Problem Set 4

Yaoxi Shi

8 November 2019

Factor Analysis

1.How do CFA and EFA differ?

EFA is used to explore how many latent factors could be extracted to explain the covariance of the variables, it's atheorrtical and there is no specific priori hypothesis before the analysis. Also, the factor structure of the data is informed by the results of the analysis.

CFA is hypothesis-based, it's about using the factor analysis to confirm whether the pre-established theory is aligned with the reseults, for example, whether the number of the factors and the loading comfirms the theory.

2.Fit three exploratory factor analysis models initialized at 2, 3, and 4 factors. Present the loadings from these solutions and discuss in substantive terms. How does each fit? What sense does this give you of the underlying dimensionality of the space? And so on.

```
dataraw=read.csv("~/Uchicago/courses/Unsupervised machine learning/Problem-Set-4/Problem-Set-4-master/c
data=dataraw %>% mutate_at(.vars = vars(-X), scale)
factorana1=fa(data[,-1], nfactors = 2)
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
factorana1$loadings
```

```
##
## Loadings:
##             MR1    MR2
## idealpoint   0.449  0.429
## polity       0.995
## polity2      0.995
## democ        0.931
## autoc       -0.969  0.159
## unreg        0.412 -0.131
## physint             0.782
## speech       0.631  0.154
## new_empinx   0.802  0.197
## wecon               0.509
## wopol        0.551
## wosoc        0.286  0.497
## elecsd       0.852
## gdp.pc.wdi          0.673
## gdp.pc.un           0.671
## pop.wdi      0.204 -0.476
## amnesty            -0.821
## statedept          -0.849
## milper       0.158 -0.468
## cinc         0.211 -0.366
## domestic9    0.288 -0.479
##
##                  MR1   MR2
```

```
## SS loadings    6.523 4.527
## Proportion Var 0.311 0.216
## Cumulative Var 0.311 0.526
```

```
factorana2=fa(data[,-1], nfactors = 3)
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
factorana2$loadings
```

```
##
## Loadings:
##            MR1    MR2    MR3
## idealpoint  0.432  0.468
## polity      0.992
## polity2     0.992
## democ       0.910  0.144
## autoc      -0.994  0.191
## unreg       0.413 -0.129
## physint            0.737 -0.136
## speech      0.646  0.128
## new_empinx  0.840  0.131 -0.125
## wecon              0.518
## wopol       0.552
## wosoc       0.263  0.547
## elecsd      0.858
## gdp.pc.wdi         0.856  0.158
## gdp.pc.un          0.853  0.157
## pop.wdi                   0.892
## amnesty           -0.715  0.243
## statedept         -0.803  0.144
## milper                    0.949
## cinc                      0.999
## domestic9   0.269 -0.443
##
##                  MR1   MR2   MR3
## SS loadings    6.466 4.275 2.881
## Proportion Var 0.308 0.204 0.137
## Cumulative Var 0.308 0.512 0.649
```

```
factorana3=fa(data[,-1], nfactors = 4)
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
factorana3$loadings
```

```
##
## Loadings:
##            MR1    MR3    MR4    MR2
## idealpoint  0.467         0.214 -0.294
## polity      0.995
## polity2     0.995
## democ       0.922         0.127
## autoc      -0.986         0.146
## unreg       0.405                0.165
## physint     0.119               -0.761
## speech      0.658               -0.109
```

```
## new_empinx  0.855               -0.145
## wecon       0.105         0.390 -0.170
## wopol       0.555
## wosoc       0.300         0.350 -0.239
## elecsd      0.865
## gdp.pc.wdi                0.986
## gdp.pc.un                 0.979
## pop.wdi            0.923
## amnesty            0.177 -0.197  0.602
## statedept  -0.137        -0.139  0.783
## milper             0.965
## cinc               0.981  0.111
## domestic9   0.247         0.204  0.757
##
##                   MR1   MR3   MR4   MR2
## SS loadings     6.605 2.811 2.426 2.370
## Proportion Var 0.315 0.134 0.116 0.113
## Cumulative Var 0.315 0.448 0.564 0.677
```

The 2-factor model could explain 52.6% of the total variance, with first factor explaing 31.1% and the second one 21.6%. The fit overall looks not very ideal. The 3-fator model explains more variance, which is 64.9%, with the first factor explains 30.8%, the second 20.4% and the third factor 13.7%, there are some overlaps among loadings of the variables of the factors, but according to the loading values, the three factors look not highly correlated, the results look pretty good. As for the 4-factor model, the total variance explained by the model (67.7%) doesn't increase much compared to the 3-factor model, also, the 2nd, 3rd and 4th factor all explain similar amount of variance, they could be combined together. Also, we could find that the loadings have many overlaps. Therefore, overall 3-factor model is the best among these three, suggesting that there are 3 potential underlying dimensions in the space.

