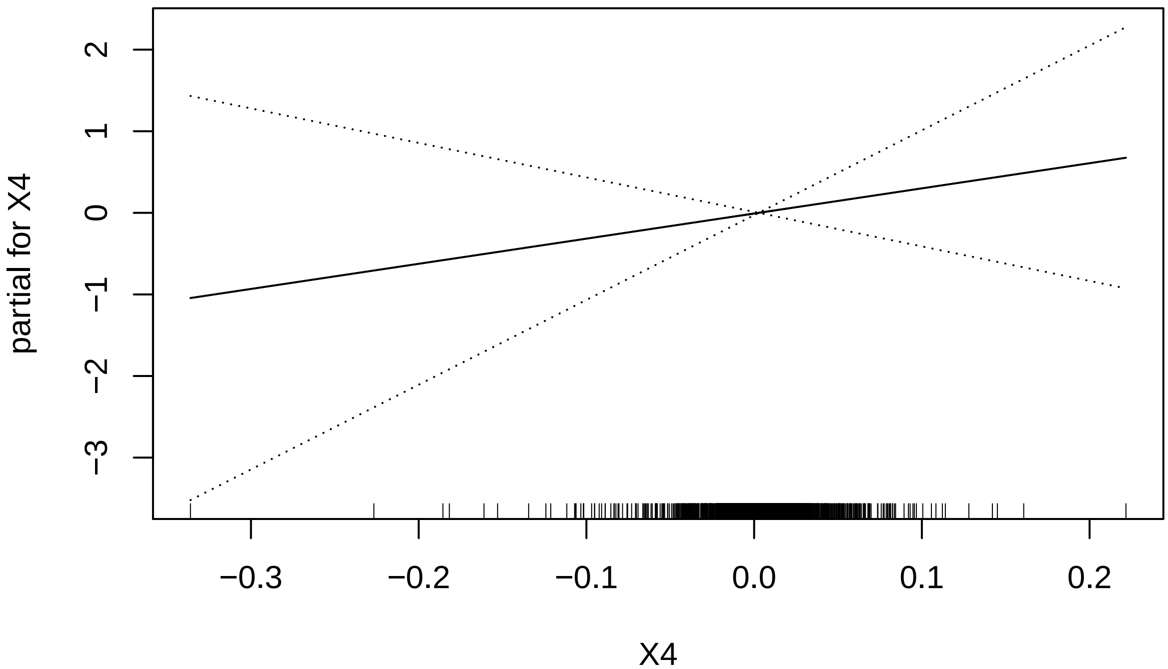

partial for X1
1
0
−1
−2
−3
−0.20
−0.15
−0.10
−0.05
0.00
X1



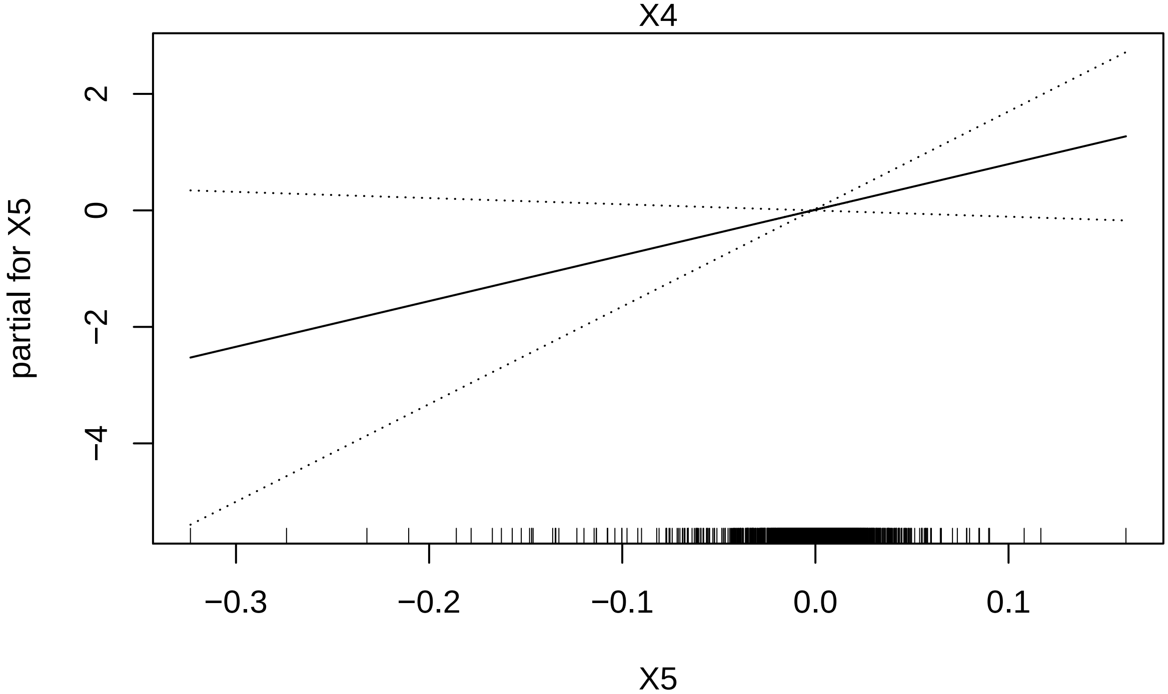
4
2
0
−2
−4
partial for X2
−0.20
−0.15
−0.10
−0.05
0.00
0.05
0.10
X2

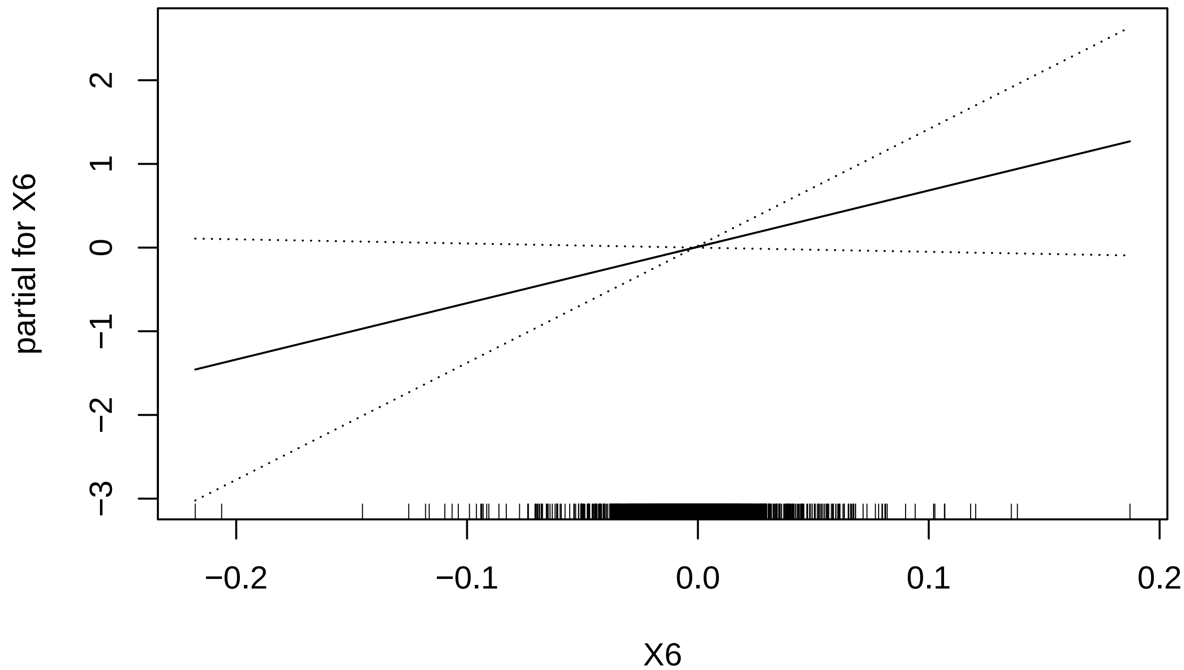
10
5
0
partial for X3
0.0
0.2
0.4
0.6
0.8
X3

