08-Tarea

1. ENIGH

Para este ejercicio usaremos los datos de la ENIGH 2014. En particular las variables alimentos, vestido, vivienda, salud, comunica, educacion y esparci (esparcimiento) que indican el gasto trimestral en cada una de las categorías.

1. Calcula los deciles de ingreso usando la variable de ingreso corriente (ing_cor).

Debes tomar en cuenta el diseño de la muestra, puedes usar la función `survey_quantile()` del paquete `srvyr` o `svyquantile()` del paquete `survey`. Reporta las estimaciones y sus errores estándar usando el bootstrap de Rao y Wu.

```
library(readr)
library(dplyr)

concentrado_hogar <- read_csv("concentradohogar.csv")

hogar <- concentrado_hogar %>%
    select(folioviv, foliohog, est_dis, upm, factor_hog, ing_cor, alimentos,
        vestido, vivienda, salud, transporte, comunica, educacion, esparci)

glimpse(hogar)
```

```
## Rows: 19,479
## Columns: 14
## $ folioviv   <chr> "0100008302", "0100008303", "0100008304", "0100008305", ...
## $ foliohog   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 1,...
## $ est_dis    <chr> "005", "005", "005", "005", "005", "005", "005", "005", ...
## $ upm        <chr> "00670", "00670", "00670", "00670", "00600", "00600", "0...
## $ factor_hog <dbl> 694, 694, 694, 694, 660, 660, 660, 660, 660, 583, 583, 5...
## $ ing_cor    <dbl> 39786.79, 19523.90, 99257.66, 87883.68, 84427.04, 232013...
## $ alimentos  <dbl> 11172.63, 1941.38, 16759.08, 8678.43, 12873.09, 18530.30...
## $ vestido    <dbl> 0.00, 0.00, 489.12, 293.47, 0.00, 4519.53, 0.00, 0.00, 0...
## $ vivienda   <dbl> 2199.32, 0.00, 2989.43, 1100.00, 3062.50, 1580.00, 5090....
## $ salud      <dbl> 1401.83, 117.38, 1242.37, 420.63, 0.00, 33534.75, 2974.8...
## $ transporte <dbl> 5490.71, 0.00, 5829.25, 1440.00, 7006.44, 16669.55, 7896...
## $ comunica   <dbl> 1110.00, 0.00, 2540.32, 1440.00, 1200.00, 4703.22, 2670....
## $ educacion  <dbl> 0.00, 0.00, 0.00, 3019.35, 0.00, 22285.15, 0.00, 24262.1...
## $ esparci    <dbl> 900.00, 0.00, 0.00, 0.00, 1050.00, 7066.23, 12182.60, 0....
```

```
library(srvyr)
library(survey)

enigh_design <- hogar %>%
```

```
    as_survey_design(ids = upm, weights = factor_hog, strata = est_dis)

set.seed(2020)

enigh_boot <- enigh_design %>%
    as_survey_rep(type = "subbootstrap", replicates = 500)

enigh_boot %>%
    srvyr::summarise(mean_ingcor = survey_mean(ing_cor))
```

```
## # A tibble: 1 x 2
##   mean_ingcor mean_ingcor_se
##         <dbl>          <dbl>
## 1      39719.           973.
```

```
qq <- svyquantile(~ing_cor, enigh_boot, quantiles = seq(0.1, 1, 0.1), interval.type = "quantile")
qq
```

```
## Statistic:
##         ing_cor
## q0.1   10622.27
## q0.2   14775.08
## q0.3   18597.46
## q0.4   22682.01
## q0.5   27185.82
## q0.6   32725.96
## q0.7   40057.31
## q0.8   51990.48
## q0.9   76284.93
## q1   4150377.02
## SE:
##           ing_cor
## q0.1     149.6292
## q0.2     180.8199
## q0.3     201.8311
## q0.4     222.9581
## q0.5     236.9197
## q0.6     331.2135
## q0.7     475.4123
## q0.8     787.6235
## q0.9    1388.5234
## q1   1342955.8688
```

2. Crea una nueva variable que indique el decil de ingreso para cada hogar. Tips: 1) una función que puede resultar útil es `cut2()` (de `Hmisc`),