3.Rotate the 3-factor solution using any oblique method you would like and present a visual of the unrotated and rotated versions side-by-side. How do these differ and why does this matter (or not)?

```
r.factor=fa(data[,-1],nfactors = 3, rotate = "oblimin")
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
nonr.factors=fa(data[,-1],nfactors = 3, rotate = "none")
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
rot.pattern=as.data.frame(r.factor$loadings[1:21,])
nonr.pattern = as.data.frame(nonr.factors$loadings[1:21,])

nonrot1=xyplot(MR2 ~ MR1, data = nonr.pattern,
       aspect = 1,
       xlim = c(-.8, 1.2),
       ylim = c(-.5, .8),
       panel = function (x, y) {
         panel.segments(c(0, 0), c(0, 0),
             c(1, 0), c(0, 1), col = "gray")
         panel.text(1, 0, labels = "Initial\n(unrotated)\nfactor 1",
                    cex = .65, pos = 3, col = "gray")
         panel.text(0, .7, labels = "Initial\n(unrotated)\nfactor 2",
                    cex = .65, pos = 4, col = "gray")
         panel.segments(rep(0, 8), rep(0, 8), x, y,
            col = "black")
```

```
          panel.text(x[-20], y[-20], labels = rownames(nonrot.pattern)[-20],
             pos = 4, cex = .5)
          panel.text(x[20], y[20], labels = rownames(nonrot.pattern)[20],
                     pos = 3, cex = .5)
        },
        main = "Unrotated Factor Pattern",
        xlab = "",
        ylab = "",
)
nonrot2=xyplot(MR3 ~ MR1, data = nonr.pattern,
        aspect = 1,
        xlim = c(-.8, 1.2),
        ylim = c(-.5, .8),
        panel = function (x, y) {
          panel.segments(c(0, 0), c(0, 0),
              c(1, 0), c(0, 1), col = "gray")
          panel.text(1, 0, labels = "Initial\n(unrotated)\nfactor 1",
                     cex = .65, pos = 3, col = "gray")
          panel.text(0, .7, labels = "Initial\n(unrotated)\nfactor 2",
                     cex = .65, pos = 4, col = "gray")
          panel.segments(rep(0, 8), rep(0, 8), x, y,
             col = "black")
          panel.text(x[-20], y[-20], labels = rownames(nonrot.pattern)[-20],
             pos = 4, cex = .5)
          panel.text(x[20], y[20], labels = rownames(nonrot.pattern)[20],
                     pos = 1, cex = .5)
        },
        main = "Unrotated Factor Pattern",
        xlab = "",
        ylab = "",
)

nonrot3=xyplot(MR3 ~ MR2, data = nonr.pattern,
        aspect = 1,
        xlim = c(-.5, .8),
        ylim = c(-.5, .8),
        panel = function (x, y) {
          panel.segments(c(0, 0), c(0, 0),
              c(1, 0), c(0, 1), col = "gray")
          panel.text(1, 0, labels = "Initial\n(unrotated)\nfactor 1",
                     cex = .65, pos = 3, col = "gray")
          panel.text(0, .7, labels = "Initial\n(unrotated)\nfactor 2",
                     cex = .65, pos = 4, col = "gray")
          panel.segments(rep(0, 8), rep(0, 8), x, y,
             col = "black")
          panel.text(x[-20], y[-20], labels = rownames(nonrot.pattern)[-20],
             pos = 4, cex = .5)
          panel.text(x[20], y[20], labels = rownames(nonrot.pattern)[20],
                     pos = 1, cex = .5)
        },
        main = "Unrotated Factor Pattern",
        xlab = "",
        ylab = "",
```