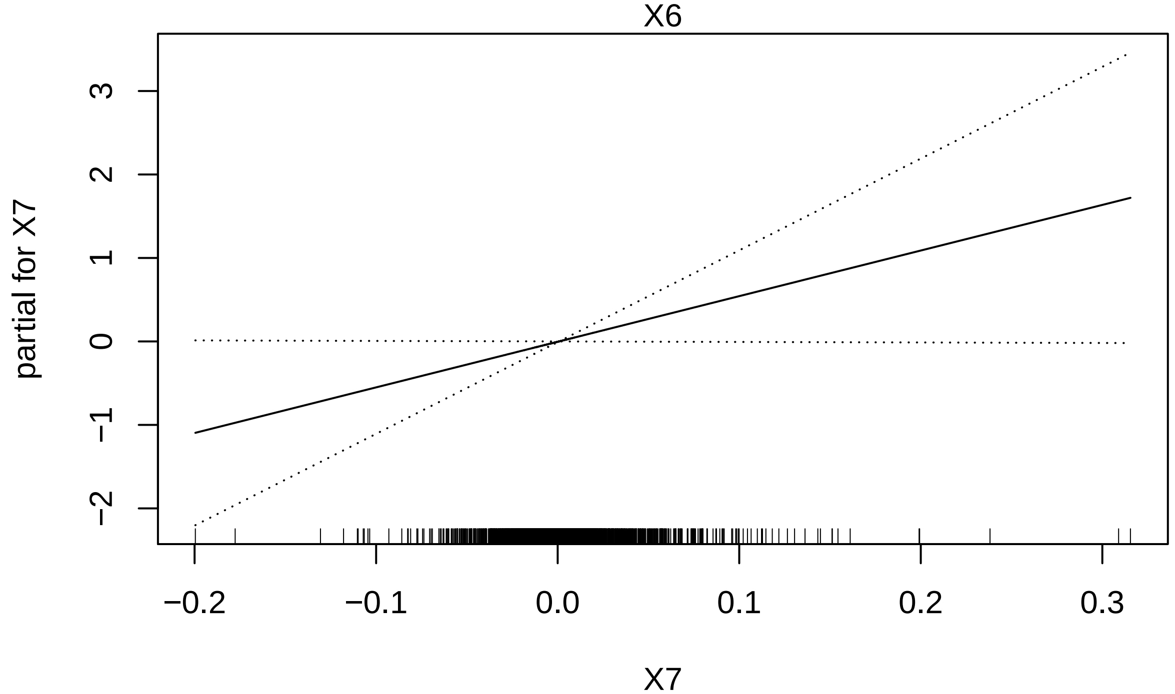


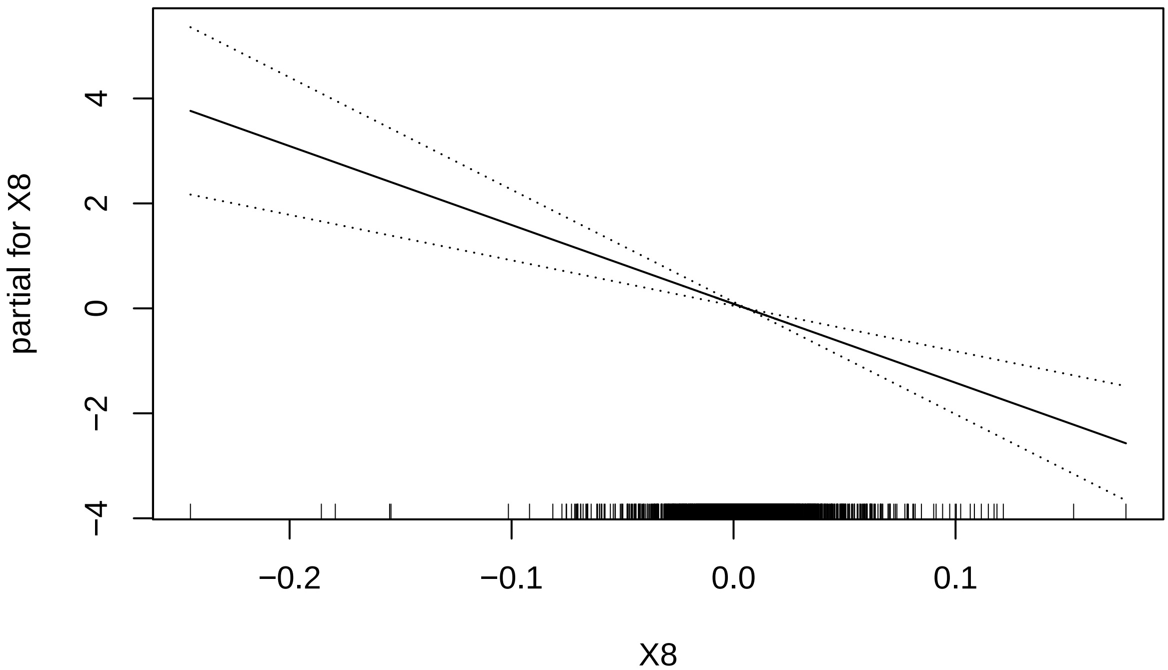
2
1
0
−1
−2
−3
partial for X4
−0.3
−0.2
−0.1
0.0
0.1
0.2
X4

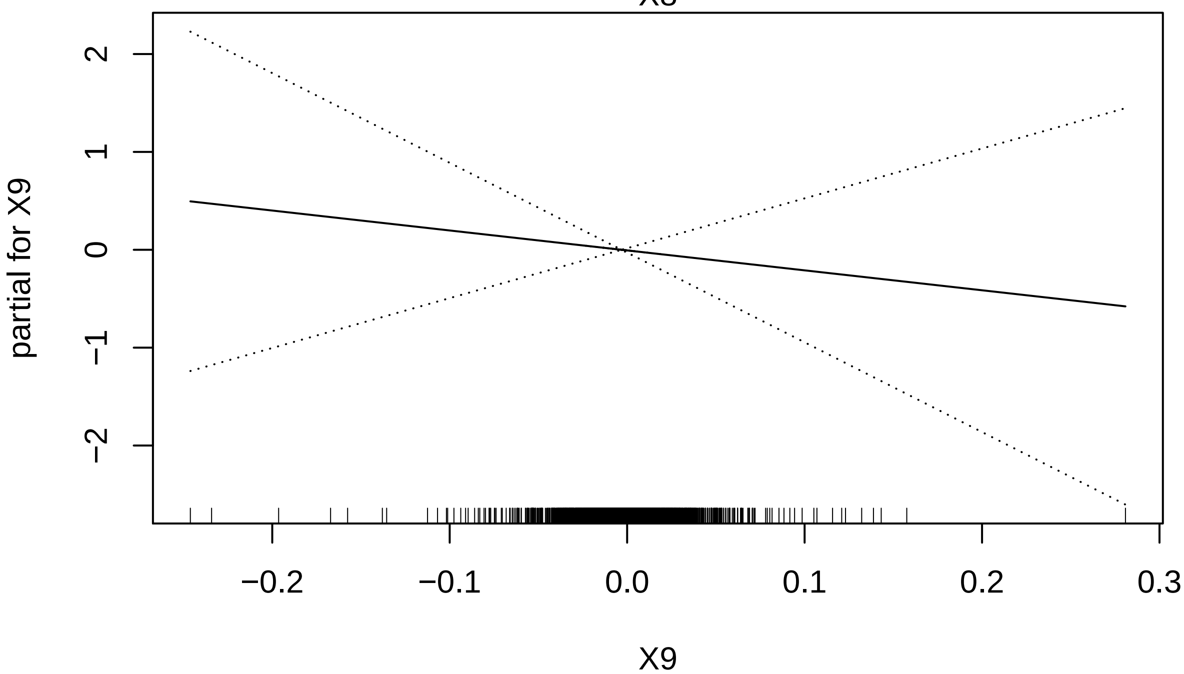
2
0
−2
−4
partial for X5
−0.3
−0.2
−0.1
0.0
0.1
X5


