capstone project

Hanjie Shi

8/8/2018

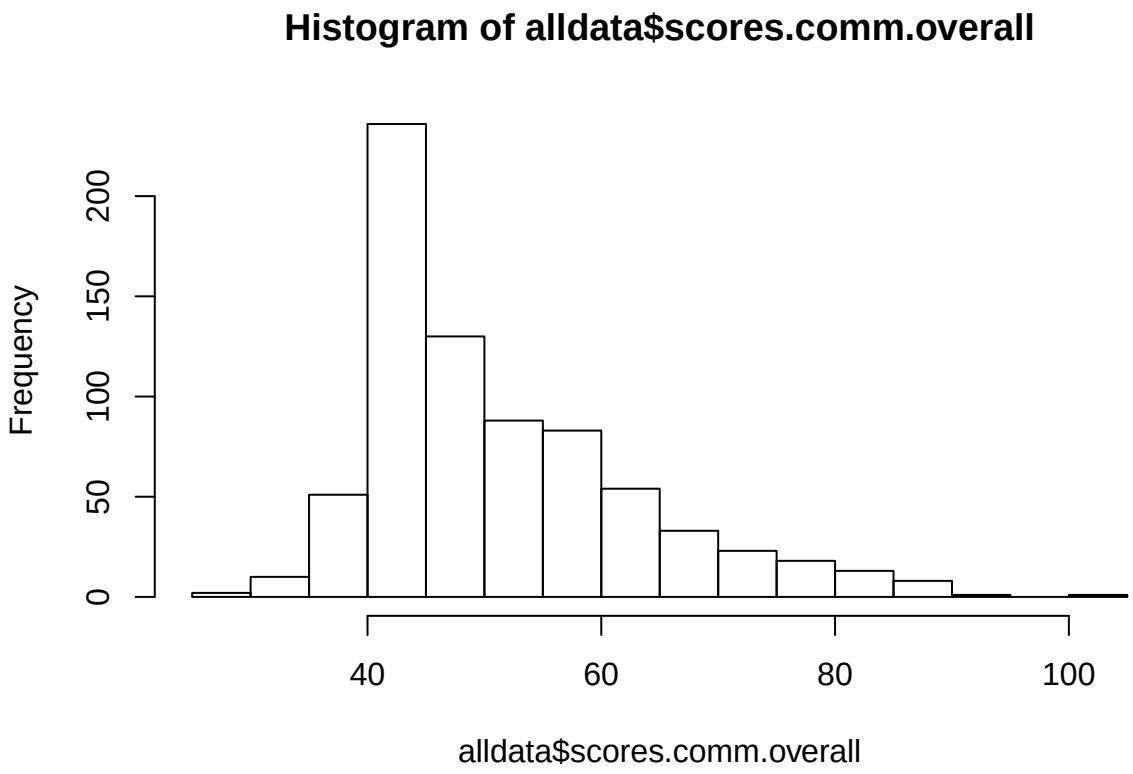
Objective:

This project using vairous machine learning and regression techniques to analyze the relationship between rank/scores of a company with respect to different predictors. Also I have combine the stock return data to the datasets to find out top factors could influence the stock price. It could provide as an outlook to the company to improve performance in the future.

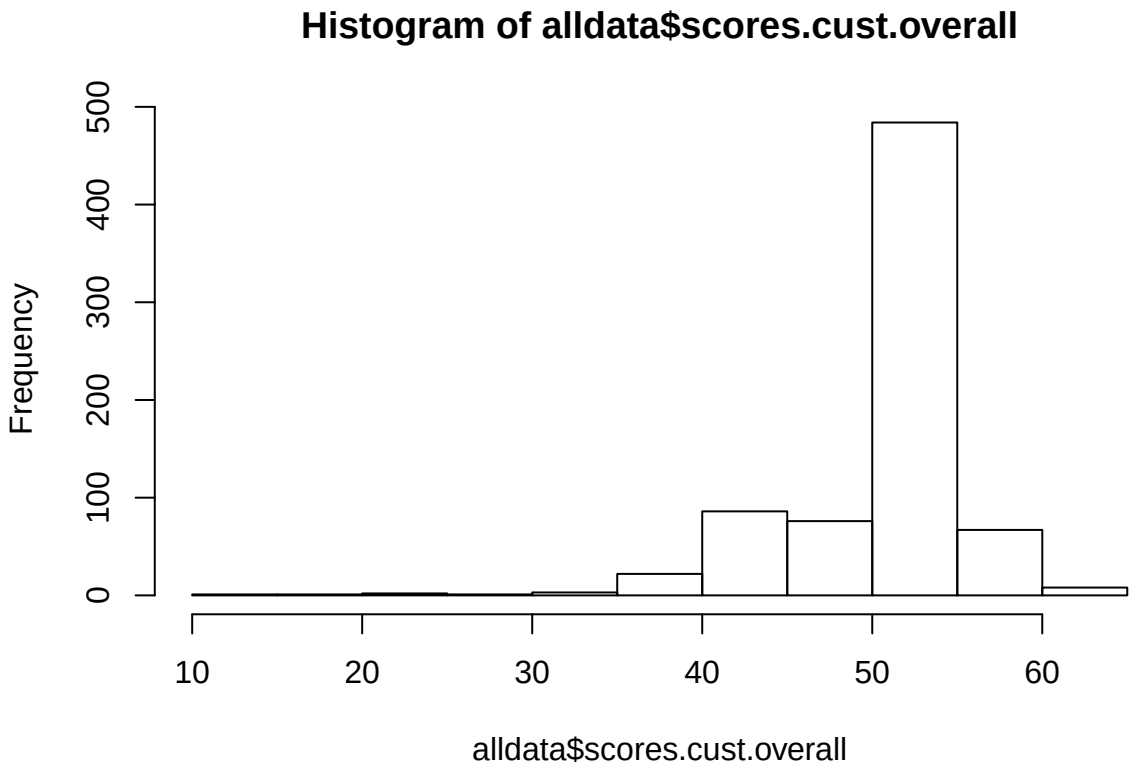
For the specfic machine learning techniques, I have used randomforest models since it reduces variance comparing to the simple tree models and I used the variable importance algorithm (at each split, you can calculate how much this split reduces node impurity. For regression trees, indeed, the difference between RSS before and after the split. This is summed over all splits for that variable, over all trees). Besides, random forest could resolve 'small n big p' problems.