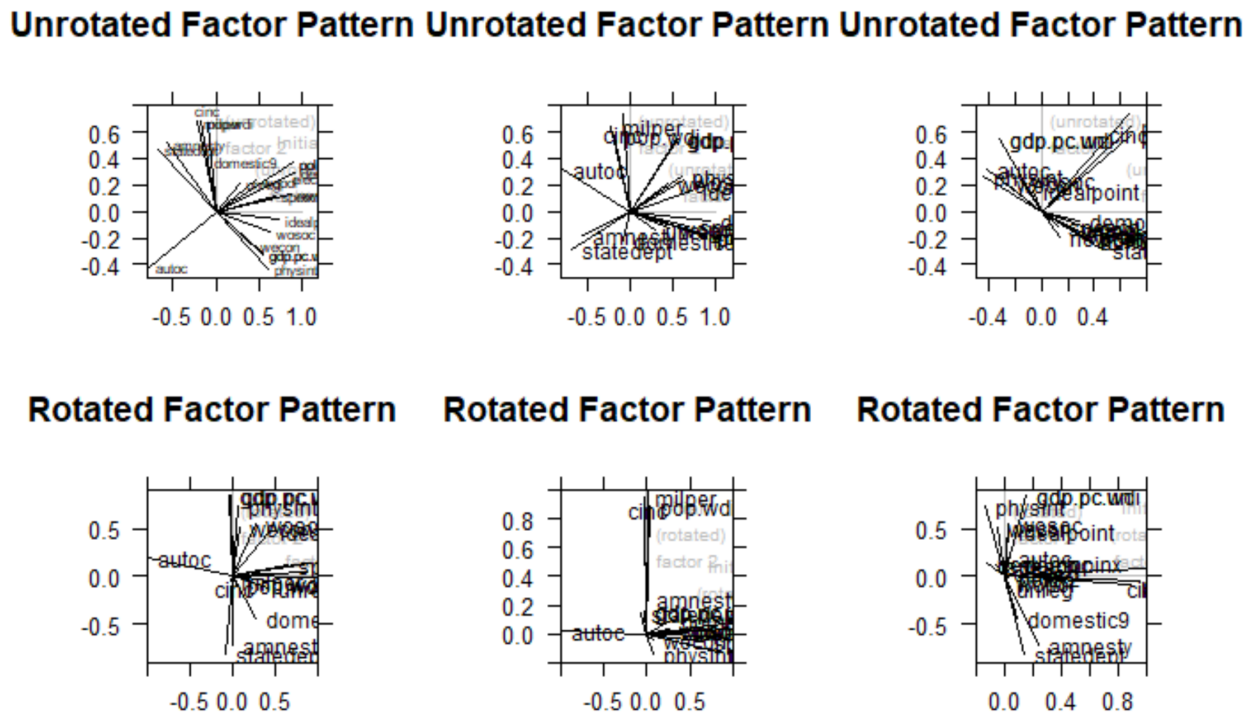
```
)


rot1=xyplot(MR2 ~ MR1, data = rot.pattern,
       aspect = 1,
       xlim = c(-1, 1),
       ylim = c(-.9, .9),
       panel = function (x, y) {
         panel.segments(c(0, 0), c(0, 0),
             c(1, 0), c(0, 1), col = "gray")
         panel.text(1, 0, labels = "Initial\n(rotated)\nfactor 1",
                    cex = .65, pos = 3, col = "gray")
         panel.text(0, .7, labels = "Initial\n(rotated)\nfactor 2",
                    cex = .65, pos = 4, col = "gray")
         panel.segments(rep(0, 8), rep(0, 8), x, y,
            col = "black")
         panel.text(x[-20], y[-20], labels = rownames(nonrot.pattern)[-20],
            pos = 4, cex = .5)
         panel.text(x[20], y[20], labels = rownames(nonrot.pattern)[20],
                    pos = 1, cex = .5)
       },
       main = "Rotated Factor Pattern",
       xlab = "",
       ylab = "",
)

rot2=xyplot(MR3 ~ MR1, data = rot.pattern,
       aspect = 1,
       xlim = c(-1, 1),
       ylim = c(-.2, 1),
       panel = function (x, y) {
         panel.segments(c(0, 0), c(0, 0),
             c(1, 0), c(0, 1), col = "gray")
         panel.text(1, 0, labels = "Initial\n(rotated)\nfactor 1",
                    cex = .65, pos = 3, col = "gray")
         panel.text(0, .7, labels = "Initial\n(rotated)\nfactor 2",
                    cex = .65, pos = 4, col = "gray")
         panel.segments(rep(0, 8), rep(0, 8), x, y,
            col = "black")
         panel.text(x[-20], y[-20], labels = rownames(nonrot.pattern)[-20],
            pos = 4, cex = .5)
         panel.text(x[20], y[20], labels = rownames(nonrot.pattern)[20],
                    pos = 1, cex = .5)
       },
       main = "Rotated Factor Pattern",
       xlab = "",
       ylab = "",
)

rot3=xyplot(MR2 ~ MR3, data = rot.pattern,
       aspect = 1,
       xlim = c(-.2, 1),
       ylim = c(-.9, .9),
```

```
        panel = function (x, y) {
          panel.segments(c(0, 0), c(0, 0),
              c(1, 0), c(0, 1), col = "gray")
          panel.text(1, 0, labels = "Initial\n(rotated)\nfactor 1",
                     cex = .65, pos = 3, col = "gray")
          panel.text(0, .7, labels = "Initial\n(rotated)\nfactor 2",
                     cex = .65, pos = 4, col = "gray")
          panel.segments(rep(0, 8), rep(0, 8), x, y,
             col = "black")
          panel.text(x[-20], y[-20], labels = rownames(nonrot.pattern)[-20],
             pos = 4, cex = .5)
          panel.text(x[20], y[20], labels = rownames(nonrot.pattern)[20],
                     pos = 1, cex = .5)
        },
        main = "Rotated Factor Pattern",
        xlab = "",
        ylab = "",
)
```