partial for X6
X6
−3
−2
−1
0
1
2
−0.2
−0.1
0.0
0.1
0.2

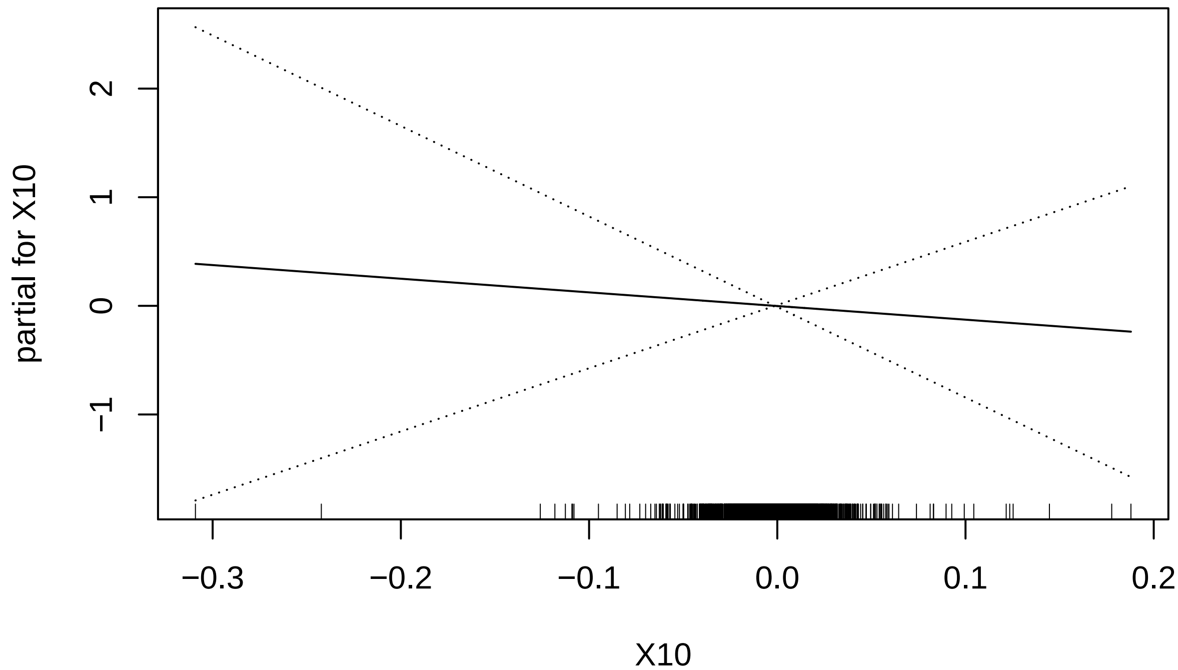

partial for X7
X7
−2
−1
0
1
2
3
−0.2
−0.1
0.0
0.1
0.2
0.3

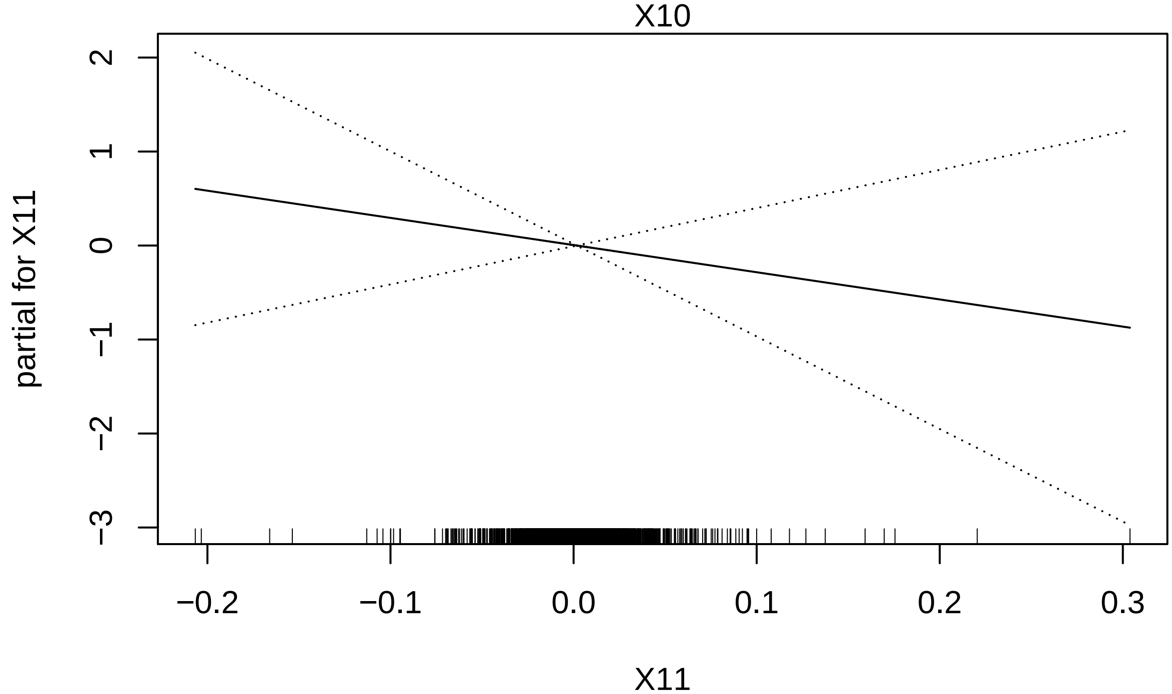


partial for X8
4
2
0
−2
−4
−0.2
−0.1
0.0
0.1
X8

partial for X9
2
1
0
−1
−2
−0.2
−0.1
0.0
0.1
0.2
0.3
X9

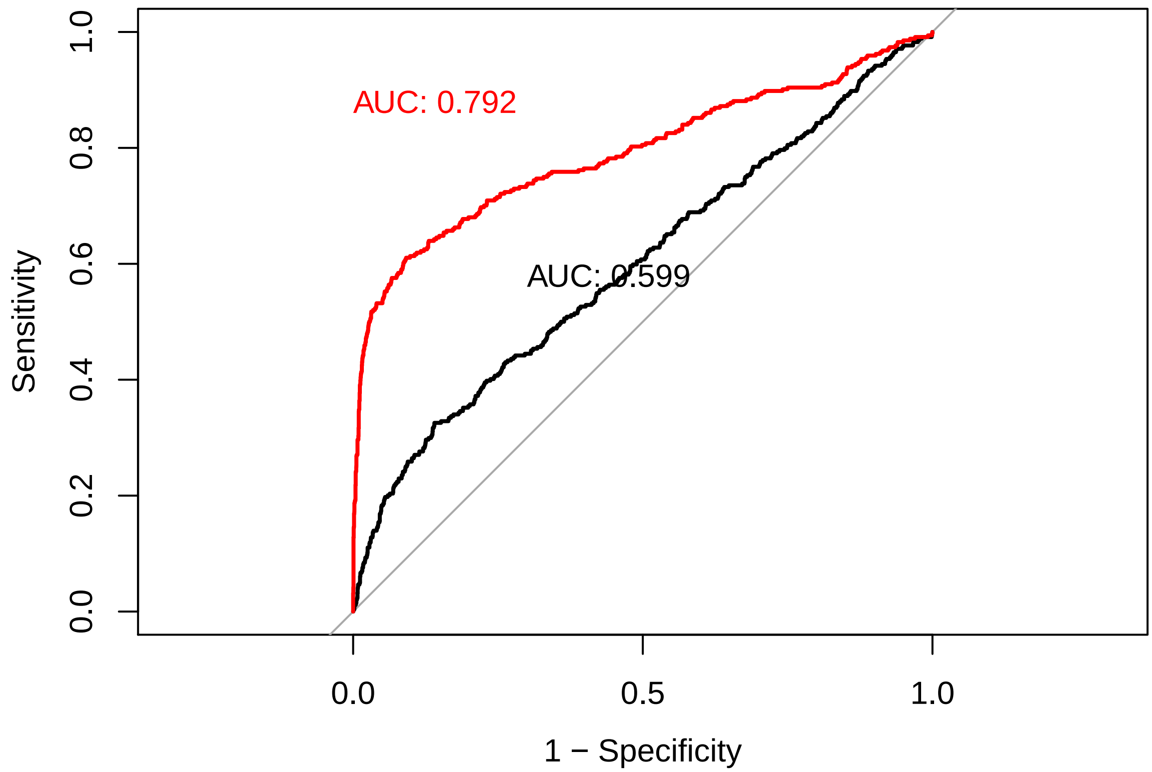



```
yhatg = predict(fit.gam, crowd[!train,])
fit.gama=gam(y ~ month+diversity+pastideas+pastaccept+commentsC+age+X1+X2
            +X3+X4+X5+X6+X7+X8+X9+X10+X11+votes+comments, binomial, data=crowd[train,])
```

```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

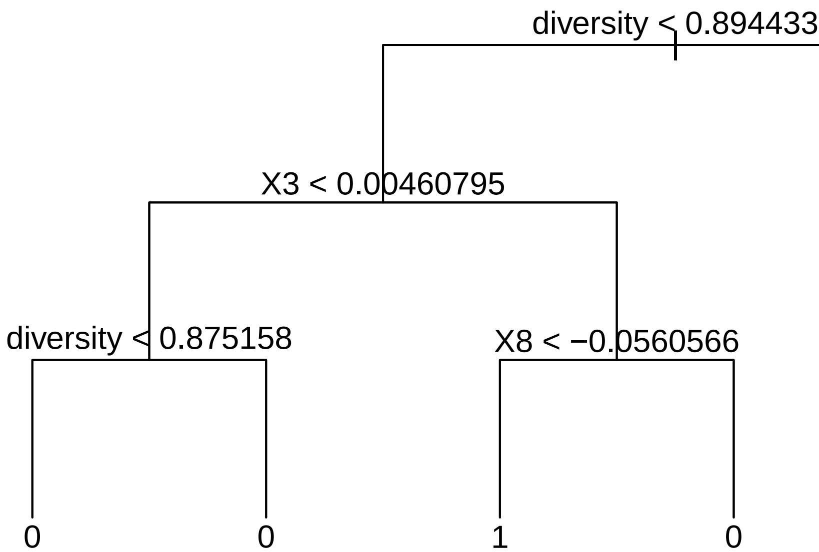
```
yhatga = predict(fit.gama, crowd[!train,])
plot.roc(crowd$y[!train], as.vector(yhatg), legacy.axes=T,
         print.auc=T, print.auc.x=.7, print.auc.y=.6)
plot.roc(crowd$y[!train], as.vector(yhatga), add=T, col=2,
         print.auc=T, print.auc.x=1, print.auc.y=.9, print.auc.col=2)
```



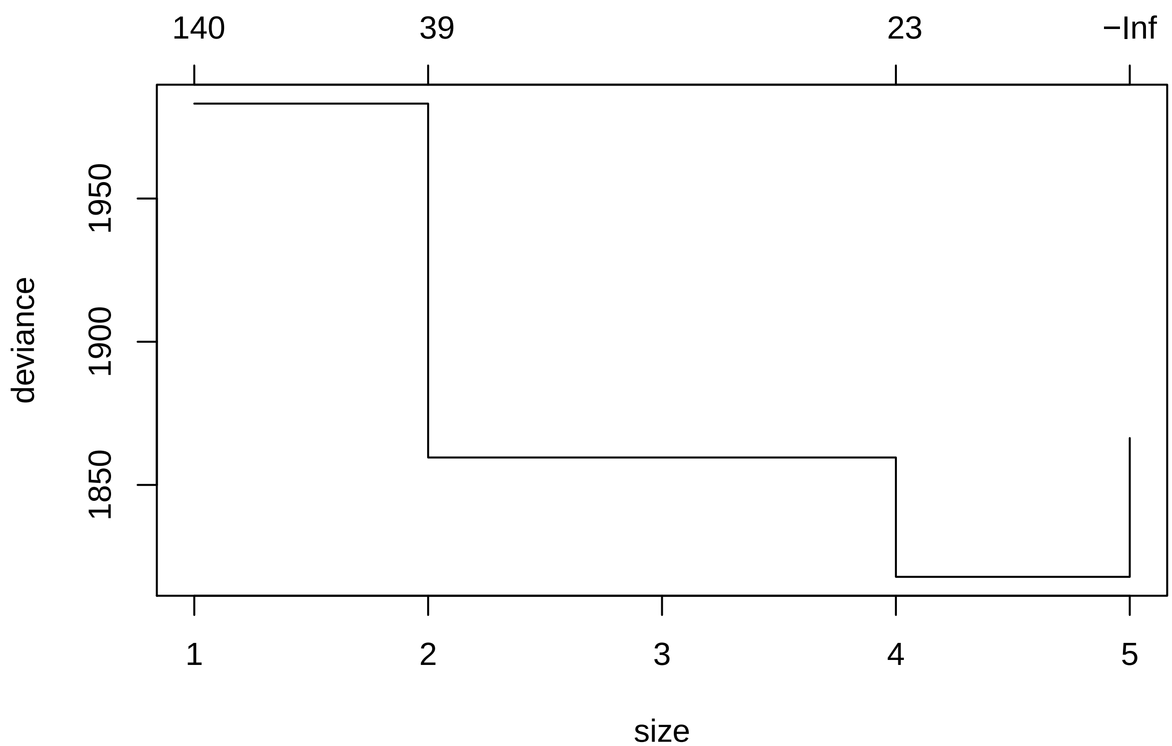