When I fit the full data and I want to do variable selection, Lasso regression and stepwise algo are two good ways to reduce variables. Lasso regression add the penalty terms comparing to the classis linear models. In addition, I have used cross validation to find the optimal lambda values to fit the model. The stepwise regression find out the variable with lowest AIC.

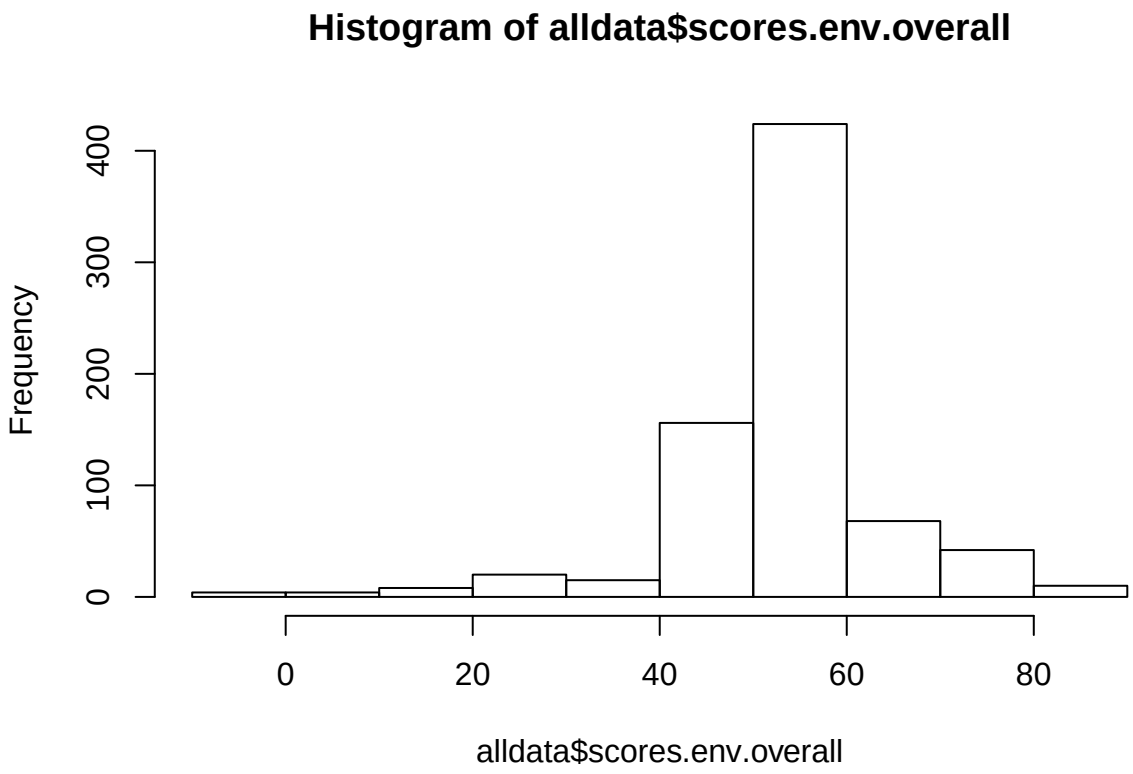
```
price<-read.csv('price.csv')
colnames(price)[1]<-"Date"
odd_indexes<-seq(1,nrow(price),2)
price<-price[odd_indexes,]
rownames(price) <- price[,1]
logprice<-log(price[,2:ncol(price)])
lret<-apply(logprice,2,diff)
ret_mean<-as.data.frame(apply(lret,2,mean))
colnames(ret_mean)[1]<-"mean"
ret_mean <- cbind(ticker = rownames(ret_mean), ret_mean)
rownames(ret_mean) <- 1:nrow(ret_mean)

stock_data<-read.csv('jc_companies.csv')
alldata<-merge(stock_data,ret_mean,by='ticker')
alldata<-as.data.table(alldata)
alldata<-na.omit(alldata)
hist(alldata$scores.comm.overall)
```



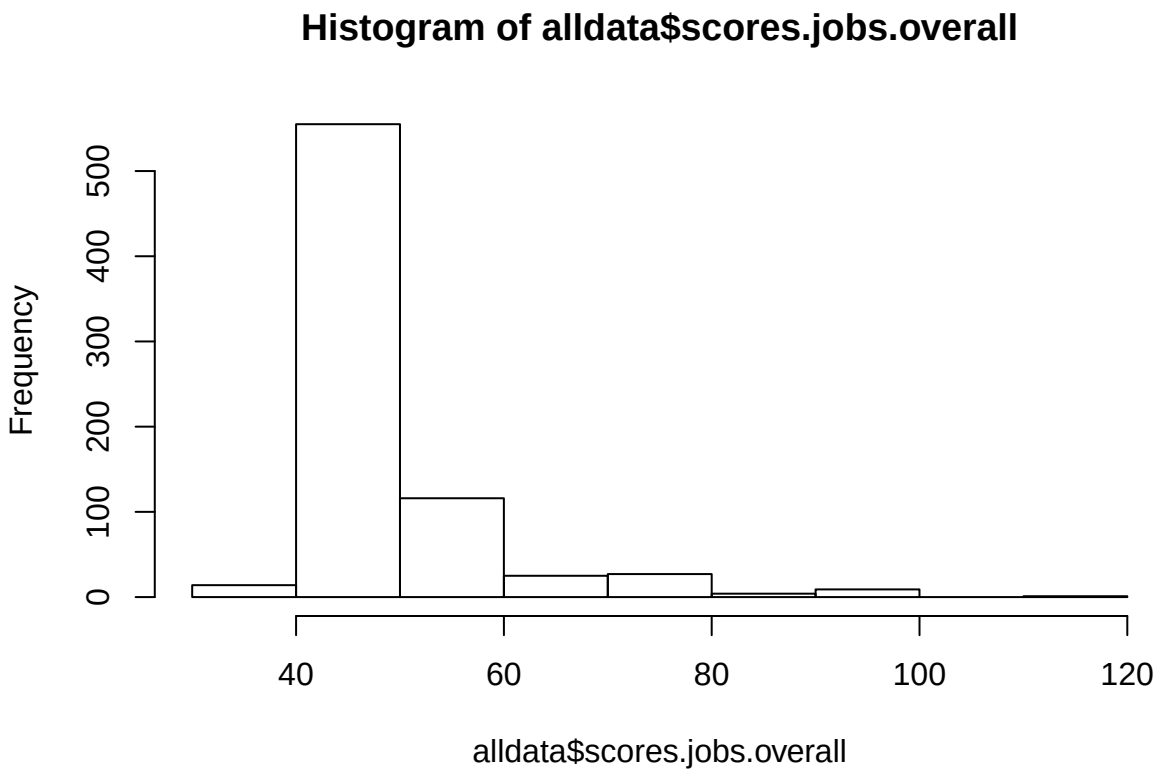

```
hist(alldata$scores.cust.overall)
```