By using oblimin methods, the structure of the factors axes looks more clear and simple than the unrotated versions. Rotation could improve the interpretability of the results.

Principal Components Analysis

1.What is the statistical difference between PCA and FA? Describe the basic construction of each approach using equations and then point to differences that exist across these two widely used methods for reducing dimensionality.

FA: $X_1 = b_1F + d_1U_1$ PCA: $F_1 = L_1X_1 + L_2X_2 + ... + L_kX_k$

In factor analysis, observed variables are linear combinations of the underlying and unique factors, and the

factors are linear combinations that maximize the shared portion of the variance, which is the underlying "latent constructs". The results of the analysis are the assumed factors that cause the observed variabes. Also, we assume the latent variables in FA are normally distributed.

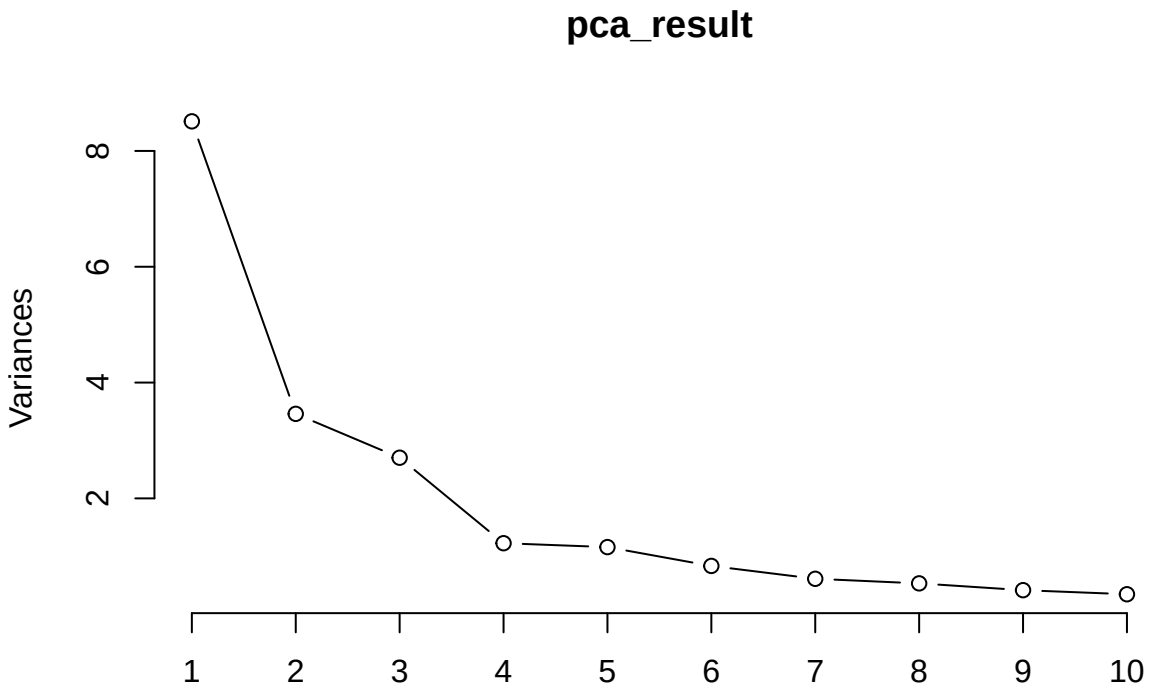
For PCA, components are a linear combination of the observed variables weighted by eigenvectors, which maximize the total variance. And is doesn't impose any distributioal assumptions on the latent factors.