```r
#Classifier tree/ all
library(tree)
fit.tree = tree(factor(y) ~ month+diversity+pastideas+pastaccept+commentsC+age+X1+X2
                +X3+X4+X5+X6+X7+X8+X9+X10+X11, crowd[train,])
fit.tree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 3506 1982.00 0 ( 0.91843 0.08157 )
##    2) diversity < 0.894433 3244 1507.00 0 ( 0.93804 0.06196 )
##      4) X3 < 0.00460795 2842 1145.00 0 ( 0.94898 0.05102 )
##        8) diversity < 0.875158 1363  745.00 0 ( 0.92223 0.07777 ) *
##        9) diversity > 0.875158 1479  360.50 0 ( 0.97363 0.02637 ) *
##      5) X3 > 0.00460795 402  324.60 0 ( 0.86070 0.13930 )
##       10) X8 < -0.0560566 18   24.06 1 ( 0.38889 0.61111 ) *
##       11) X8 > -0.0560566 384  277.50 0 ( 0.88281 0.11719 ) *
##    3) diversity > 0.894433 262  330.20 0 ( 0.67557 0.32443 ) *
```

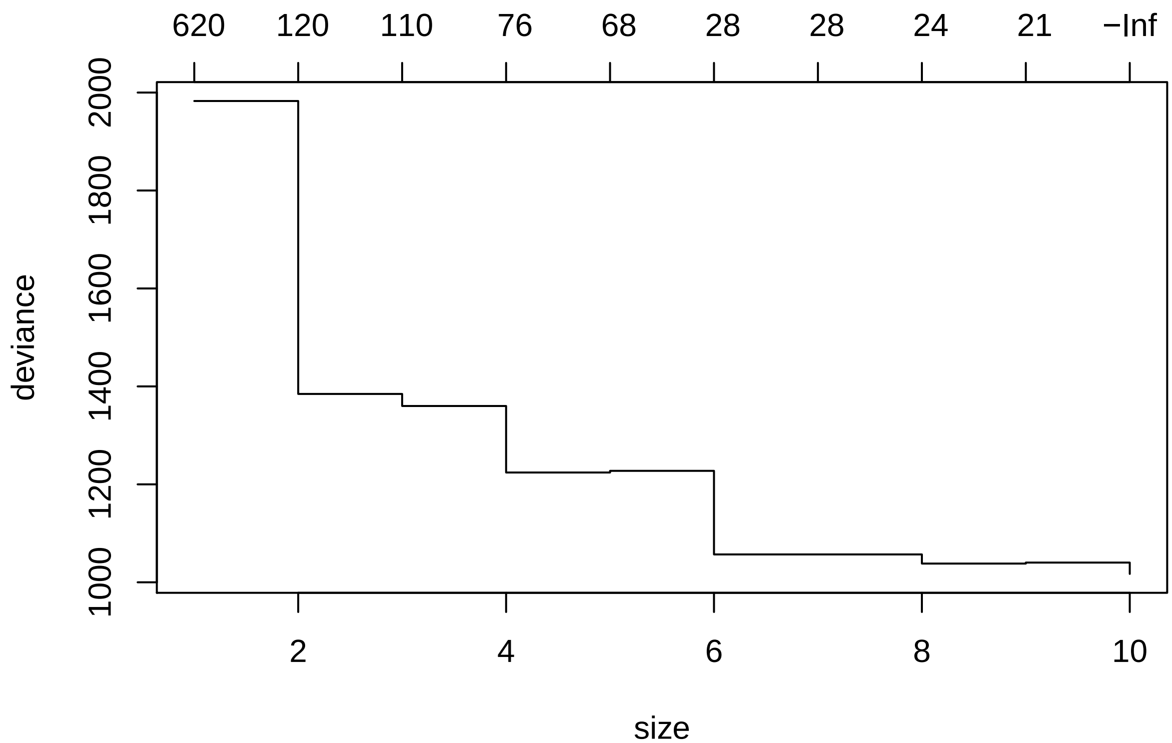
```r
plot(fit.tree, type = "uniform")
text(fit.tree)
```



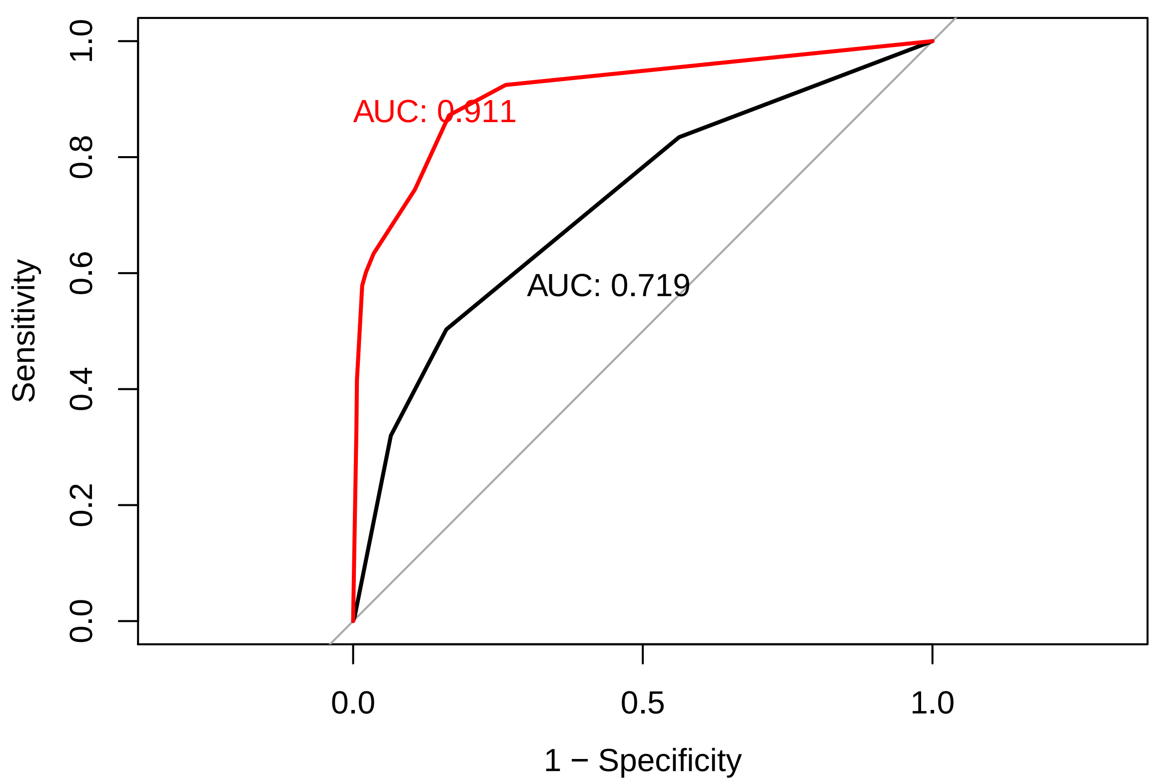

```
plot(cv.tree(fit.tree))
```




```
yhatt = predict(fit.tree, newdata = crowd[!train,])
fit.treea = tree(factor(y) ~ . , crowd[train,])
yhatta = predict(fit.treea, newdata=crowd[!train,])
plot(cv.tree(fit.treea))
```




```
plot.roc(crowd$y[!train], as.vector(yhatt[,2]), legacy.axes=T,
         print.auc=T, print.auc.x=.7, print.auc.y=.6)
plot.roc(crowd$y[!train], as.vector(yhatta[,2]), add=T, col=2,
         print.auc=T, print.auc.x=1, print.auc.y=.9, print.auc.col=2)
```



