Statistical Methods 2: Porfolio 1

Kieran Morris

1 Factor Analysis on mtcars dataset

We import the `mtcars` dataset, which has 32 observations on 11 variables. We will attempt to perform factor analysis on it - ideally matching up with results from PCA as presented in the lecture notes.

```
data(mtcars)
data <- mtcars
print(data)
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```r
print(dim(data))
```

```
## [1] 32 11
```

1.1 Choosing the number of factors

Since we have 11 variables we can write out

$$\triangle_{p,k} = (11-k)^2/2 - (11+k)/2,$$

which we see is negative when $k < 6$, so our possible values for the loading sizes are $\{1, 2, 3, 4, 5, 6\}$.

To strike a balance between accuracy and interpatibility we will use $k = 4$ as our number of factors.

1.2 Finding the loading matrix

We then perform the factor analysis on the `mtcars` dataset with the function `factanal`, specifying our number of factors as 4 and use `"varimax"` to rotate the factors to a simpler form.

```r
FA <- factanal(mtcars, factors = 4, rotation = "varimax")
#Perform factor analysis
Lambda <- FA$loadings
#Estimate lambda
Spec_Var <- FA$uniquenesses
Lambda
```

```
##
## Loadings:
##      Factor1 Factor2 Factor3 Factor4
## mpg   0.640  -0.481  -0.423  -0.185
## cyl  -0.606   0.720   0.247   0.114
## disp -0.652   0.573   0.167   0.463
## hp   -0.259   0.733   0.453   0.272
## drat  0.808  -0.263
## wt   -0.742   0.264   0.408   0.400
## qsec -0.194  -0.925  -0.188
## vs    0.272  -0.805  -0.208
## am    0.898
## gear  0.896           0.220
## carb          0.517   0.846
##
##                Factor1 Factor2 Factor3 Factor4
## SS loadings      4.203   3.531   1.490   0.514
## Proportion Var   0.382   0.321   0.135   0.047
## Cumulative Var   0.382   0.703   0.839   0.885
```

```r
Spec_Var
```

```
##        mpg        cyl       disp         hp       drat         wt       qsec
## 0.14533236 0.03946129 0.00500000 0.11713365 0.27321671 0.05370001 0.07124583
##         vs         am       gear       carb
## 0.22748373 0.17687840 0.14845291 0.00500000
```

By convention `factanal` uses maximum likelihood estimation to find `Lambda`. We see that `R` returns the matrix along with some meta data about the columns of the matrix. Fortunately the specific variances can be obtained through `$uniquenesses`.

Next we compute how good an estimate our Λ and Φ are.

```
Phi <- diag(FA$uniquenesses)
#make Phi
R <- Lambda%*%t(Lambda) + Phi
#estimate the correlation matrix
max(abs(cor(data) - R))
```

```
## [1] 0.05399648
```

```
#find the maximum difference between the estimated correlation matrix and the actual correlation matrix
```