2.Fit a PCA model. Present the proportion of explained variance across the first 10 components. What do these values tell you substantively (e.g., how many components likely characterize these data?)?

```
data2=data[,-1]
row.names(data2)=data$X
pca_result=prcomp(data2,scale. = T)
summary(pca_result)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     2.9173 1.8600 1.6439 1.10713 1.07631 0.91289
## Proportion of Variance 0.4053 0.1648 0.1287 0.05837 0.05516 0.03968
## Cumulative Proportion  0.4053 0.5700 0.6987 0.75708 0.81225 0.85193
##                           PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     0.78181 0.72948 0.64421 0.58703 0.55164 0.49341
## Proportion of Variance 0.02911 0.02534 0.01976 0.01641 0.01449 0.01159
## Cumulative Proportion  0.88104 0.90638 0.92614 0.94255 0.95704 0.96864
##                           PC13   PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.46337 0.3995 0.32765 0.29011 0.24347 0.18215
## Proportion of Variance 0.01022 0.0076 0.00511 0.00401 0.00282 0.00158
## Cumulative Proportion  0.97886 0.9865 0.99157 0.99558 0.99840 0.99998
##                           PC19      PC20      PC21
## Standard deviation     0.01990 5.378e-16 2.786e-16
## Proportion of Variance 0.00002 0.000e+00 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00 1.000e+00
```