2) si usas el paquete `srvyr` puedes usar `mutate()` sobre el objeto `survey` con pesos de replicaciones bootstrap.

```
library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following object is masked from 'package:survey':
##
##     deff

## The following object is masked from 'package:srvyr':
##
##     summarize

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
hogar$decil<-as.character(cut2(hogar$ing_cor,qq))
hogar
```

```
## # A tibble: 19,479 x 15
##    folioviv foliohog est_dis upm   factor_hog ing_cor alimentos vestido vivienda
##    <chr>       <dbl> <chr>   <chr>      <dbl>   <dbl>     <dbl>   <dbl>    <dbl>
##  1 0100008~        1 005     00670        694  39787.    11173.     0      2199.
##  2 0100008~        1 005     00670        694  19524.     1941.     0         0
##  3 0100008~        1 005     00670        694  99258.    16759.   489.     2989.
##  4 0100008~        1 005     00670        694  87884.     8678.   293.     1100
##  5 0100010~        1 005     00600        660  84427.    12873.     0      3062.
##  6 0100010~        1 005     00600        660 232014.    18530.  4520.     1580
##  7 0100010~        1 005     00600        660  90941.    10819.     0      5090.
##  8 0100010~        1 005     00600        660  70309.    12491.     0      1755
##  9 0100010~        1 005     00600        660  92508.     6853.     0     15456.
## 10 0100018~        1 004     00570        583  80865.    12876.    84.1    2053
## # ... with 19,469 more rows, and 6 more variables: salud <dbl>,
## #   transporte <dbl>, comunica <dbl>, educacion <dbl>, esparci <dbl>,
## #   decil <chr>
```

3. Estima para cada decil, el porcentaje del gasto en cada categoría (), reporta el error estándar de las estimaciones, usa el bootstrap de Rao y Wu.

Tip: - 1) agrega una variable que indica para cada hogar el porcentaje de gasto en cada categoría,

- 2) si usas srvyr puedes usar la función `group_by()` para estimar la media del porcentaje de gasto por decil.

```r
fun_zero <- function(x){
  if(x==0){
    return(.01)
  }
  else{
    return(x)
  }
}

den <- sapply((hogar$alimentos +
               hogar$vestido +
               hogar$vivienda +
               hogar$salud +
               hogar$transporte +
               hogar$comunica +
               hogar$educacion +
               hogar$esparci)
              ,fun_zero)

hogar$p_alim <- hogar$alimentos/den
hogar$p_vestido <- hogar$vestido/den
hogar$p_vivienda <- hogar$vivienda/den
hogar$p_salud <- hogar$salud/den
hogar$p_transporte <- hogar$transporte/den
hogar$p_comunica <- hogar$comunica/den
hogar$p_educacion <- hogar$educacion/den
hogar$p_esparci <- hogar$esparci/den

enigh_design2 <- hogar %>%
    as_survey_design(ids = upm, weights = factor_hog, strata = est_dis)

enigh_boot2 <- enigh_design2 %>%
    as_survey_rep(type = "subbootstrap", replicates = 500)

err <- data.frame(enigh_boot2 %>%
                    srvyr::group_by(decil) %>%
                    srvyr::summarise(mean_palim = survey_mean(p_alim),
                                     mean_pvestido = survey_mean(p_vestido),
                                     mean_pvivienda = survey_mean(p_vivienda),
                                     mean_psalud = survey_mean(p_salud),
                                     mean_ptransporte = survey_mean(p_transporte),
                                     mean_pcomunica = survey_mean(p_comunica),
                                     mean_peducacion = survey_mean(p_educacion),
                                     mean_pesparci = survey_mean(p_esparci)))
err
```

```
##                decil mean_palim mean_palim_se mean_pvestido mean_pvestido_se
## 1  [      0,  10622)  0.6294435   0.006179444    0.02736907      0.001512451
## 2  [  10622,  14775)  0.5714863   0.005307653    0.02838772      0.001487148
## 3  [  14775,  18597)  0.5397263   0.004739619    0.02656811      0.001196838
## 4  [  18597,  22682)  0.5241734   0.004553359    0.02746975      0.001133882
## 5  [  22682,  27186)  0.4963811   0.004725181    0.02771933      0.001052654
## 6  [  27186,  32726)  0.4774084   0.004633190    0.03010250      0.001690471
```

```
## 7  [  32726,  40057)  0.4531834   0.004553305    0.03174477      0.001237088
## 8  [  40057,  51990)  0.4307530   0.004128175    0.03465659      0.001226922
## 9  [  51990,  76285)  0.3976248   0.004519442    0.03584568      0.001285889
## 10 [  76285,4150377]  0.3235232   0.004963262    0.04236249      0.001626010
##    mean_pvivienda mean_pvivienda_se mean_psalud mean_psalud_se mean_ptransporte
## 1       0.1303336       0.004492607  0.02595083    0.002099225        0.1141557
## 2       0.1361030       0.003748815  0.02888787    0.002448672        0.1362703
## 3       0.1350544       0.003089434  0.02196807    0.001491518        0.1608099
## 4       0.1251411       0.002956287  0.02208445    0.001585537        0.1705709
## 5       0.1265108       0.003079101  0.01927620    0.001386682        0.1940208
## 6       0.1181212       0.002808902  0.02016949    0.001509973        0.2072038
## 7       0.1194003       0.003125958  0.02006211    0.001266755        0.2175729
## 8       0.1121987       0.003047056  0.02301865    0.001771198        0.2250942
## 9       0.1069972       0.003153454  0.02551481    0.001878510        0.2430384
## 10      0.1049304       0.004560591  0.03788884    0.002856598        0.2496626
##    mean_ptransporte_se mean_pcomunica mean_pcomunica_se mean_peducacion
## 1          0.003823567     0.02038837       0.001225578      0.03526563
## 2          0.003946988     0.02873609       0.001137577      0.05406708
## 3          0.003786735     0.03508495       0.001326424      0.06293573
## 4          0.003212897     0.03812413       0.001226898      0.07481153
## 5          0.003487618     0.04200418       0.001267451      0.07375984
## 6          0.003564719     0.04871028       0.001343528      0.07380523
## 7          0.004077495     0.05337407       0.001346526      0.08019964
## 8          0.003017469     0.05613178       0.001304853      0.08622370
## 9          0.003536906     0.06117834       0.001160116      0.09402181
## 10         0.004975099     0.06678864       0.001566101      0.12187710
##    mean_peducacion_se mean_pesparci mean_pesparci_se
## 1         0.002126691   0.009966443     0.0009167362
## 2         0.003263709   0.012302919     0.0007619998
## 3         0.003162207   0.016388612     0.0010599630
## 4         0.003610525   0.017113520     0.0010756223
## 5         0.003339691   0.018185868     0.0008796838
## 6         0.003450374   0.020952883     0.0011905733
## 7         0.003441691   0.024114423     0.0010537542
## 8         0.003607545   0.030272428     0.0019951849
## 9         0.004031063   0.035778979     0.0014494231
## 10        0.006658080   0.052622285     0.0022912849
```