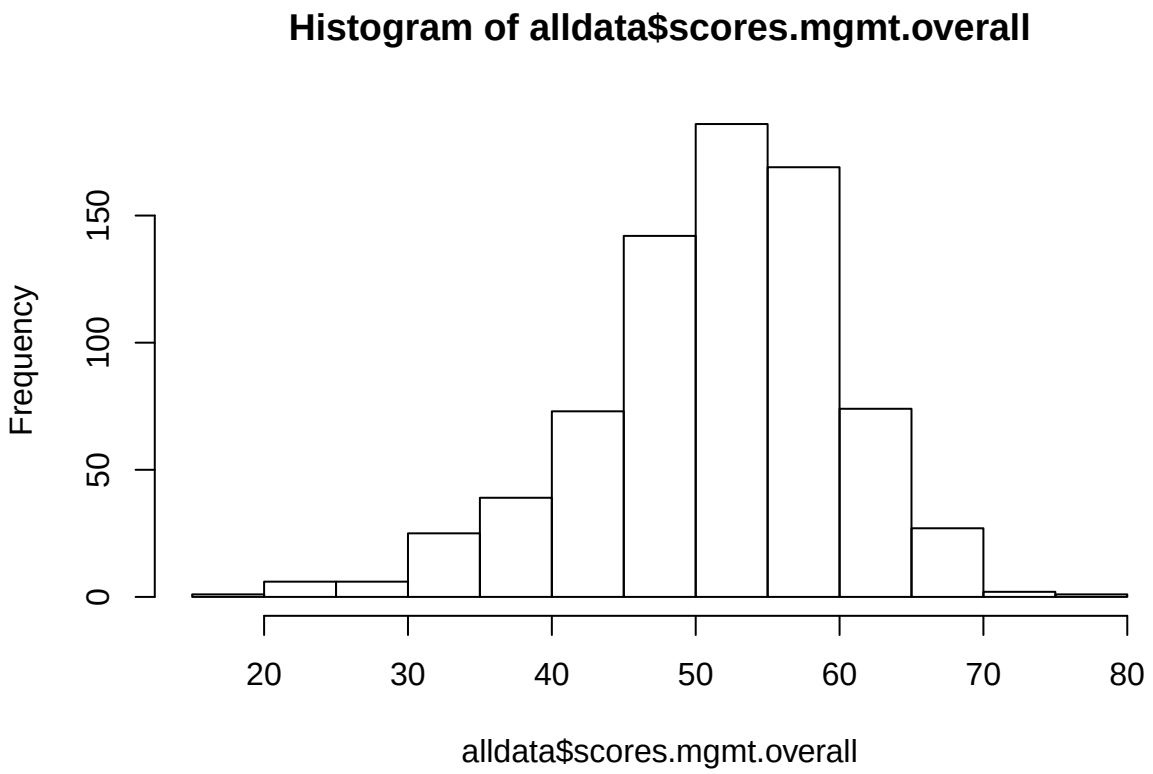

```
hist(alldata$scores.env.overall)
```



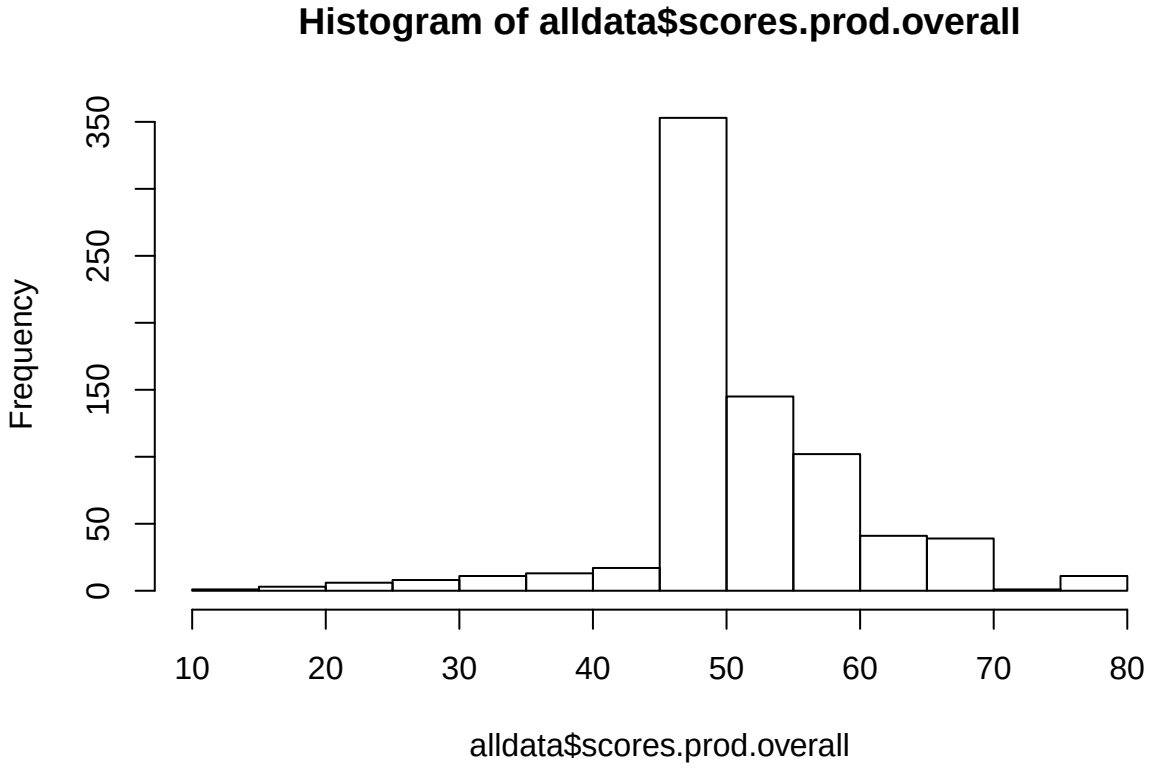

```
hist(alldata$scores.jobs.overall)
```