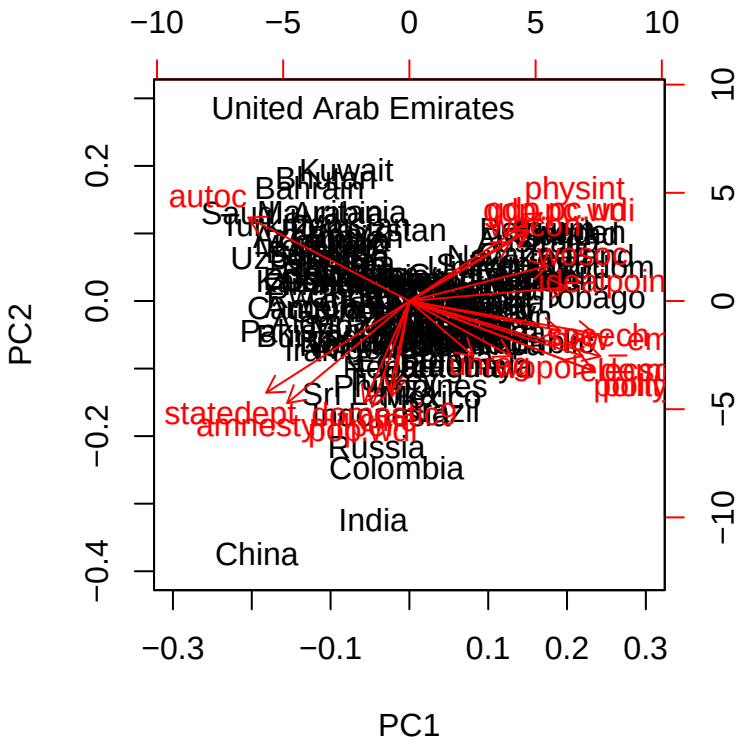
```
plot(pca_result, type="l")
```



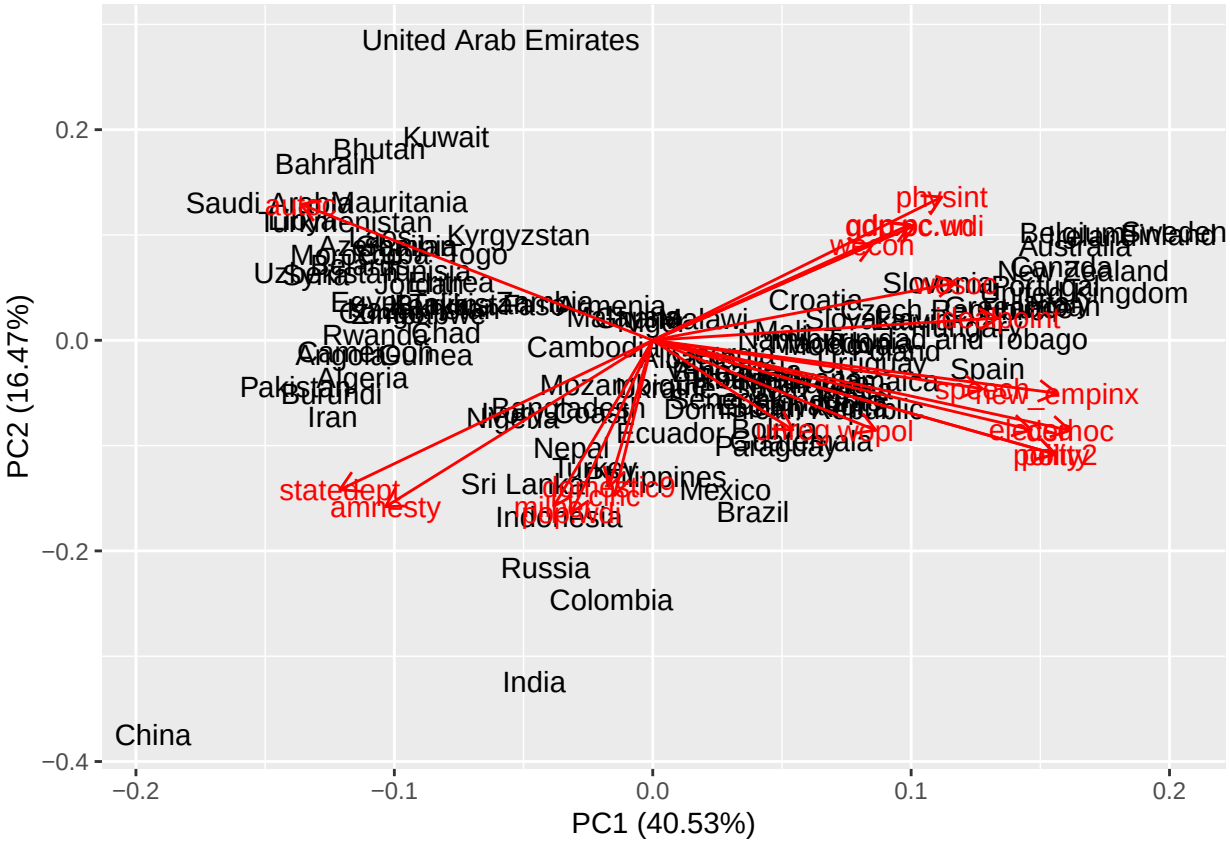
From the outcomes of the PCA, we find that the first componant could explain 40.53% of the variance, and there are 3 principle componants that explain more than 10% of the variance, the fourth one could only explain 5.8%. The cumulative proportion of the variance that the first three componants explains is 69.87%, and the following componants could ony add few to it, suggesting that the first three componants could charaterize these variabels.

3.Present a biplot of the PCA fit from the previous question. Describe what you see (e.g., which countries are clustered together? Which input features are doing the bulk of the explaining? How do you know this?

```
biplot(pca_result, expand=1.5, xlim=c(-.3, .3), ylim=c(-.4, .3))
```




```
autoplot(pca_result, shape=F, loadings.label = T)
```



From the plot, we could find that there are roughly two clusters of countries, one cluster on the left upper side includes countries in mideast, north Africa and east Europe, such as Kuwait, Moroco, Iran, Pakistan etc, another cluster of coutries in the right side includes Australia, UK, Spain, Nepal, Brazil, these are mainly north and west European, Asian, north and south American countries. Features like "polity", "polity2", "democ", "autoc","new_empinx" have high absolut loading values on PC1, and features like "pop.wid", "amnesty", "statedept" have high absolut loadings on PC2, suggesting they explains higher variances among the whole feature space.