```r
#RF/ all
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

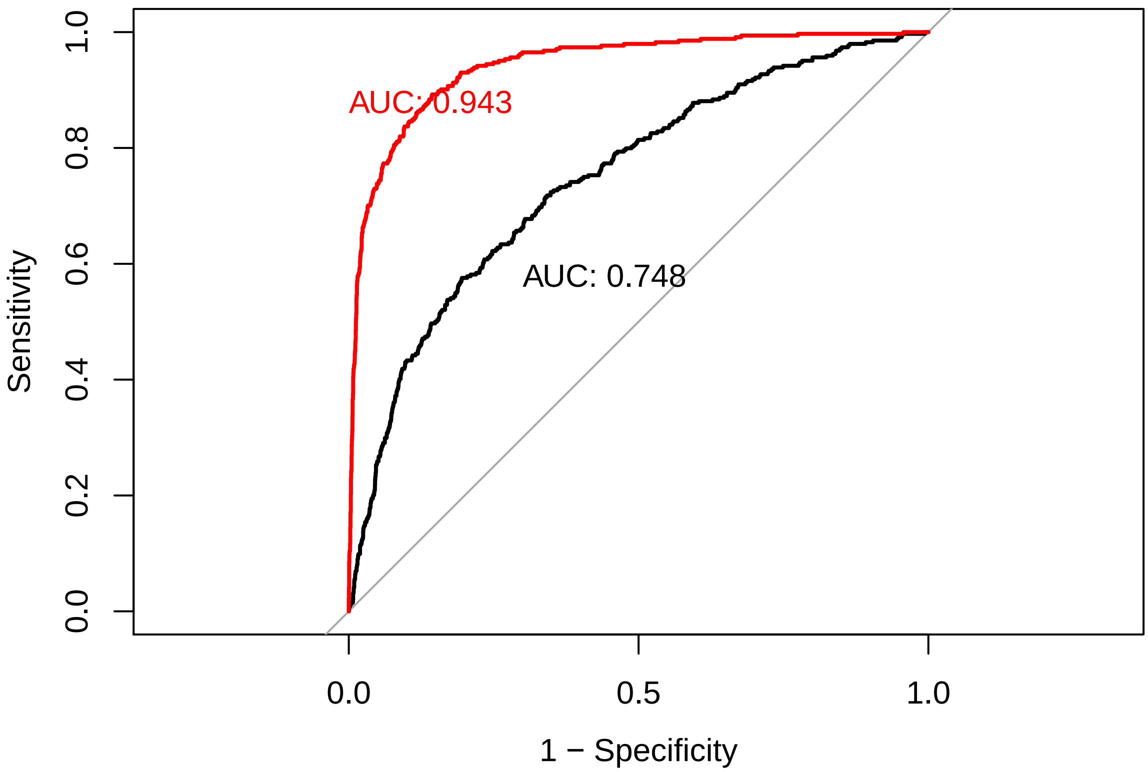
```r
fitcc=randomForest(x=crowd[train,c(1:6,9:19)],y=crowd$y[train],extest=crowd[!train,c(1:6,9:19)], ntree=
```

```
## Warning in randomForest.default(x = crowd[train, c(1:6, 9:19)], y =
## crowd$y[train], : The response has five or fewer unique values. Are you
## sure you want to do regression?
```

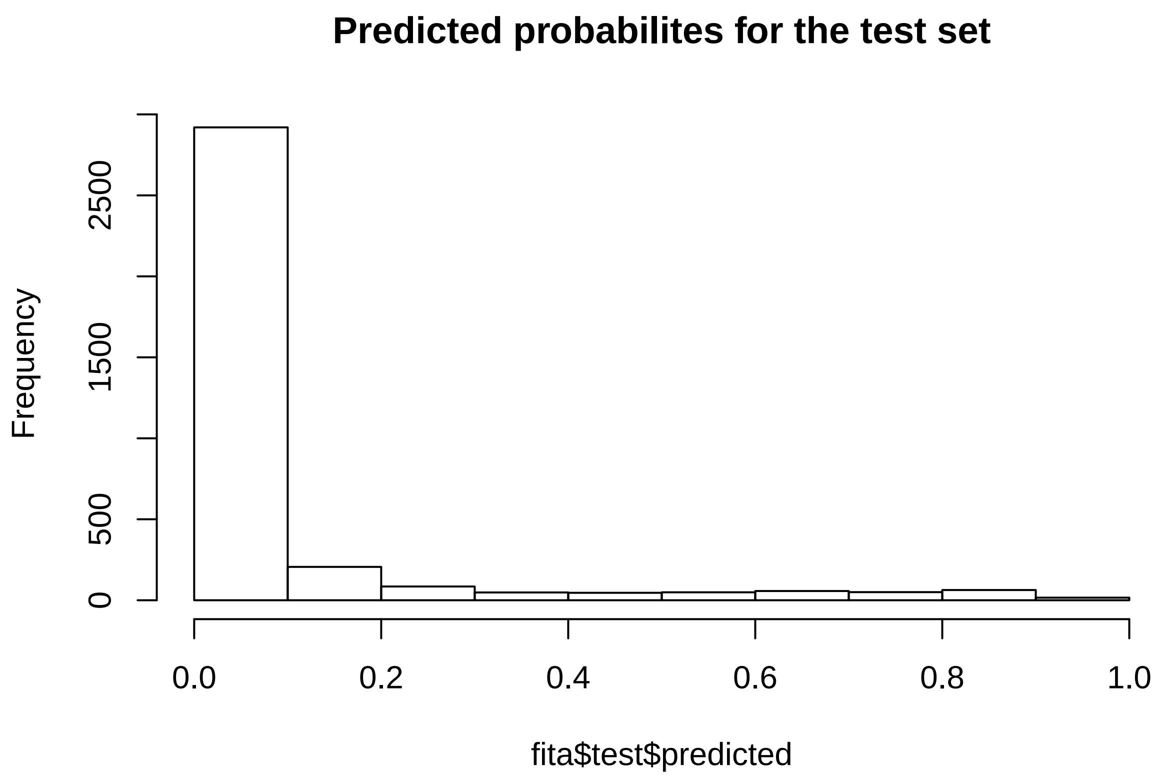
```r
fita=randomForest(x=crowd[train, c(1:19)], y=crowd$y[train], xtest=crowd[!train,c(1:19)], ntree=1000, ke
```

```
## Warning in randomForest.default(x = crowd[train, c(1:19)], y =
## crowd$y[train], : The response has five or fewer unique values. Are you
## sure you want to do regression?
```

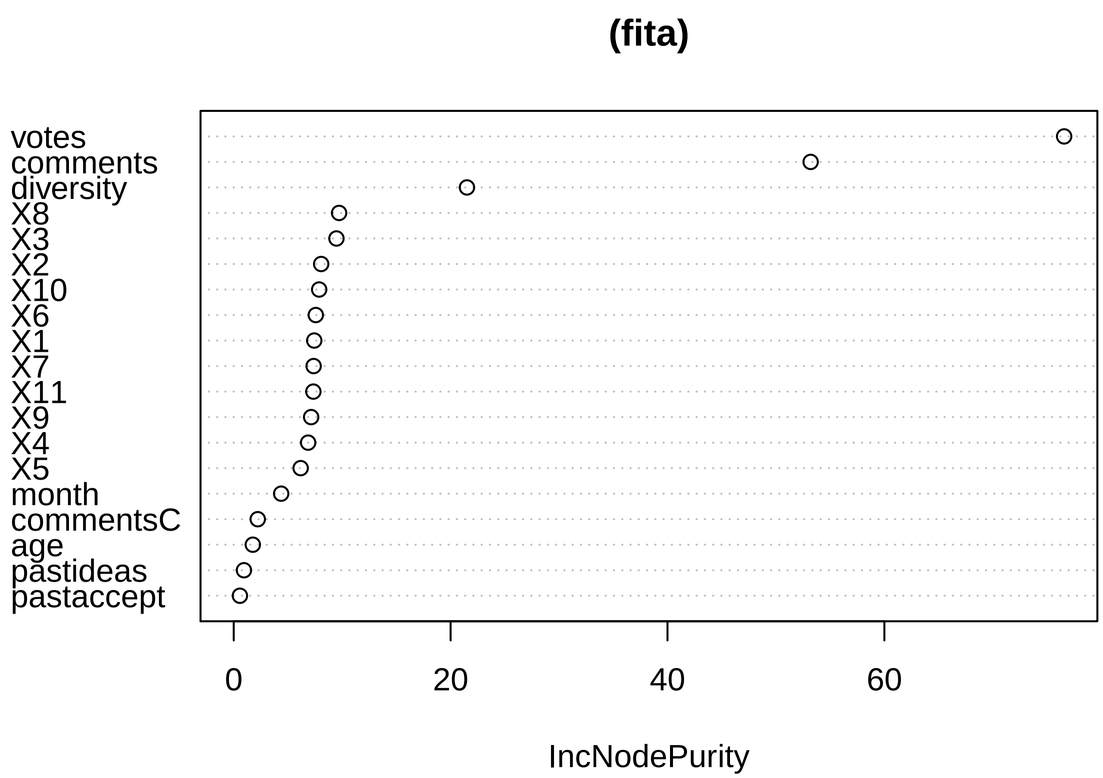
```r
fitvaluea=predict(fita, newdata=crowd[!train,])
fitvaluecc=predict(fitcc, newdata=crowd[!train,c(1:6,9:19)])
plot.roc(crowd$y[!train], as.vector(fitvaluecc), legacy.axes=T,
         print.auc=T, print.auc.x=.7, print.auc.y=.6)
plot.roc(crowd$y[!train], as.vector(fitvaluea), add=T, col=2,
         print.auc=T, print.auc.x=1, print.auc.y=.9, print.auc.col=2)
```