4. Realiza una gráfica con las estimaciones del paso 3.

```
library(ggplot2)
library(patchwork)
x_ <- c(0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1)
err$decil_r <- x_

df <- err %>% select("decil_r", "mean_palim", "mean_palim_se")

g_alim <- ggplot(err %>% select("decil_r", "mean_palim", "mean_palim_se"),
            aes(x_,
                mean_palim*100,
                ymin = mean_palim*100 - mean_palim_se*100,
                ymax = mean_palim*100 + mean_palim_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
```

```
   xlab("decil") +
  ylab("alimentos")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_vestido <- ggplot(err %>% select("decil_r", "mean_pvestido", "mean_pvestido_se"),
            aes(x_,
                mean_pvestido*100,
                ymin = mean_pvestido*100 - mean_pvestido_se*100,
                ymax = mean_pvestido*100 + mean_pvestido_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("vestido")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_vivienda <- ggplot(err %>% select("decil_r", "mean_pvivienda", "mean_pvivienda_se"),
            aes(x_,
                mean_pvivienda*100,
                ymin = mean_pvivienda*100 - mean_pvivienda_se*100,
                ymax = mean_pvivienda*100 + mean_pvivienda_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("vivienda")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_salud <- ggplot(err %>% select("decil_r", "mean_psalud", "mean_psalud_se"),
            aes(x_,
                mean_psalud*100,
                ymin = mean_psalud*100 - mean_psalud_se*100,
                ymax = mean_psalud*100 + mean_psalud_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("salud")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_transporte <- ggplot(err %>% select("decil_r", "mean_ptransporte", "mean_ptransporte_se"),
            aes(x_,
                mean_ptransporte*100,
                ymin = mean_ptransporte*100 - mean_ptransporte_se*100,
                ymax = mean_ptransporte*100 + mean_ptransporte_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("transporte")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_educa <- ggplot(err %>% select("decil_r", "mean_peducacion", "mean_peducacion_se"),
            aes(x_,
                mean_peducacion*100,
                ymin = mean_peducacion*100 - mean_peducacion_se*100,
                ymax = mean_peducacion*100 + mean_peducacion_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("educacion")
```

```
## Warning: Ignoring unknown parameters: width
```