This is not too bad, and in fact this error is decreasing in k, so a higher factor count would be more accurate, but remember we must embrace the tradeoff for interpretability. Below we compute the conversion matrix $A_k^{(FA)}$ and find our factors.

```
Factor_Matrix <- solve(t(Lambda)%*%solve(Phi)%*%Lambda)%*%t(Lambda)%*%solve(Phi)
factors <- as.matrix(data)%*%(t(Factor_Matrix))
factors
```

```
##                       Factor1   Factor2    Factor3   Factor4
## Mazda RX4            76.18678 30.334680  -52.25984  418.4990
## Mazda RX4 Wag        75.99825 29.892103  -51.97332  418.6932
## Datsun 710           52.42120 17.702410  -35.76848  288.4026
## Hornet 4 Drive      118.45256 46.261196  -89.32305  670.7470
## Hornet Sportabout   166.87527 72.928768 -128.38297  930.4241
## Valiant             102.52462 39.271396  -76.83402  585.2707
## Duster 360          170.39801 79.675743 -129.29084  930.8556
## Merc 240D            67.82695 20.298466  -46.01766  387.4994
## Merc 230             66.05451 20.534723  -44.29133  373.7272
## Merc 280             79.15630 31.223516  -54.36436  438.4678
## Merc 280C            78.80207 30.844883  -54.06898  438.3277
## Merc 450SE          128.75342 57.571679  -97.00386  713.8025
## Merc 450SL          128.94091 57.589899  -97.07432  714.0148
## Merc 450SLC         128.50605 57.308312  -96.80458  713.6683
## Cadillac Fleetwood  217.17495 94.075464 -165.72210 1217.4942
## Lincoln Continental 212.29529 92.879372 -161.93140 1186.9163
## Chrysler Imperial   204.90017 90.991023 -155.91229 1136.8047
## Fiat 128             39.21261  9.524106  -24.30790  214.8241
## Honda Civic          37.44455  8.267029  -21.78822  206.0798
## Toyota Corolla       35.99613  7.956561  -21.58612  195.7091
## Toyota Corona        56.95138 19.221536  -39.45927  318.9482
## Dodge Challenger    146.09709 63.096683 -112.17624  821.0142
## AMC Javelin         139.72663 60.345355 -107.05965  785.1730
## Camaro Z28          165.83196 78.022904 -125.78101  904.9670
## Pontiac Firebird    184.84770 79.941693 -142.48274 1033.3632
## Fiat X1-9            38.72895  9.988752  -24.45431  214.2609
## Porsche 914-2        59.34733 20.924019  -39.84805  320.4541
## Lotus Europa         49.90123 18.481481  -32.16343  257.2456
```

```
## Ford Pantera L      168.97471 80.918837 -127.89365  909.5488
## Ferrari Dino         73.38434 33.818576  -47.71782  381.3839
## Maserati Bora       150.46983 77.525435 -108.17759  782.4445
## Volvo 142E           58.97032 21.137538  -39.67125  321.9702
```

Here we can see our data has reduced down to 6 dimensions, almost half of the original 11.

1.3 Interpreting our Factors

Since the loading matrix represents the correlation between the factors and the variables, we can find the factors which are highly correlated with different variables:

```r
# Get the loadings
loadings <- FA$loadings
loadings
```

```
##
## Loadings:
##      Factor1 Factor2 Factor3 Factor4
## mpg   0.640  -0.481  -0.423  -0.185
## cyl  -0.606   0.720   0.247   0.114
## disp -0.652   0.573   0.167   0.463
## hp   -0.259   0.733   0.453   0.272
## drat  0.808  -0.263
## wt   -0.742   0.264   0.408   0.400
## qsec -0.194  -0.925  -0.188
## vs    0.272  -0.805  -0.208
## am    0.898
## gear  0.896           0.220
## carb          0.517   0.846
##
##                Factor1 Factor2 Factor3 Factor4
## SS loadings      4.203   3.531   1.490   0.514
## Proportion Var   0.382   0.321   0.135   0.047
## Cumulative Var   0.382   0.703   0.839   0.885
```

We see that : - `Factor1` is highly negatively correlated with `cyl`,`disp` and `wt` and highly positively correlated with `mpg`, `drat`,`am` and `gear`.

- `Factor2` is highly negatively correlated with `qsec` and `vs` and highly positively correlated with `hp` `cyl` and `disp`.
- `Factor3` is highly highly positively correlated with `carb` but is not particularly correlated with any other variable.
- `Factor4` is slightly correlated with `disp` and `wt` but is not particularly correlated with any other variable.

It is hard to discern what exactly these variables are, by observing the `Cumulative Var` row in `loadings` we see as we go up the factors we have less explained variance, similar to PCA. With PCA we were abke to cluster them based on country of origin which gave us a very nice explaination of the principle components, however in this case we cannot do that.