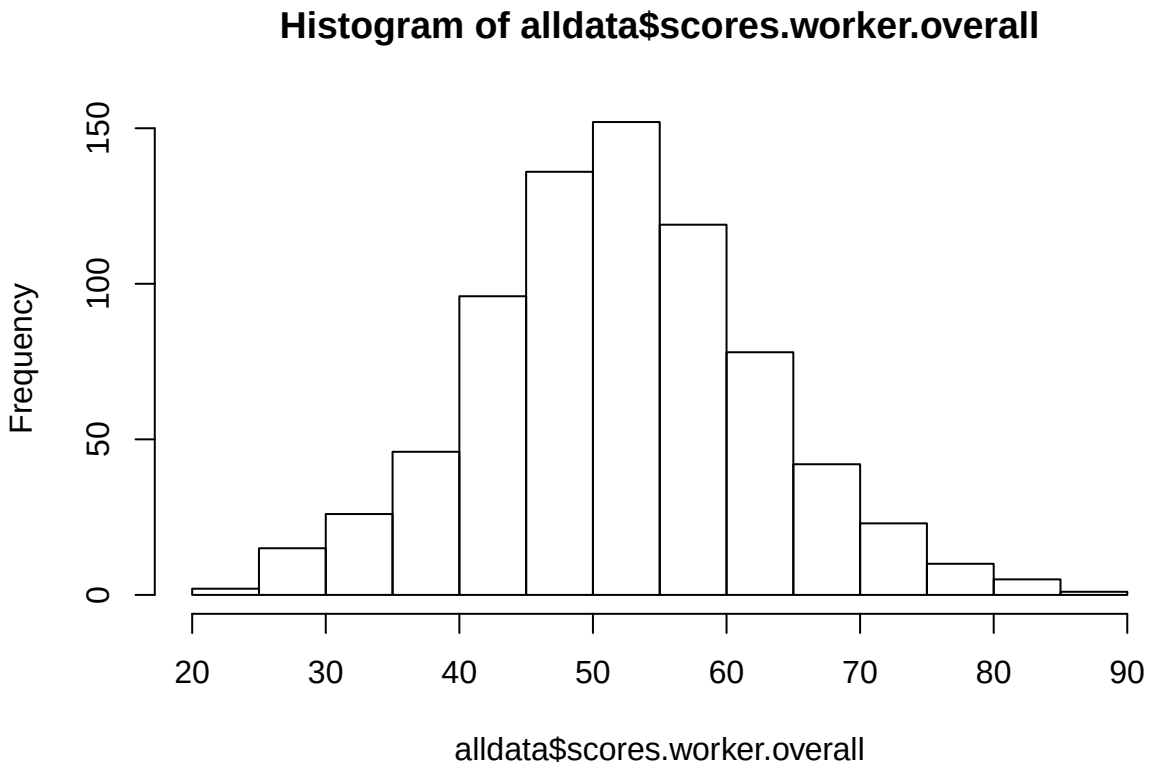

```
hist(alldata$scores.mgmt.overall)
```




```
hist(alldata$scores.prod.overall)
```




```
hist(alldata$scores.worker.overall)
```



Average Score by Industry

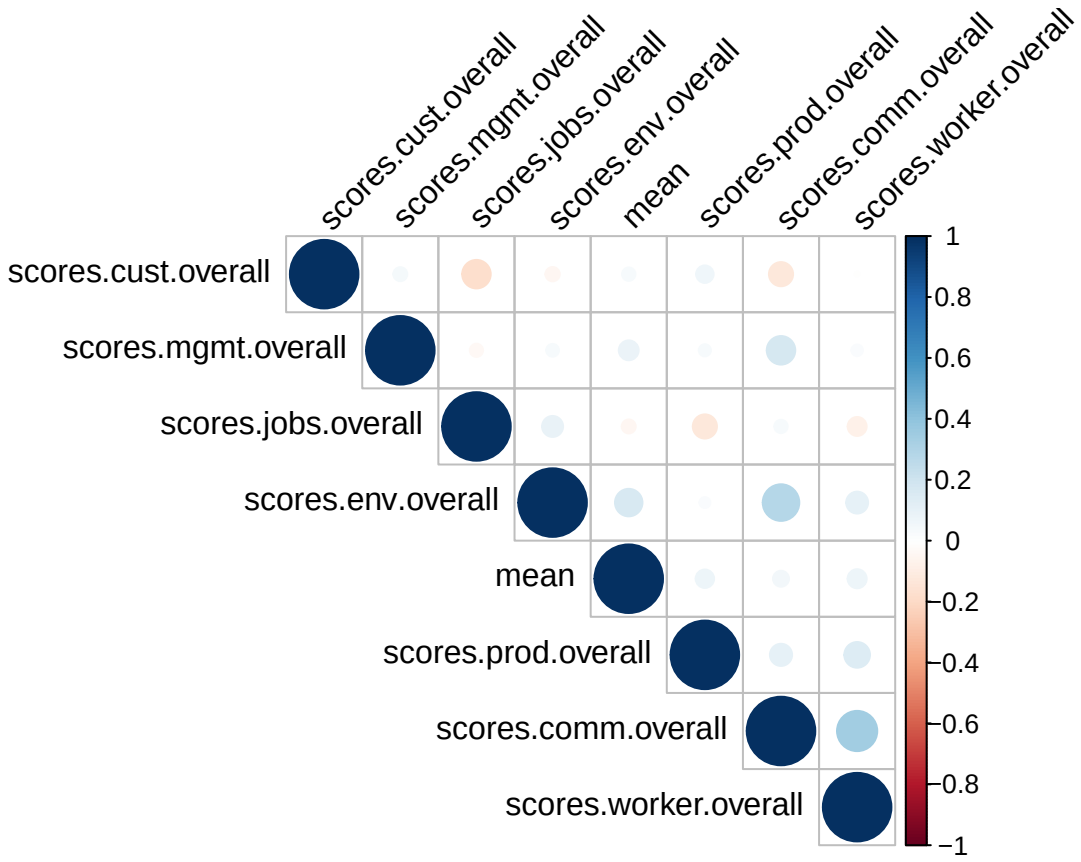
```
summary.function <- function(data, byvar=NULL,ordervar="Average_Score") {
  out<-data[,list(Average_Score=mean(scores.overall,na.rm=TRUE),
                  Variance_Score=var(scores.overall,na.rm=TRUE),
                  Total_Companies=.N),by=byvar]
  setorderv(out, ordervar)
  return(out)
}

summary.function(alldata,byvar="rank.industry")
```

```
##     rank.industry Average_Score Variance_Score Total_Companies
##  1:            40      46.18078       2.921904               3
##  2:            42      46.23802       1.423630               2
##  3:            41      46.42858       2.075616               2
##  4:            44      46.53371             NA               1
##  5:            39      46.64202       2.579078               3
##  6:            43      46.65339             NA               1
##  7:            36      46.82059       2.391955               6
##  8:            35      46.95235       1.827737               5
##  9:            37      47.00445       2.074978               4
## 10:            38      47.01700       1.613210               3
## 11:            31      47.05568       1.374235               6
## 12:            34      47.40851       1.372679               6
## 13:            24      47.41804       5.579055              20
## 14:            32      47.61403       2.292664               6
```