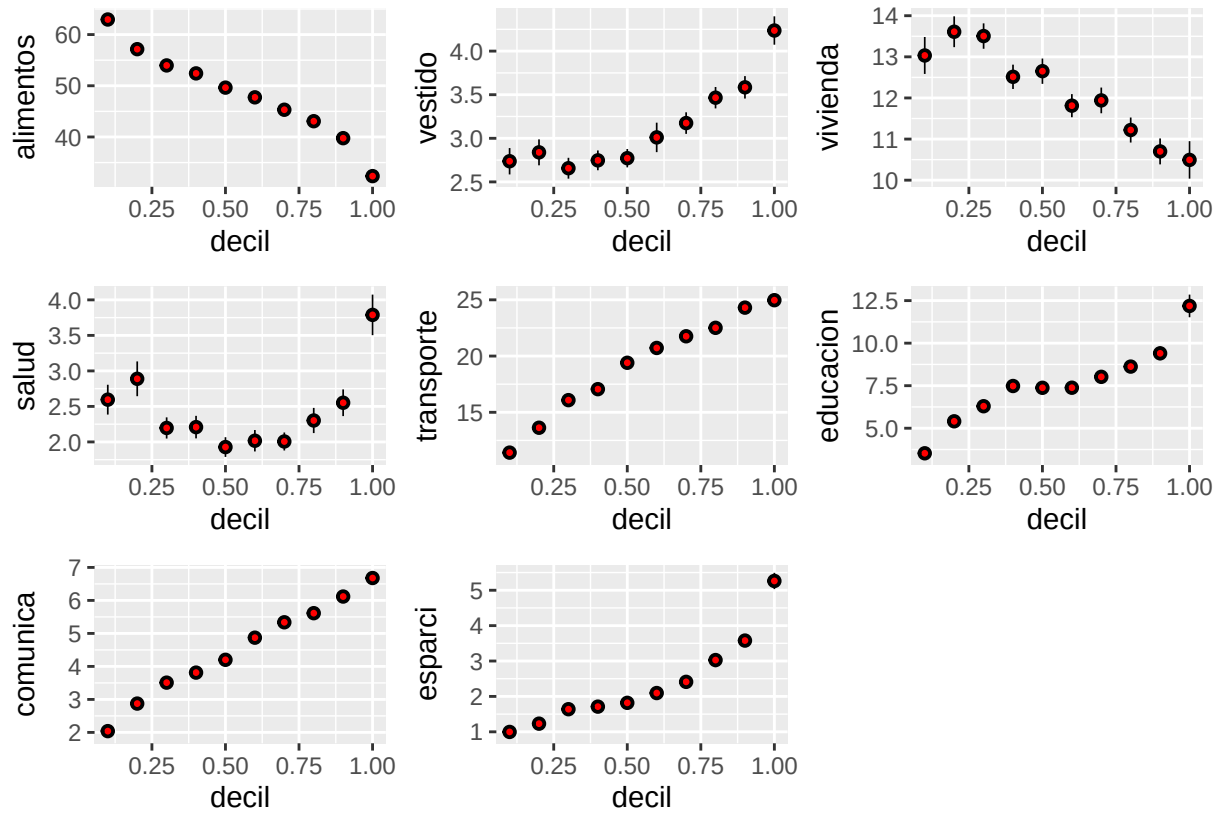
```
g_comunica <- ggplot(err %>% select("decil_r", "mean_pcomunica", "mean_pcomunica_se"),
            aes(x_,
                mean_pcomunica*100,
                ymin = mean_pcomunica*100 - mean_pcomunica_se*100,
                ymax = mean_pcomunica*100 + mean_pcomunica_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("comunica")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_esparci <- ggplot(err %>% select("decil_r", "mean_pesparci", "mean_pesparci_se"),
            aes(x_,
                mean_pesparci*100,
                ymin = mean_pesparci*100 - mean_pesparci_se*100,
                ymax = mean_pesparci*100 + mean_pesparci_se*100)) +
  geom_pointrange(width = 0.2, size = 0.3, color="black", fill="red", shape=21) +
  xlab("decil") +
  ylab("esparci")
```

```
## Warning: Ignoring unknown parameters: width
```

```
g_alim + g_vestido + g_vivienda + g_salud + g_transporte + g_educa + g_comunica +g_esparci
```