Bonus Question:

1.Fit a sparse PCA model and a probabilistic PCA model. Compare these results substantively. What does each tell you and why do these distinctions matter in terms of inference (or not)?

```
# Sparse PCA
spca=spca(data2, k=3)
```

```
## [1] "Iteration:    1, Objective: 3.36548e+02, Relative improvement Inf"
```

```
spca
```

```
## Standard deviations:
## [1] 2.917 1.860 1.644
##
## Eigenvalues:
## [1] 8.507 3.458 2.701
##
## Sparse loadings:
##         [,1]   [,2]   [,3]
```

```
##  [1,]  0.259  0.041 -0.095
##  [2,]  0.302 -0.210  0.080
##  [3,]  0.302 -0.210  0.080
##  [4,]  0.313 -0.166  0.017
##  [5,] -0.264  0.250 -0.155
##  [6,]  0.105 -0.165  0.091
##  [7,]  0.217  0.264 -0.114
##  [8,]  0.247 -0.089  0.071
##  [9,]  0.304 -0.097  0.116
## [10,]  0.165  0.176 -0.126
## [11,]  0.167 -0.166  0.071
## [12,]  0.227  0.105 -0.144
## [13,]  0.284 -0.165  0.083
## [14,]  0.196  0.211 -0.309
## [15,]  0.193  0.213 -0.310
## [16,] -0.061 -0.317 -0.432
## [17,] -0.200 -0.306  0.056
## [18,] -0.235 -0.275  0.121
## [19,] -0.074 -0.306 -0.456
## [20,] -0.029 -0.286 -0.497
## [21,] -0.033 -0.270  0.118
```

```
summary(spca)
```

```
##                          PC1   PC2   PC3
## Explained variance     8.507 3.458 2.701
## Standard deviations    2.917 1.860 1.644
## Proportion of variance 0.405 0.165 0.129
## Cumulative proportion  0.405 0.570 0.698
```

```
#probabilistic PCA model
ppca=pca(data2, method = "ppca", nPcs = 3)
```

```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
## Warning in principal(r = r, nfactors = nfactors, residuals = residuals, :
## The matrix is not positive semi-definite, scores found from Structure
## loadings
```

```
ppca
```

```
## Principal Components Analysis
## Call: principal(r = r, nfactors = nfactors, residuals = residuals,
##     rotate = rotate, n.obs = n.obs, covar = covar, scores = scores,
##     missing = missing, impute = impute, oblique.scores = oblique.scores,
##     method = method, nPcs = 3)
## Standardized loadings (pattern matrix) based upon correlation matrix
##              PC1     h2   u2 com
## idealpoint  0.76 0.5711 0.43   1
## polity      0.88 0.7785 0.22   1
## polity2     0.88 0.7785 0.22   1
## democ       0.91 0.8352 0.16   1
## autoc      -0.77 0.5949 0.41   1
## unreg       0.31 0.0943 0.91   1
## physint     0.63 0.4008 0.60   1
## speech      0.72 0.5216 0.48   1
```

```
## new_empinx  0.89 0.7846 0.22   1
## wecon       0.48 0.2317 0.77   1
## wopol       0.49 0.2389 0.76   1
## wosoc       0.66 0.4395 0.56   1
## elecsd      0.83 0.6875 0.31   1
## gdp.pc.wdi  0.57 0.3291 0.67   1
## gdp.pc.un   0.56 0.3178 0.68   1
## pop.wdi    -0.18 0.0318 0.97   1
## amnesty    -0.59 0.3423 0.66   1
## statedept  -0.69 0.4697 0.53   1
## milper     -0.22 0.0470 0.95   1
## cinc       -0.08 0.0070 0.99   1
## domestic9  -0.10 0.0092 0.99   1
##
##                 PC1
## SS loadings    8.51
## Proportion Var 0.41
##
## Mean item complexity =  1
## Test of the hypothesis that 1 component is sufficient.
##
## The root mean square of the residuals (RMSR) is  0.19
##  with the empirical chi square  1610.76  with prob <  8.8e-224
##
## Fit based upon off diagonal values = 0.8
```

Setting the number of the PC equals to 3, the variance that explained the three PCs and by each of the PC doesn't have substantive difference. Sparse PCA is good for error construction, and useful for dataset if there are a lot of missing values, it attempts to find sparse weight vectors thus to improve the interpretability of the model. Probabilistic PCA derives parameter estimates to probabilistically map the space rather than simply maximizing feature variance when simply maximizing feature variance. The few difference in this date might because there is no missing value in this case.