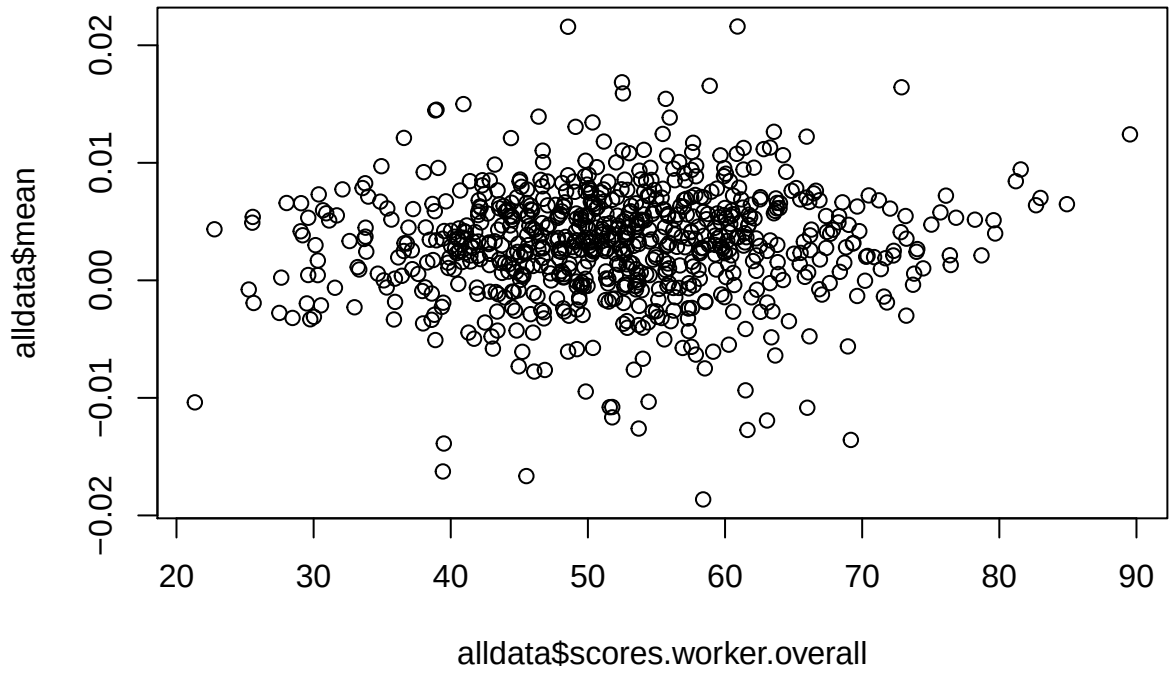
```
## 15:              33      47.69878       1.582248              6
## 16:              25      47.72498       5.411311             16
## 17:              30      47.73368       3.119934              8
## 18:              23      47.79685       4.547563             21
## 19:              22      48.11397       4.125735             19
## 20:              29      48.14690       2.862440              9
## 21:              26      48.15004       4.410751             13
## 22:              28      48.24964       2.983054             10
## 23:              27      48.35340       3.455388             11
## 24:              21      48.53750       3.938239             18
## 25:              20      49.06504       4.428496             20
## 26:              18      49.40874       3.907501             23
## 27:              19      49.49910       3.764446             20
## 28:              15      49.75833       5.689554             25
## 29:              17      49.85269       4.165848             21
## 30:              16      50.04037       4.724302             22
## 31:              14      50.32511       5.234150             26
## 32:              13      51.05559       4.892193             24
## 33:              12      51.33537       5.262717             27
## 34:              11      51.51788       5.669276             28
## 35:              10      51.57490       9.329316             27
## 36:               9      51.65807       8.778486             30
## 37:               8      52.15532      10.256412             33
## 38:               7      52.78966       9.460078             33
## 39:               6      53.66746       9.123718             33
## 40:               5      54.72917       8.757702             31
## 41:               4      55.71571       9.512196             33
## 42:               3      56.80687      12.527280             33
## 43:               2      58.03637      12.896545             31
## 44:               1      60.36683      20.238127             32
##     rank.industry Average_Score Variance_Score Total_Companies
```

Correlation Matrix of Stock price with scores

```
#cor(alldata$scores.worker.overall,alldata$mean,use="complete.obs")
#cor(alldata[,6:52],alldata$mean,use="complete.obs")
res<-cor(alldata[,c(7:13,53)])
corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```




```
plot(alldata$scores.worker.overall,alldata$mean)##The Correlation seems weak with respect to mean
```



Random Forest By Industry (Response as overall score)

```
rflist<-lapply(split(alldata,alldata$rank.industry),function(d) randomForest(scores.overall~.,d[,c(6,14

#varimp<-varImpPlot(rflist$`1`)
#varImpPlot(rflist$)
#rownames(varimp1)[apply(varimp1, 2, which.max)]
#rownames(varimp1)[order(varimp1, decreasing=TRUE)][1:3]
varimp<-function(fit){
  plot<-as.data.frame(importance(fit))
  return(rownames(plot)[order(plot$IncNodePurity, decreasing=TRUE)][1:3])
}
#names(rflist)<-c(1:32)
temp<- c(varimp(rflist$`1`),varimp(rflist$`2`),varimp(rflist$`3`),varimp(rflist$`4`),varimp(rflist$`5`)
         ,varimp(rflist$`33`),varimp(rflist$`34`))
temp<-as.data.frame(temp)
summary(temp$temp)
```