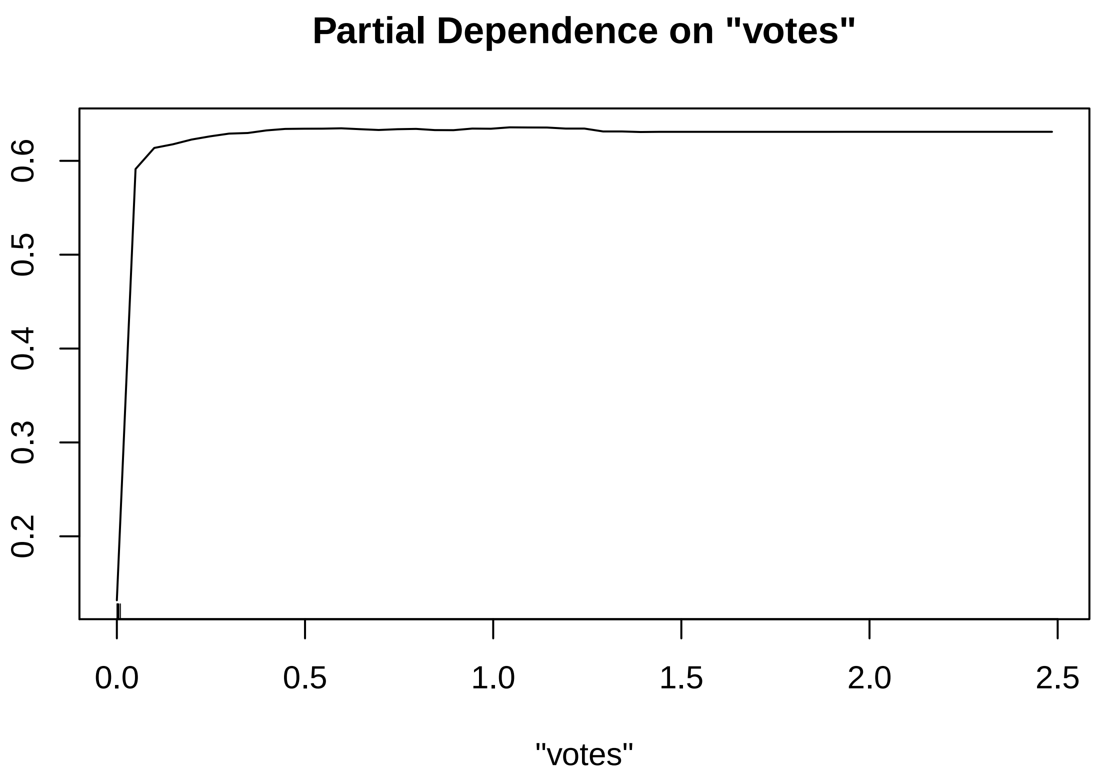

```r
#analyze RF
hist(fita$test$predicted, main = "Predicted probabilites for the test set")
```



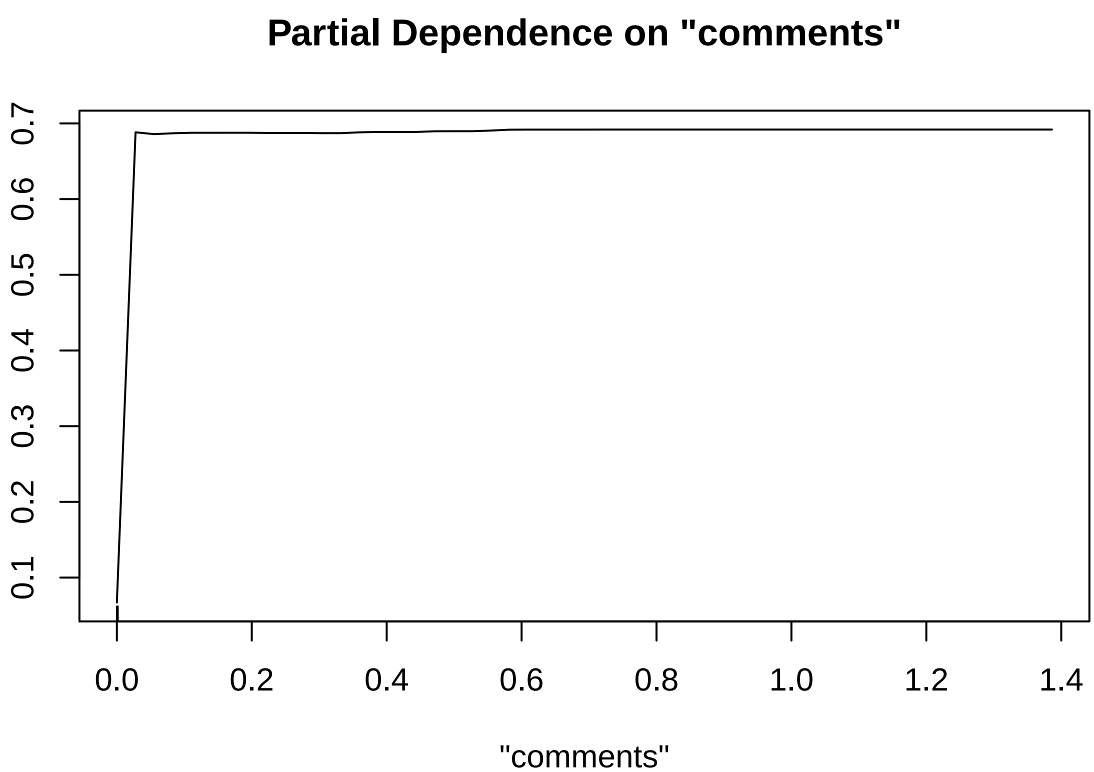

```r
varImpPlot((fita))
```



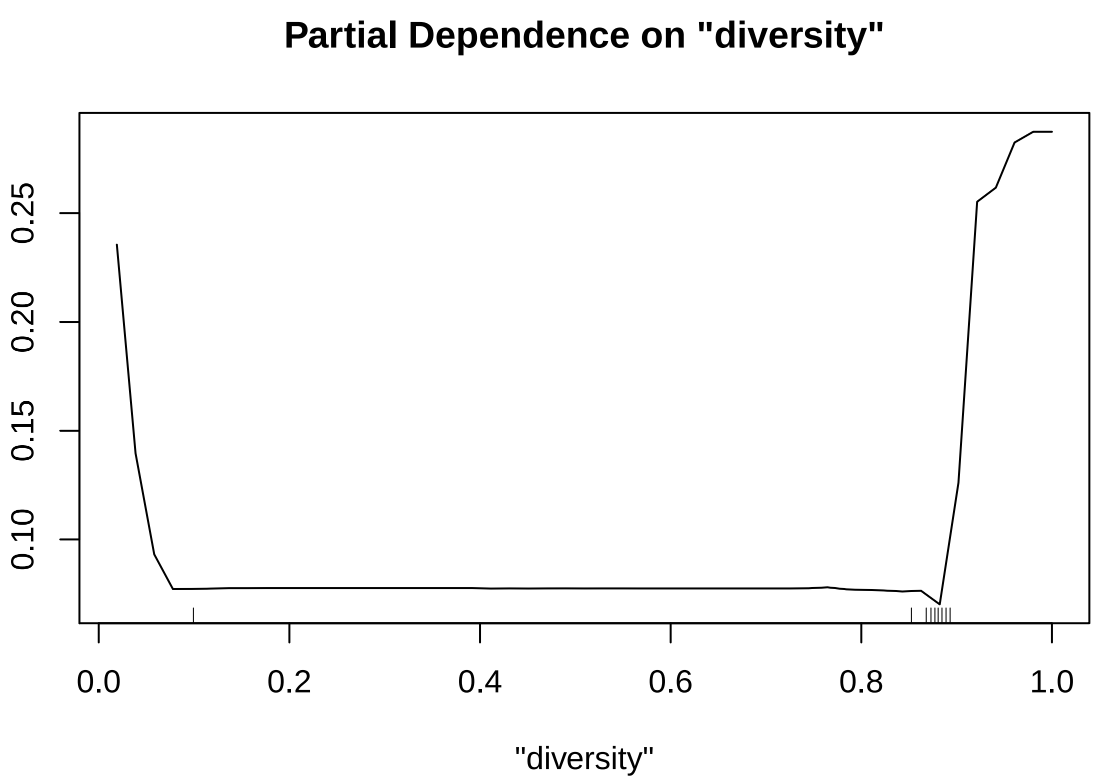

```r
partialPlot(fita, crowd[train,], "votes")
```




```r
partialPlot(fita, crowd[train,], "comments")
```




```r
partialPlot(fita, crowd[train,], "diversity")
```




```
# GBM/ all
library(gbm)
```

```
## Loaded gbm 2.1.5
```

```
fit = gbm(y ~ ., data=crowd[train,], interaction.depth=1, n.trees=500, shrinkage=0.02)
```

```
## Distribution not specified, assuming bernoulli ...
```