1.4 Comparison aganist PCA

For the purposes of comparison, we will also perform principle component analysis on `mtcars` and comare the results to factor analysis. Below we reperorm FA with $k = 2$.

```
FA <- factanal(mtcars, factors = 2, rotation = "varimax")
#Perform factor analysis
Lambda <- FA$loadings
#Estimate lambda
Phi <- diag(FA$uniquenesses)
Factor_Matrix <- solve(t(Lambda)%*%solve(Phi)%*%Lambda)%*%t(Lambda)%*%solve(Phi)
factors <- as.matrix(data)%*%(t(Factor_Matrix))
factors
```

```
##                       Factor1   Factor2
## Mazda RX4           -27.44912  37.01325
## Mazda RX4 Wag       -27.62859  36.81676
## Datsun 710          -18.03997  26.30081
## Hornet 4 Drive      -49.10826  44.34232
## Hornet Sportabout   -66.00627  71.05368
## Valiant             -43.26920  39.97538
## Duster 360          -61.85214  88.52414
## Merc 240D           -28.39202  21.90634
## Merc 230            -26.02756  28.28852
## Merc 280            -29.32302  39.95645
## Merc 280C           -29.61396  39.83437
## Merc 450SE          -48.85778  64.44566
## Merc 450SL          -48.75176  64.32998
## Merc 450SLC         -49.07676  64.31711
## Cadillac Fleetwood  -88.43715  88.99516
## Lincoln Continental -85.36259  90.29541
## Chrysler Imperial   -79.77773  91.86899
## Fiat 128            -12.82129  16.30159
## Honda Civic         -12.76207  13.30069
## Toyota Corolla      -11.19135  15.12876
## Toyota Corona       -21.28120  27.65412
## Dodge Challenger    -59.27476  61.41522
## AMC Javelin         -56.48285  59.99247
## Camaro Z28          -59.79257  87.81139
## Pontiac Firebird    -74.22789  74.75167
## Fiat X1-9           -13.23345  16.83340
## Porsche 914-2       -19.48207  27.83794
## Lotus Europa        -12.59133  30.21554
## Ford Pantera L      -57.50886  93.05516
## Ferrari Dino        -20.09966  51.75002
## Maserati Bora       -42.99449 105.59940
## Volvo 142E          -19.86581  31.50732
```

Now we perform PCA on `mtcars`. We choose scaled for our purposes.

```
library(mogavs)
PCS <- prcomp(mtcars,center = TRUE,scale = TRUE)
```

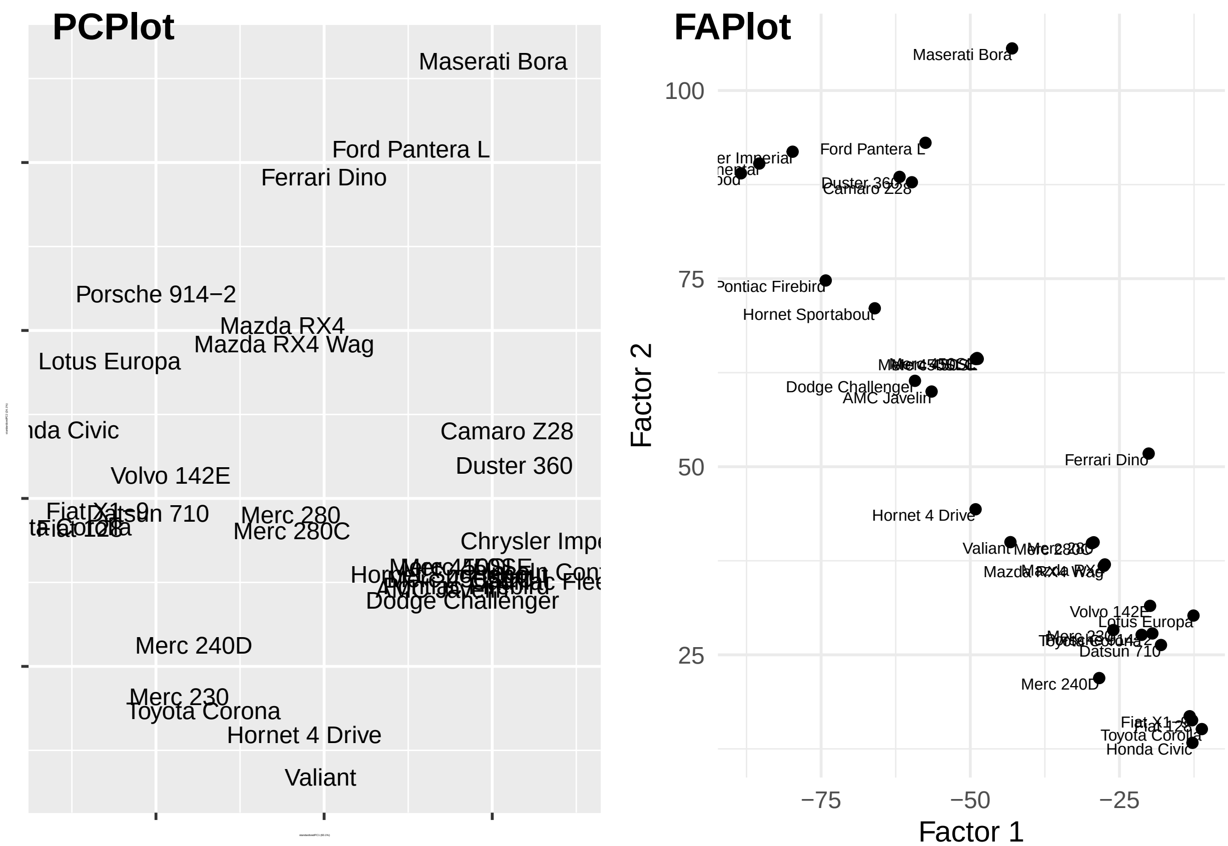
Below we plot both the PCA and FA results together, as we can see there aren't a lot of similarities in this case, this may be because either the error caused by $k = 2$ is too high, or that the factors and PCS are simply not related in this case.

```r
library(ggplot2)
library(ggbiplot)
library(cowplot)

df <- as.data.frame(factors)
names(df) <- c("Factor1", "Factor2")
df$CarModel <- rownames(mtcars)

FAPlot <- ggplot(df, aes(x = Factor1, y = Factor2)) +
  geom_point() +
  theme_minimal() +
  geom_text(aes(label = CarModel), vjust = 1, hjust = 1,size = 2) +
  labs(x = "Factor 1", y = "Factor 2")

PCPlot <- ggbiplot(PCS,ellipse = TRUE,labels = rownames(mtcars),var.axes = FALSE)+
theme(text = element_text(size = 0.5))
plot_grid(PCPlot,FAPlot,labels = c("PCPlot","FAPlot"),ncol = 2)
```