```
##      scores.comm.abuse    scores.comm.charity   scores.comm.conflict
##                      2                      1                      2
##       scores.comm.rels        scores.cust.exp   scores.env.efficient
##                      1                      2                      6
##        scores.env.mgmt   scores.env.pollution     scores.jobs.growth
##                      3                      1                      3
##       scores.jobs.size  scores.mgmt.integrity     scores.mgmt.profit
##                      5                      1                      5
##     scores.mgmt.return        scores.prod.ben  scores.worker.balance
##                      3                      4                      9
##   scores.worker.career      scores.worker.ceo  scores.worker.fairpay
##                      3                      5                      4
##   scores.worker.health scores.worker.hiredisc   scores.worker.living
##                      6                      1                     10
##     scores.worker.open      scores.worker.pto   scores.worker.retire
##                      5                      1                     16
##     scores.worker.safe
##                      3
```

Random Forest By Industry (Response as stock mean)

```
rflist<-lapply(split(alldata,alldata$rank.industry),function(d) randomForest(mean~.,d[,c(14:53)],importa

#varimp<-varImpPlot(rflist$`1`)
#varImpPlot(rflist$)
#rownames(varimp1)[apply(varimp1, 2, which.max)]
#rownames(varimp1)[order(varimp1, decreasing=TRUE)][1:3]
varimp<-function(fit){
  plot<-as.data.frame(importance(fit))
  return(rownames(plot)[order(plot$IncNodePurity, decreasing=TRUE)][1:3])
}
#names(rflist)<-c(1:32)
temp<- c(varimp(rflist$`1`),varimp(rflist$`2`),varimp(rflist$`3`),varimp(rflist$`4`),varimp(rflist$`5`)
```

```
temp<-as.data.frame(temp)
summary(temp$temp)
```

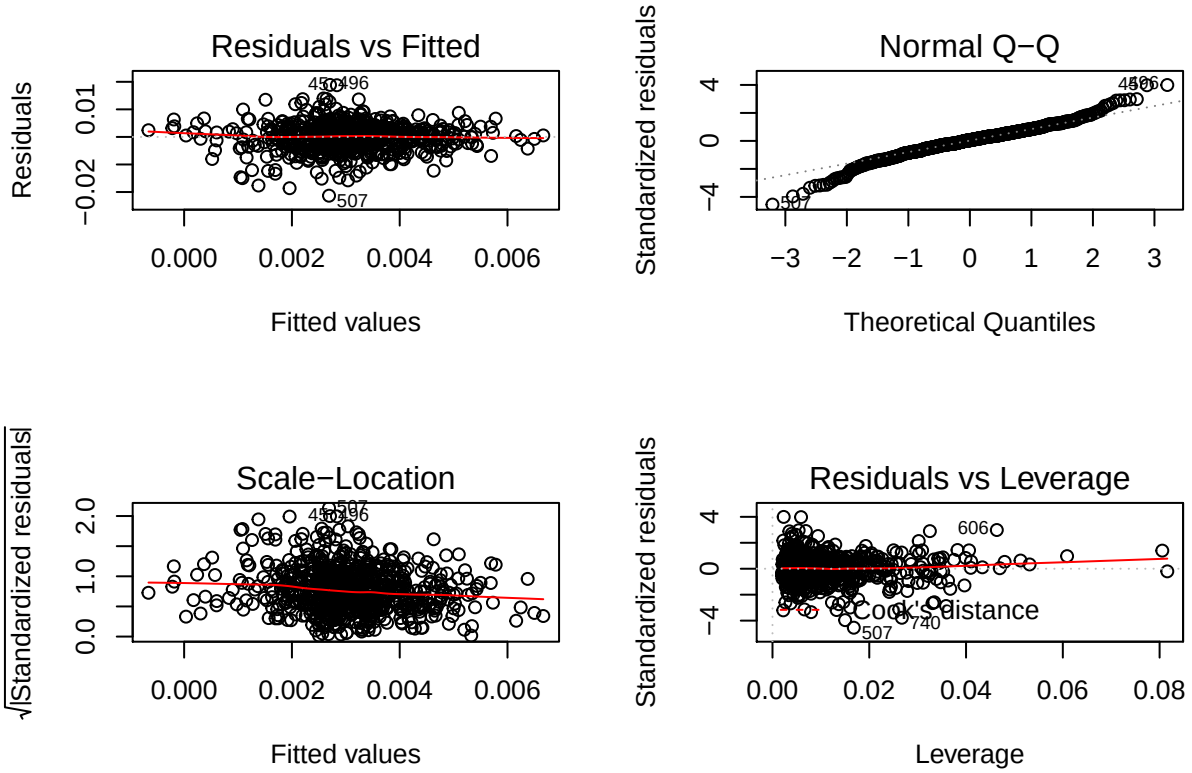
```
##      scores.comm.abuse    scores.comm.charity   scores.comm.conflict
##                      1                      1                      3
##        scores.cust.exp   scores.env.efficient        scores.env.mgmt
##                      1                      4                      3
##   scores.env.pollution     scores.jobs.growth       scores.jobs.size
##                      3                      9                      9
##  scores.mgmt.integrity       scores.mgmt.laws     scores.mgmt.profit
##                      3                      2                     12
##     scores.mgmt.return        scores.mgmt.tax  scores.worker.balance
##                     15                      2                      4
##   scores.worker.career      scores.worker.ceo  scores.worker.fairpay
##                      4                      3                      3
##   scores.worker.health scores.worker.hiredisc   scores.worker.living
##                      4                      1                      1
##     scores.worker.open      scores.worker.pto   scores.worker.retire
##                      7                      3                      3
##     scores.worker.safe
##                      1
```

Full Data

```
data_noind<-alldata[,6:53]
lmfit_main_feature<-lm(mean~.,data_noind[,c(2:8,48)])
summary(lmfit_main_feature)
```