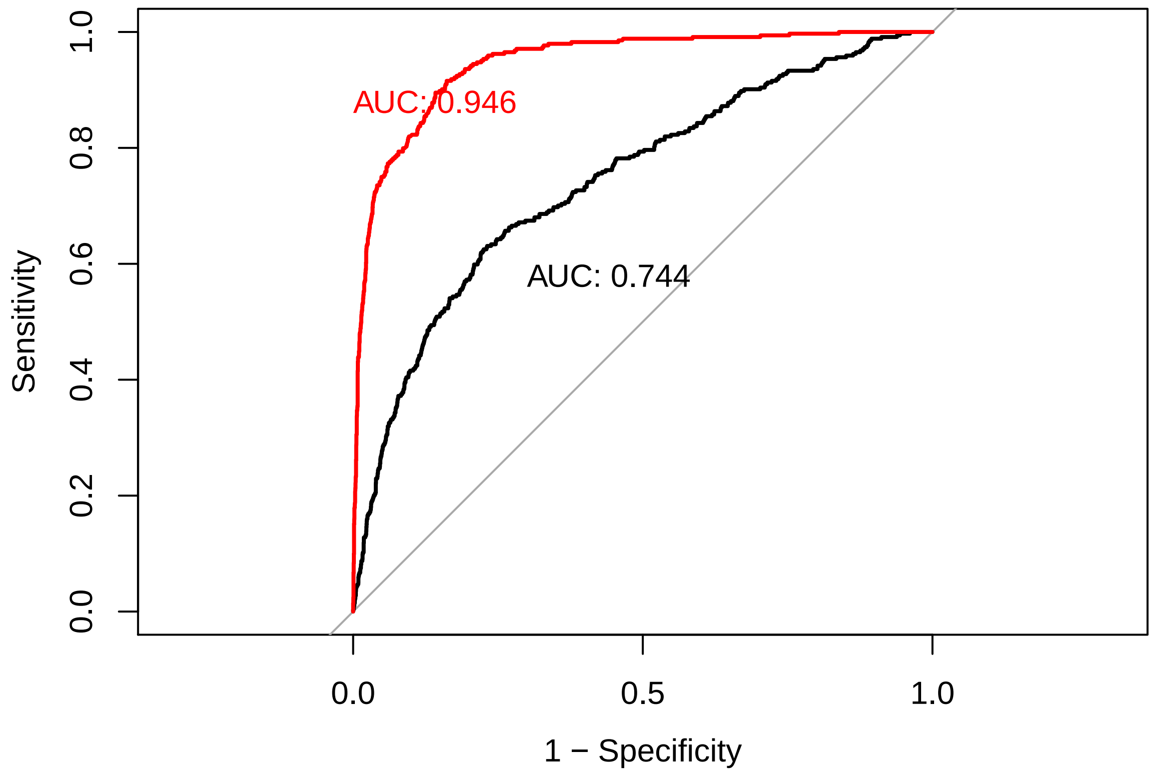
```
fitdp2= gbm(y ~ ., data=crowd[train,], interaction.depth=2, n.trees=500, shrinkage=0.02) # find that we
```

```
## Distribution not specified, assuming bernoulli ...
```

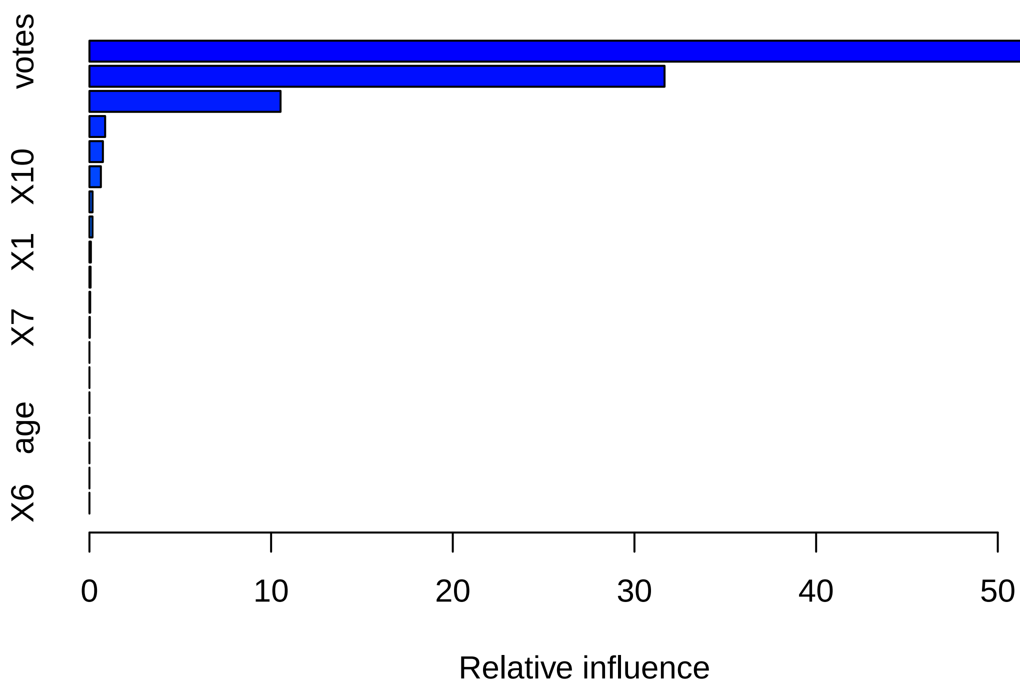
```
yhat = predict(fit, newdata=crowd[!train,], n.trees=500)
fitcc =gbm(y~month+diversity+pastideas+pastaccept+commentsC+age+X1+X2
           +X3+X4+X5+X6+X7+X8+X9+X10+X11, data=crowd[train,],
           interaction.depth = 2, n.trees = 500, shrinkage = 0.02)
```

```
## Distribution not specified, assuming bernoulli ...
```

```
yhatcc=predict(fitcc, newdata=crowd[!train,], n.trees=500)
plot.roc(crowd$y[!train], as.vector(yhatcc), legacy.axes=T,
         print.auc=T, print.auc.x=.7, print.auc.y=.6, print.quc.col=1)
plot.roc(crowd$y[!train], as.vector(yhat), add=T, col=2,
         print.auc=T, print.auc.x=1, print.auc.y=.9, print.auc.col=2)
```




```r
summary(fit)
```




```
##                   var     rel.inf
## votes           votes 55.04241635
## comments     comments 31.65762565
## diversity   diversity 10.51881399
## X8                 X8  0.86415074
## X3                 X3  0.74176020
```

```
## X10               X10  0.62922726
## X2                 X2  0.17228313
## commentsC   commentsC  0.16853567
## X1                 X1  0.07977617
## X11               X11  0.06296731
## X9                 X9  0.04094980
## X7                 X7  0.02149372
## month           month  0.00000000
## pastideas   pastideas  0.00000000
## pastaccept pastaccept  0.00000000
## age               age  0.00000000
## X4                 X4  0.00000000
## X5                 X5  0.00000000
## X6                 X6  0.00000000
```

```
mean((crowd$y[!train] - yhat)^2)
```

```
## [1] 15.12981
```

```
#try pca
fitpca= prcomp(crowd[,1:19], scale=T)
summary(fitpca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     1.6280 1.4248 1.23950 1.14735 1.11179 1.10424
## Proportion of Variance 0.1395 0.1069 0.08086 0.06929 0.06506 0.06418
## Cumulative Proportion  0.1395 0.2463 0.32720 0.39649 0.46155 0.52572
##                            PC7     PC8    PC9    PC10    PC11    PC12
## Standard deviation     1.04668 1.00838 0.9853 0.92964 0.91303 0.85768
## Proportion of Variance 0.05766 0.05352 0.0511 0.04549 0.04388 0.03872
## Cumulative Proportion  0.58338 0.63690 0.6880 0.73348 0.77736 0.81608
##                           PC13   PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.84562 0.8236 0.78023 0.75003 0.72515 0.52086
## Proportion of Variance 0.03764 0.0357 0.03204 0.02961 0.02768 0.01428
## Cumulative Proportion  0.85371 0.8894 0.92145 0.95106 0.97874 0.99302
##                           PC19
## Standard deviation     0.36426
## Proportion of Variance 0.00698
## Cumulative Proportion  1.00000
```