2. Componentes Principales

Los datos *marks* (Mardia, Kent y Bibby, 1979) contienen los puntajes de 88 estudiantes en 5 pruebas: mecánica, vectores, álgebra, análisis y estadística. Cada renglón corresponde a la calificación de un estudiante en cada prueba. Para este ejercicio no es necesario que conozcas componentes principales pues puedes implementar el bootstrap siguiendo el código propuesto y discutiremos los detalles del análisis en la próxima clase.

```
data(marks, package = "ggm")
glimpse(marks)
```

```
## Rows: 88
## Columns: 5
## $ mechanics  <dbl> 77, 63, 75, 55, 63, 53, 51, 59, 62, 64, 52, 55, 50, 65, ...
## $ vectors    <dbl> 82, 78, 73, 72, 63, 61, 67, 70, 60, 72, 64, 67, 50, 63, ...
## $ algebra    <dbl> 67, 80, 71, 63, 65, 72, 65, 68, 58, 60, 60, 59, 64, 58, ...
## $ analysis   <dbl> 67, 70, 66, 70, 70, 64, 65, 62, 62, 62, 63, 62, 55, 56, ...
## $ statistics <dbl> 81, 81, 81, 68, 63, 73, 68, 56, 70, 45, 54, 44, 63, 37, ...
```

Un análisis de componentes principales proseguiría como sigue:

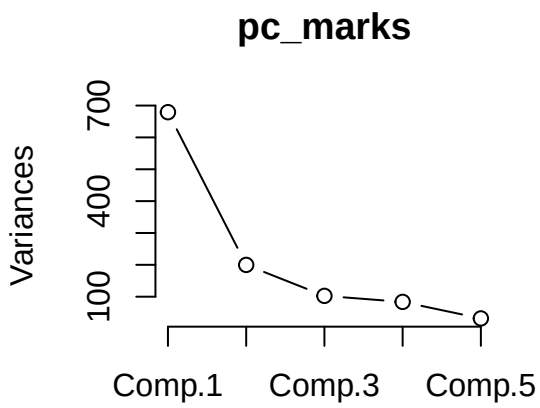
```
pc_marks <- princomp(marks)
summary(pc_marks)
```

```
## Importance of components:
##                            Comp.1     Comp.2      Comp.3     Comp.4     Comp.5
## Standard deviation     26.061142 14.1355705 10.12760414 9.14706148 5.63807655
## Proportion of Variance  0.619115  0.1821424  0.09349705 0.07626893 0.02897653
## Cumulative Proportion   0.619115  0.8012575  0.89475453 0.97102347 1.00000000
```

```
loadings(pc_marks)
```

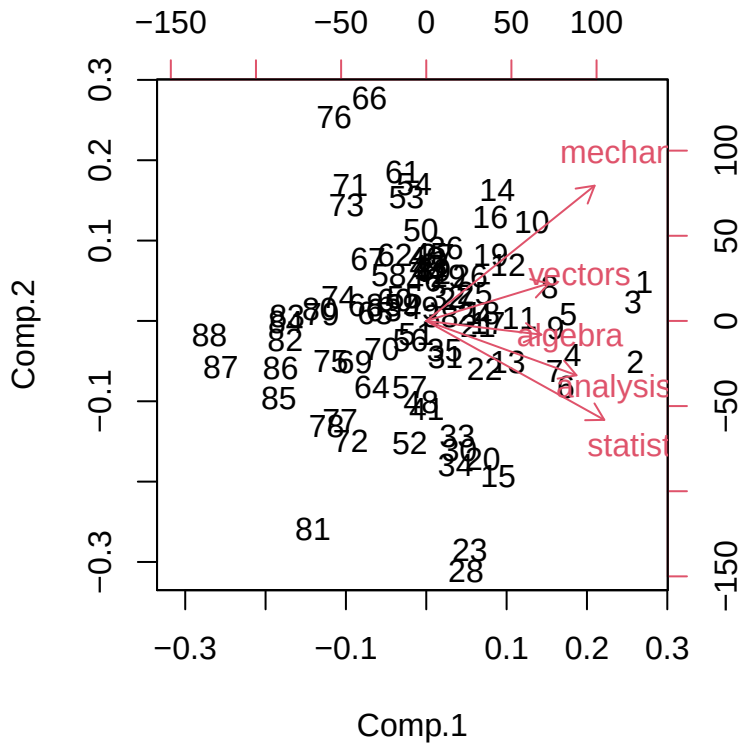
```
##
## Loadings:
##            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## mechanics   0.505  0.749  0.300  0.296
## vectors     0.368  0.207 -0.416 -0.783  0.189
## algebra     0.346        -0.145        -0.924
## analysis    0.451 -0.301 -0.597  0.518  0.286
## statistics  0.535 -0.548  0.600 -0.176  0.151
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