```
##
## Call:
## lm(formula = mean ~ ., data = data_noind[, c(2:8, 48)])
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0213268 -0.0025065  0.0002472  0.0027261  0.0188715
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -5.495e-03  2.681e-03  -2.050   0.0407 *
## scores.comm.overall   -1.064e-05  1.683e-05  -0.632   0.5274
## scores.cust.overall    2.474e-05  3.137e-05   0.789   0.4305
## scores.env.overall     6.860e-05  1.584e-05   4.331 1.69e-05 ***
## scores.jobs.overall   -1.888e-05  1.890e-05  -0.999   0.3182
## scores.mgmt.overall    4.187e-05  2.001e-05   2.093   0.0367 *
## scores.prod.overall    3.223e-05  2.191e-05   1.471   0.1417
## scores.worker.overall  2.575e-05  1.745e-05   1.476   0.1404
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.004731 on 743 degrees of freedom
## Multiple R-squared:  0.04209,    Adjusted R-squared:  0.03306
## F-statistic: 4.664 on 7 and 743 DF,  p-value: 3.967e-05
```

```r
par( mfrow = c( 2, 2 ) )
plot(lmfit_main_feature)
```




```r
step_lm<-stepAIC(lmfit_main_feature,direction = "both",trace = FALSE)
summary(step_lm)
```

```
##
## Call:
## lm(formula = mean ~ scores.env.overall + scores.mgmt.overall +
##     scores.prod.overall, data = data_noind[, c(2:8, 48)])
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0213252 -0.0025382  0.0003077  0.0027341  0.0192560
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -4.581e-03  1.665e-03  -2.751  0.00609 **
## scores.env.overall    6.585e-05  1.514e-05   4.348 1.56e-05 ***
## scores.mgmt.overall   4.163e-05  1.966e-05   2.117  0.03455 *
## scores.prod.overall   3.937e-05  2.149e-05   1.832  0.06731 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.004733 on 747 degrees of freedom
## Multiple R-squared:  0.03622,    Adjusted R-squared:  0.03235
## F-statistic: 9.358 on 3 and 747 DF,  p-value: 4.456e-06
```

```r
lmfit_sub_feature<-lm(mean~.,data_noind[,c(9:48)])
summary(lmfit_sub_feature)
```