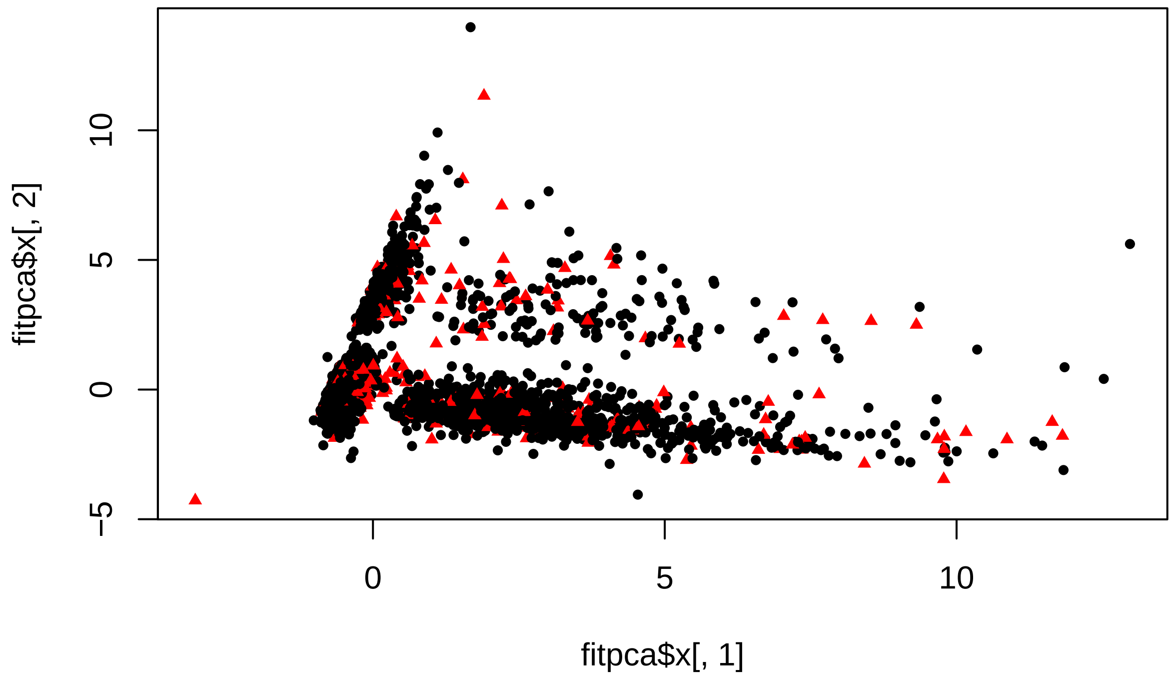
```
fitpca$rotation
```

```
##                     PC1         PC2          PC3          PC4          PC5
## month       0.001810577 -0.02120306  0.010098075 -0.025499146  0.047064618
## diversity  -0.118308021 -0.62020978  0.009965743 -0.008425856  0.040825885
## pastideas   0.553512805 -0.12176910 -0.036053390  0.074397922 -0.022691975
## pastaccept  0.457796217 -0.10288292 -0.001964032  0.016721906  0.027389392
## commentsC   0.379187307 -0.06861053  0.048988189  0.012949639 -0.006821879
## age         0.530669982 -0.10208939 -0.030873241  0.065835032 -0.035484227
## votes       0.042992680  0.01634128  0.532503323 -0.319358019  0.278281817
## comments    0.072091515  0.05288627  0.544052708 -0.317071812  0.262324830
## X1         -0.144058172 -0.35552681  0.172638872  0.276520838  0.070271718
## X2          0.089298801  0.19550112 -0.259630078 -0.266295634 -0.015311221
## X3         -0.026334197  0.04391016  0.366323646  0.335436078 -0.375541403
## X4          0.025719406  0.34991498  0.216652534  0.385605837 -0.082469609
```

```
## X5          0.012559742  0.01134860  0.012115435 -0.189317079  0.007315969
## X6          0.008740007 -0.02548163 -0.060386239  0.263834691  0.478207893
## X7          0.015392881  0.06045585  0.132827711 -0.336959712 -0.260749971
## X8          0.006195661  0.19428485 -0.281797167 -0.284580998  0.001687943
## X9          0.057585040  0.38670295 -0.010851100  0.142439070  0.268959850
## X10        -0.030969688 -0.25979086 -0.020162485 -0.239362302 -0.289957412
## X11        -0.052347889 -0.15193858 -0.189407447 -0.017048379  0.491062897
##                     PC6           PC7         PC8          PC9
## month      -0.1649161406 -0.0251440845 -0.77593442  0.579746862
## diversity   0.0555619939  0.0678842350  0.02219956  0.040316928
## pastideas   0.0001317376  0.0093917125  0.01298128 -0.033050133
## pastaccept -0.0528512165  0.0034205038 -0.00548753 -0.020827367
## commentsC  -0.0937482048 -0.0130187839 -0.04255297  0.046558709
## age        -0.0058577459  0.0132990594  0.01725255 -0.018318955
## votes       0.1465940030  0.0253999633 -0.08619250 -0.034866772
## comments    0.1197232052  0.0006690717  0.02788105 -0.031904675
## X1         -0.3381221065 -0.0446116462 -0.06397222 -0.141412720
## X2          0.3832298904  0.1892165158  0.12185380  0.289590032
## X3          0.0180774233  0.2914602087  0.09619248  0.130928243
## X4          0.0985229484  0.4111200976 -0.06624820  0.004853525
## X5         -0.4676191581  0.5531439934  0.19383042  0.208908192
## X6          0.0992787312  0.2983656331 -0.23528975 -0.393699472
## X7         -0.4921416558 -0.0676514426  0.09512542 -0.152122281
## X8         -0.1899887623  0.2973117940 -0.34061949 -0.446540036
## X9         -0.2216789077 -0.3355092485  0.10948851  0.039733362
## X10         0.2914183398  0.1959331426 -0.11552289 -0.135207927
## X11        -0.1008029477  0.2509744294  0.33316896  0.306493727
##                  PC10         PC11        PC12         PC13         PC14
## month      -0.08927475  0.082439172 -0.06751703  0.064010970 -0.002762619
## diversity  -0.06859048 -0.067036358 -0.01032548 -0.034634637  0.060770676
## pastideas   0.05932272  0.003150666 -0.07087442  0.215094231  0.054673345
## pastaccept  0.03630493 -0.102661915 -0.24681219  0.200621437  0.170758287
## commentsC  -0.11360802  0.119669567  0.46608270 -0.674562754 -0.201962267
## age         0.02273462  0.028478836  0.04431338  0.125410295 -0.014608905
## votes       0.08175909 -0.071021682  0.05933595  0.183030315  0.060399407
## comments    0.01695187  0.009763738  0.04398890 -0.094599089  0.019484784
## X1          0.10942575 -0.117311894  0.19350813 -0.138620827  0.621768131
## X2         -0.31464610 -0.226802731  0.01378318 -0.208271745  0.577082682
## X3         -0.34059633  0.151669008  0.09615964  0.272981398  0.107277762
## X4          0.17775738  0.038059800  0.12568529 -0.092345376  0.010252244
## X5          0.32408474 -0.160940856 -0.35562876 -0.214435662 -0.062901865
## X6         -0.43357018  0.031497264 -0.37567764 -0.174106199 -0.083958091
## X7         -0.56913968  0.137480308 -0.10883432  0.008528762  0.001296235
## X8          0.10423699 -0.002457631  0.41952633  0.244603478  0.152547260
## X9          0.12354543  0.453679880 -0.18712217 -0.090777507  0.357436878
## X10         0.21237966  0.688671374 -0.19983724 -0.140206959  0.141077661
## X11        -0.12387465  0.377281132  0.33415811  0.291958349 -0.077111384
##                  PC15          PC16        PC17         PC18
## month       0.05606463  0.0587637108  0.02031807  0.010721677
## diversity  -0.05387082  0.0172266211  0.03628543 -0.755625806
## pastideas   0.11874912  0.0351958212  0.20356400  0.005088593
## pastaccept -0.15747514  0.0489102902 -0.74415433 -0.001379942
## commentsC  -0.24034100 -0.1334923695 -0.13101415  0.006262871
## age         0.19739852 -0.0043578127  0.50421221 -0.006570019
```

```
## votes       -0.07861608 -0.6659424025  0.04409236  0.001261964
## comments     0.05047435  0.7038422223  0.03029889 -0.005038546
## X1           0.16581784  0.0004599241  0.02523929  0.322638807
## X2           0.04412677 -0.0343288821  0.05938102 -0.008101662
## X3          -0.49573028  0.0711673751  0.10586442  0.040501327
## X4           0.53185345 -0.0629588963 -0.21076603 -0.320265781
## X5          -0.16527947 -0.0232842650  0.14582210  0.036303250
## X6          -0.05862371 -0.0236484467  0.08335773  0.098396868
## X7           0.37123081 -0.0945185161 -0.07763985 -0.099794331
## X8          -0.21807150  0.1123937261  0.02273989 -0.176705349
## X9          -0.22198050 -0.0384443586  0.11671492 -0.350183062
## X10          0.06588200 -0.0391618028 -0.04975637  0.165326172
## X11          0.14764877  0.0045082364 -0.13476398  0.135133556
##                      PC19
## month       -0.0039628482
## diversity   -0.0029513271
## pastideas   -0.7453091289
## pastaccept   0.2289015097
## commentsC   -0.0537266610
## age          0.6229648075
## votes       -0.0105652284
## comments     0.0017301621
## X1           0.0037015305
## X2          -0.0109607865
## X3           0.0046172725
## X4          -0.0052443691
## X5          -0.0061576082
## X6           0.0004136252
## X7          -0.0221208800
## X8           0.0013545400
## X9           0.0038906861
## X10          0.0167498416
## X11          0.0002259030
```

```r
plot(fitpca$x[,1], fitpca$x[,2], col=1+crowd$y, pch=16+crowd$y, cex=0.7)
```




```
library(scatterplot3d)
plot3d <- with(crowd, scatterplot3d(crowd[,7], crowd[,8], crowd[,2], color = 1+crowd$y, pch = 16),cex.sy
```