```
plot(pc_marks, type = "lines")
```



Y graficamos:

```
biplot(pc_marks)
```



Los cálculos de un análisis de componentes principales involucran la matriz de covarianzas empírica G (estimaciones *plug-in*)

$$G_{jk} = \frac{1}{88}\sum_{i=1}^{8} 8(x_{ij} - \bar{x_j})(x_{ik} - \bar{x_k})$$

para $j, k = 1, 2, 3, 4, 5$, y donde $\bar{x_j} = \sum_{i=1}^{8} 8x_{ij}/88$ (la media de la i-ésima columna).

```
G <- cov(marks) * 87 / 88
G
```

```
##            mechanics   vectors   algebra  analysis statistics
## mechanics   302.2934 125.77686 100.42510 105.06508  116.07076
## vectors     125.7769 170.87810  84.18957  93.59711   97.88688
## algebra     100.4251  84.18957 111.60318 110.83936  120.48567
## analysis    105.0651  93.59711 110.83936 217.87603  153.76808
## statistics  116.0708  97.88688 120.48567 153.76808  294.37177
```

Los *pesos* y las *componentes principales* no son mas que los eigenvalores y eigenvectores de la matriz de covarianzas G, estos se calculan a través de una serie de de manipulaciones algebraicas que requieren cálculos del orden de p^3 (cuando G es una matriz de tamaño p×p).

```
eigen_G <- eigen(G)
lambda <- eigen_G$values
v <- eigen_G$vectors
lambda
```

```
## [1] 679.18311 199.81435 102.56837  83.66873  31.78791
```

```
v
```

```
##            [,1]        [,2]       [,3]         [,4]        [,5]
## [1,] -0.5054457  0.74874751  0.2997888  0.296184264 -0.07939388
## [2,] -0.3683486  0.20740314 -0.4155900 -0.782888173 -0.18887639
## [3,] -0.3456612 -0.07590813 -0.1453182 -0.003236339  0.92392015
## [4,] -0.4511226 -0.30088849 -0.5966265  0.518139724 -0.28552169
## [5,] -0.5346501 -0.54778205  0.6002758 -0.175732020 -0.15123239
```

1. Proponemos el siguiente modelo simple para puntajes correlacionados:

$$\mathbf{x}_i = Q_i \mathbf{v}$$

donde $\mathbf{x}_i$ es la tupla de calificaciones del i-ésimo estudiante, Q_i es un número que representa la habilidad del estudiante y $\mathbf{v}$ es un vector fijo con 5 números que aplica a todos los estudiantes. Si este modelo simple fuera cierto, entonces únicamente el $\hat{\lambda}_1$ sería positivo y $\mathbf{v} = \hat{v}_1$. Sea

$$\hat{\theta} = \sum_{i=1}^{5} \hat{\lambda}_i$$

el modelo propuesto es equivalente a $\hat{\theta} = 1$, inculso si el modelo es correcto, no esperamos que $\hat{\theta}$ sea exactamente uno pues hay ruido en los datos.