```
##
## Call:
## lm(formula = mean ~ ., data = data_noind[, c(9:48)])
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0187441 -0.0024035 -0.0000049  0.0026406  0.0187010
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -9.096e-03  4.420e-03  -2.058  0.03999 *
## scores.comm.abuse        1.908e-05  9.605e-06   1.987  0.04736 *
## scores.comm.charity     -9.230e-06  9.268e-06  -0.996  0.31965
## scores.comm.conflict     1.035e-05  9.505e-06   1.089  0.27668
## scores.comm.local        1.806e-06  7.159e-06   0.252  0.80087
## scores.comm.oppress      1.025e-05  2.930e-05   0.350  0.72652
## scores.comm.rels        -5.557e-06  1.011e-05  -0.549  0.58288
## scores.cust.disc        -4.245e-05  2.712e-05  -1.565  0.11796
## scores.cust.exp          5.085e-06  8.752e-06   0.581  0.56144
## scores.cust.fair        -2.130e-05  2.587e-05  -0.823  0.41070
## scores.cust.label       -2.932e-06  2.970e-05  -0.099  0.92140
## scores.cust.priv        -1.360e-06  9.852e-06  -0.138  0.89020
## scores.cust.truth        8.349e-05  3.466e-05   2.409  0.01626 *
## scores.env.efficient    -2.562e-06  1.092e-05  -0.235  0.81454
## scores.env.mgmt          3.193e-05  1.209e-05   2.641  0.00844 **
## scores.env.pollution     2.495e-05  1.287e-05   1.939  0.05288 .
## scores.jobs.growth      -1.269e-05  1.381e-05  -0.919  0.35816
## scores.jobs.size        -6.508e-06  1.585e-05  -0.411  0.68145
## scores.mgmt.integrity   -8.348e-06  7.644e-06  -1.092  0.27516
## scores.mgmt.laws         1.647e-05  1.227e-05   1.342  0.17989
## scores.mgmt.profit       1.435e-05  8.952e-06   1.603  0.10935
## scores.mgmt.reporting   -9.945e-06  1.081e-05  -0.920  0.35790
## scores.mgmt.return       5.938e-05  8.345e-06   7.115 2.73e-12 ***
## scores.mgmt.tax          8.604e-06  7.511e-06   1.146  0.25237
## scores.prod.ben         -7.390e-06  1.102e-05  -0.670  0.50282
## scores.prod.price        8.541e-07  1.696e-05   0.050  0.95985
## scores.prod.qual         3.350e-05  1.269e-05   2.639  0.00849 **
## scores.worker.balance   -1.557e-05  9.726e-06  -1.601  0.10985
## scores.worker.career     7.717e-06  1.073e-05   0.719  0.47224
## scores.worker.ceo        2.827e-05  1.116e-05   2.534  0.01150 *
## scores.worker.fairpay    4.237e-06  9.938e-06   0.426  0.67000
## scores.worker.health     3.355e-05  1.309e-05   2.563  0.01057 *
## scores.worker.hiredisc  -6.795e-06  1.219e-05  -0.558  0.57731
## scores.worker.layoff     9.798e-07  3.650e-05   0.027  0.97859
## scores.worker.living    -1.811e-06  1.019e-05  -0.178  0.85902
## scores.worker.open      -2.145e-05  1.020e-05  -2.102  0.03587 *
## scores.worker.paydisc    1.458e-05  9.099e-06   1.602  0.10952
## scores.worker.pto        3.394e-06  7.596e-06   0.447  0.65508
## scores.worker.retire    -6.553e-06  9.759e-06  -0.671  0.50217
## scores.worker.safe       4.447e-06  1.070e-05   0.416  0.67769
## ---
```

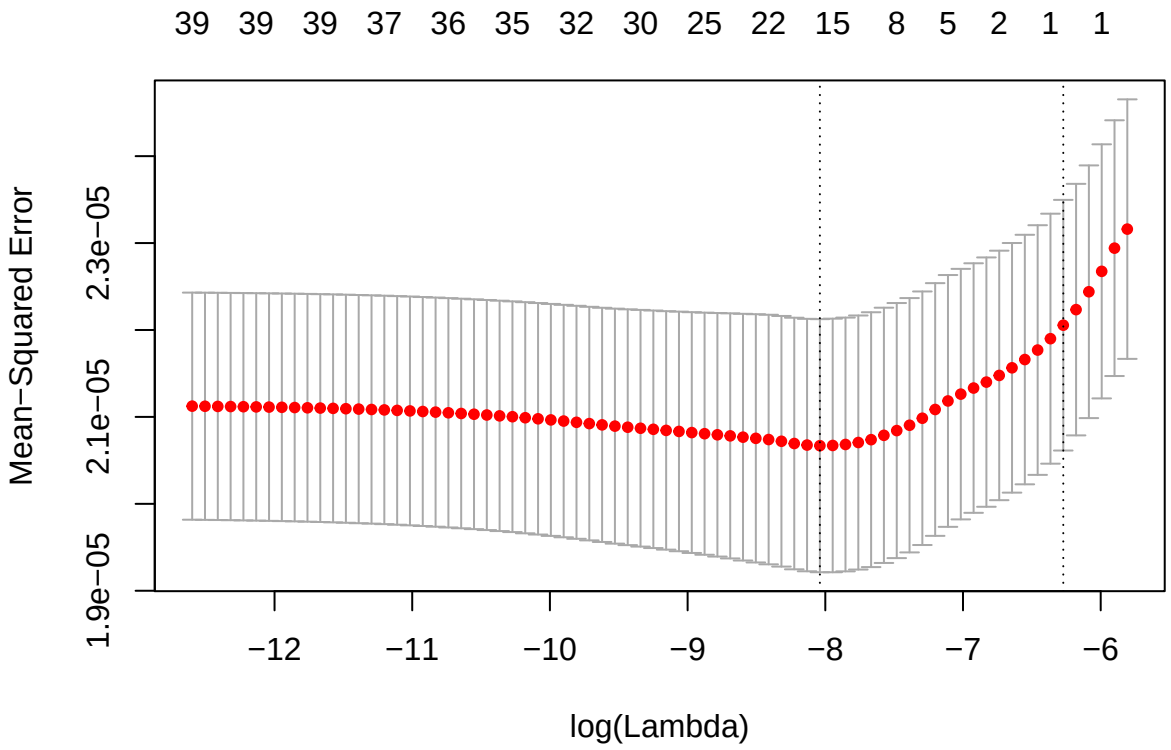
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.004461 on 711 degrees of freedom
## Multiple R-squared:  0.1849, Adjusted R-squared:  0.1402
## F-statistic: 4.136 on 39 and 711 DF,  p-value: 8.88e-15
```

```
step_lm<-stepAIC(lmfit_sub_feature,direction = "both",trace=FALSE)
summary(step_lm)
```

```
##
## Call:
## lm(formula = mean ~ scores.comm.abuse + scores.cust.disc + scores.cust.truth +
##     scores.env.mgmt + scores.env.pollution + scores.mgmt.laws +
##     scores.mgmt.profit + scores.mgmt.return + scores.prod.qual +
##     scores.worker.balance + scores.worker.ceo + scores.worker.health +
##     scores.worker.open, data = data_noind[, c(9:48)])
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0192869 -0.0024556  0.0001047  0.0025252  0.0184878
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.162e-02  2.547e-03  -4.564 5.89e-06 ***
## scores.comm.abuse      1.825e-05  8.113e-06   2.250 0.024772 *
## scores.cust.disc      -3.481e-05  2.444e-05  -1.424 0.154858
## scores.cust.truth      8.836e-05  3.393e-05   2.604 0.009393 **
## scores.env.mgmt        2.782e-05  1.073e-05   2.594 0.009682 **
## scores.env.pollution   2.856e-05  1.074e-05   2.658 0.008022 **
## scores.mgmt.laws       1.645e-05  1.128e-05   1.459 0.145115
## scores.mgmt.profit     1.777e-05  8.528e-06   2.084 0.037549 *
## scores.mgmt.return     5.639e-05  7.958e-06   7.085 3.24e-12 ***
## scores.prod.qual       3.649e-05  1.173e-05   3.109 0.001947 **
## scores.worker.balance -1.618e-05  8.442e-06  -1.916 0.055729 .
## scores.worker.ceo      2.748e-05  8.523e-06   3.224 0.001318 **
## scores.worker.health   3.863e-05  1.060e-05   3.645 0.000287 ***
## scores.worker.open    -2.119e-05  9.944e-06  -2.131 0.033448 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.004424 on 737 degrees of freedom
## Multiple R-squared:  0.1693, Adjusted R-squared:  0.1547
## F-statistic: 11.56 on 13 and 737 DF,  p-value: < 2.2e-16
```

Lasso

```
cvglmout <- cv.glmnet(as.matrix(data_noind[,9:47]), as.matrix(data_noind[,48]),alpha=0.5)
par( mfrow = c( 1, 1 ) )
plot(cvglmout)
```




```
optlambda=cvglmout$lambda.1se
lassofit=glmnet(as.matrix(data_noind[,9:47]), as.matrix(data_noind[,48]),alpha = 0.5,lambda = optlambda)
lassofit$beta
```

```
## 39 x 1 sparse Matrix of class "dgCMatrix"
##                                  s0
## scores.comm.abuse      .
## scores.comm.charity    .
## scores.comm.conflict   .
## scores.comm.local      .
## scores.comm.oppress    .
## scores.comm.rels       .
## scores.cust.disc       .
## scores.cust.exp        .
## scores.cust.fair       .
## scores.cust.label      .
## scores.cust.priv       .
## scores.cust.truth      .
## scores.env.efficient   .
## scores.env.mgmt        .
## scores.env.pollution   .
## scores.jobs.growth     .
## scores.jobs.size       .
## scores.mgmt.integrity  .
## scores.mgmt.laws       .
## scores.mgmt.profit     .
## scores.mgmt.reporting  .
## scores.mgmt.return     2.035642e-05
## scores.mgmt.tax        .
## scores.prod.ben        .
## scores.prod.price      .
```

```
## scores.prod.qual       .
## scores.worker.balance  .
## scores.worker.career   .
## scores.worker.ceo      .
## scores.worker.fairpay  .
## scores.worker.health   .
## scores.worker.hiredisc .
## scores.worker.layoff   .
## scores.worker.living   .
## scores.worker.open     .
## scores.worker.paydisc  .
## scores.worker.pto      .
## scores.worker.retire   .
## scores.worker.safe     .
```