It's hard to find similarities here, clearly the top half of the plots are vaguely similar, such as `Masaerati Bora` and `Ford Pantera` being skewed towards Factor 2 and PC2. But there are too many scramblings of data points to make any meaningful comparisons. There is also a cluster of points in the bottom right, but again there is a lot of variation around the whole plot.

1.5 Conclusion

Considering that PCA provides so much more insight about the distrubution of the data, we would be inclined to use PCA over FA in this case. We had hoped that using `mtcars`, where we know that there is a nice clustering catagorisation of the data, would provide us a good example for factor analysis. Unfortunately we were wrong.

2 Independent Component Analysis on Music Dataset

To perform ICA on the music dataset we make use of the library `fastICA`, we first load the `.wav` music files into our enviroment and use the `seewave` package to read the audio files an convert them into a dataframe to read.

```
library(tuneR)
library(seewave)
library(fastICA)
F1 <- readWave('ICA_mix_1.wav')
F2 <- readWave('ICA_mix_2.wav')
F3 <- readWave('ICA_mix_3.wav')
```

As we care about all of our audio files we `cbind` them into a single dataframe. We then scale the data to have a mean of 0 but keep the standard deviation unchanged. Thanks to the efficiency of the `fastICA` package we simply apply the function and specify how many components we think their are. Now in our case we know there are three. By convention the `fastICA` package uses $\phi(x) = \frac{1}{\alpha} \log \cosh(\alpha x)$ with $\alpha = 1$.

```
Data <- cbind(F1@left,F2@left,F3@left)
Data <- scale(Data,center = TRUE, scale = FALSE)
ica <- fastICA(Data,n.comp = 3)
Components <- ica$S


#savewav(Components[,1],f = F1@samp.rate,filename = "signal1.wav")
#savewav(Components[,2],f = F1@samp.rate,filename = "signal2.wav")
#savewav(Components[,3],f = F1@samp.rate,filename = "signal3.wav")
```

We include our `savewav` commands for clarity but do not run them. It was a success and we managed to seperate our three original files.