```
theta_hat <- lambda[1]/sum(lambda)
theta_hat
```

```
## [1] 0.619115
```

El valor de $\hat{\theta}$ mide el porcentaje de la varianza explicada en la primer componente principal, ¿qué tan preciso es $\hat{\theta}$? La complejidad matemática en el cálculo de $\hat{\theta}$ es irrelevante siempre y cuando podamos calcular $\hat{\theta}^*$ para una muestra bootstrap, en esta caso una muestra bootsrtap es una base de datos de 88×5 $\mathbf{X}^*$, donde las filas $\mathbf{x_i}^*$ de $\mathbf{X}^*$ son una muestra aleatoria de tamaño 88 de la verdadera matriz de datos.

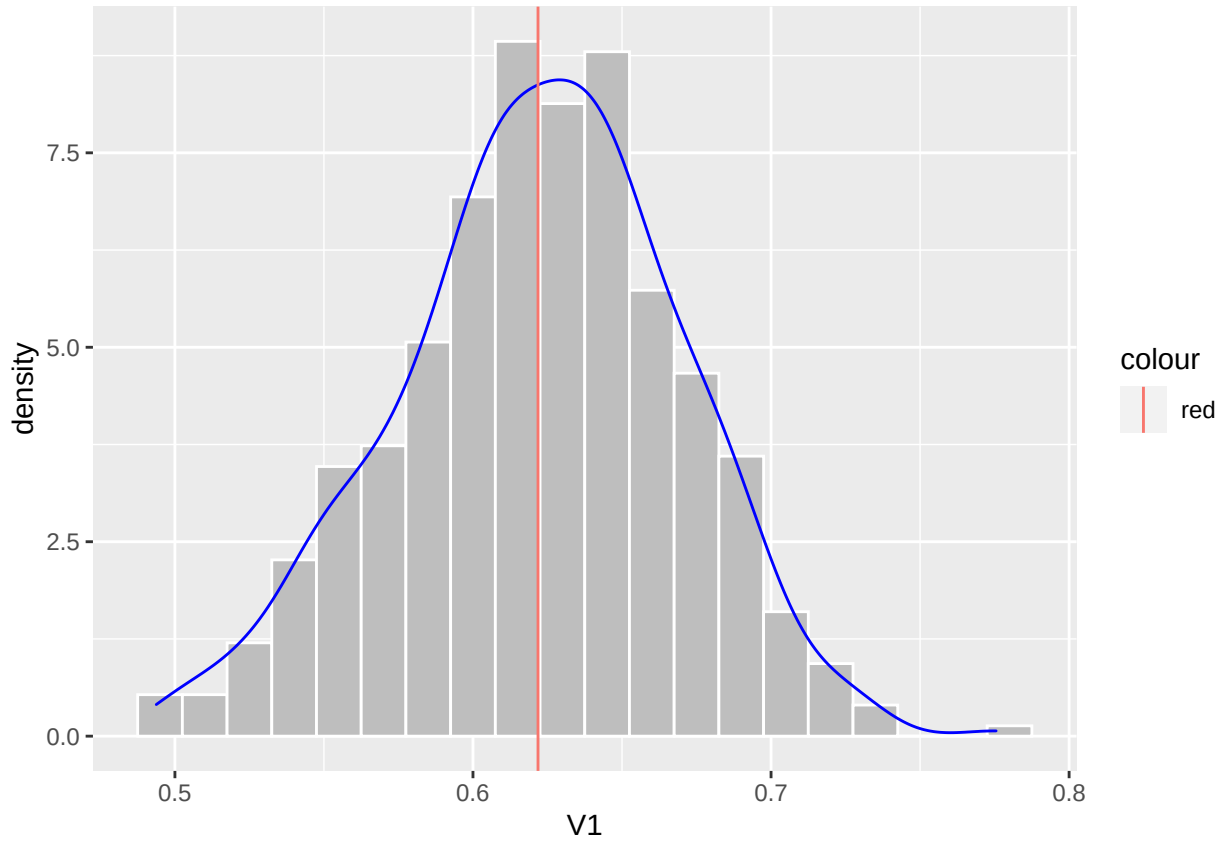
- Utiliza bootstrap para calcular el error estándar de $\hat{\theta}$.
- Grafica la distribución bootstrap.

```
p_boot <- function(){
  muestra_boot <- sample_n(marks, size = 88, replace = TRUE)
  G <- cov(muestra_boot) * 87 / 88
  eigen_G <- eigen(G)
  theta_hat <- eigen_G$values[1] / sum(eigen_G$values)
}

n <- 500

library(plyr)
```

```
## --------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:Hmisc':
##
##     is.discrete, summarize

## The following objects are masked from 'package:srvyr':
##
##     mutate, rename, summarise, summarize

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```r
thetas_boot <- rdply(n, p_boot)

ggplot(thetas_boot, aes(x = V1)) +
  geom_histogram((aes(y = ..density..)), color = "white", fill = "gray", binwidth = 0.015) +
  geom_density(color = "blue") +
  geom_vline(aes(xintercept = mean(V1), color = "red"))
```


density
7.5
5.0
2.5
0.0
0.5
0.6
0.7
0.8
V1
colour
red