

Tarea3

Minimos cuadrados

Plantear el problema de minimos cuadrados como un problema de minimizacion:

Tenemos que $Y = X\beta + \epsilon$, por lo que donde ϵ es el error, por lo que podemos plantear el problema como la minimizació de este error

$$\min ||\epsilon||^2$$

que es equivalente a

$$\min_{\beta} ||Y - X\beta||^2$$

Para encontrar el estimador $\hat{\beta}$ es necesario derivar e igualar a 0: Derivando respecto a β :

$$-2X^t(Y - X\hat{\beta}) = 0$$

$$2X^tY = 2X^tX\hat{\beta}$$

$$\hat{\beta} = (X^tX)^{-1}X^tY$$

Es el $\hat{\beta}$ que soluciona el problema de minimización.

Con esto podemos ver que para cada:

$$\hat{\beta}_i = \frac{\sum_i x_i y_i}{\sum_i x_i^2}$$

La cuál es una composición lineal de los parametros. Desde el planteamiento estamos buscando que el resultado sea lineal en los parametros, no en los datos por lo que este metodo serviría para poder aproximar polinomios de la forma $Y = X^2$

¿En qué se relaciona esto con un problema de proyección?

Recordemos que al encontrar la proyección de un vector $Proy_X(y) = \frac{\langle x, y \rangle}{\|x\|^2}y$ el error de proyección $Y - Proy_x(Y)$ será ortogonal a la proyección. En el caso del problema de minimizar el error, nosotros estamos proyectando la variable dependiente Y sobre el espacio formado por nuestros datos X. y buscamos las combinaciones sobre X que hagan este error mas pequeño.

Viéndolo como el teorema de Pitágoras el cual nos dice que el cuadrado de la hipotenusa de un triángulo es igual a la suma de los otros dos lados $h^2 = x^2 + y^2$ si el ángulo formado entre x y y es de 90 grados, en este caso $H = Y$ las variables dependientes, $x = Proy_x(Y)$ y $y = error$, entonces la manera en la que se puede cumplir el teorema es que el error sea ortogonal a la proyección (ángulo de 90 grados) sino el error sería mas grande.

¿Que logramos al agregar una columna de unos en la matriz X?

Al definir una columna de unos estamos cambiando un poco el problema. Antes de agregar la columna:

$$y_j = \beta_1 x_{1j} + \dots + \beta_n x_{nj}$$

Al definir la columna de unos estamos dando cierta holgura al modelo ya que al agregar un parámetro α no estamos forzando a que la aproximación pase por el origen.

$$y_j = \alpha + \beta_1 x_{1j} + \dots + \beta_n x_{nj}$$

Plantear el problema de regresión como un problema de estadística con errores....

¿Cuál es la función de verosimilitud del problema anterior?

Planteando el problema como

$$Y = X\beta + \epsilon$$

con

$$\epsilon \sim N(0, \sigma^2 I_n)$$

podemos probar que

$$Y \sim N(X\beta, \sigma^2 I_n)$$

Demostración:

$$Y = X\beta + \epsilon$$

$$\epsilon = Y - X\beta$$

1) Media:

$$E(\epsilon) = E(Y) - E(X\beta) = 0$$

$$E(Y) = E(X\beta)$$

$$E(Y) = X\beta$$

2) Varianza:

$$Var(\epsilon) = Var(Y) + Var(X\beta) - 2Cov(Y, X\beta) = \sigma^2 I_n$$

donde:

$$Var(X\beta) = 0$$

y

$$\begin{aligned} Cov(Y, X\beta) &= E((Y - E(Y))(X\beta - E(X\beta))) \\ &= E((Y - X\beta)(X\beta - X\beta)) = 0 \end{aligned}$$

entonces:

$$Var(\epsilon) = Var(Y) = \sigma^2 I_n$$

Ahora, falta demostrar que Y es Normal: Por propiedades de la Normal como $\epsilon \sim N(0, \sigma^2 I_n)$

$$\epsilon + X\beta \sim N(X\beta, \sigma^2 I_n)$$

(Esto era más rápido pero bueno...)

Entonces con esto podemos escribir la función de verosimilitud de Y como:

$$\begin{aligned} L(\beta, \sigma^2) &= \prod \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - X\beta)^2}{2\sigma^2}\right) \\ &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)^t(y - X\beta)\right) \end{aligned}$$

A la cual le sacamos Logaritmo para después poder buscar los parámetros que maximicen la función:

$$\log(L(\beta, \sigma^2)) = -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} (y - X\beta)^t (y - X\beta)$$

Ahora hay que maximizar esa función sobre β y σ^2 Respecto de β

$$\frac{dL}{d\beta} = \frac{1}{\sigma^2}(y - X\beta)^t X = 0$$

$$X^t y = X^t X \beta$$

$$\hat{\beta} = (X^t X)^{-1} X^t y$$

Respecto de σ^2

$$\frac{dL}{d\sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}(y - X\beta)^t(y - X\beta) = 0$$

$$\sigma^2 = \frac{1}{n}(y - X\beta)^t(y - X\beta)$$

que puede resolverse usando $\hat{\beta}$ que encontramos en la solución anterior.

Como pudimos ver el resultado de máxima verosimilitud

$$\hat{\beta} = (X^t X)^{-1} X^t y$$

Es igual al resultado de $\hat{\beta}$ de mínimos cuadrados

El teorema de Gauss Markov

El teorema de Gauss Markov plantea que un modelo de regresión lineal en el que se cumple que:

- 1) $E(\epsilon) = 0$
- 2) Los errores no están correlacionados entre sí
- 3) $Var(\epsilon_i = \sigma^2)$ para todo i , los errores tienen la misma varianza

Entonces el mejor estimador lineal inscrito de los coeficientes β es el resultante del problema de minimizar los errores al cuadrado. si existe.

Parte Aplicada

```
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.2.5
library(plotly)

## Warning: package 'plotly' was built under R version 3.2.5

##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```

data("diamonds")
head(diamonds)

## # A tibble: 6 x 10
##   carat      cut color clarity depth table price     x     y     z
##   <dbl>    <ord> <ord>   <ord> <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23    Ideal     E     SI2   61.5     55   326  3.95  3.98  2.43
## 2  0.21    Premium   E     SI1    59.8     61   326  3.89  3.84  2.31
## 3  0.23    Good     E     VS1    56.9     65   327  4.05  4.07  2.31
## 4  0.29    Premium   I     VS2    62.4     58   334  4.20  4.23  2.63
## 5  0.31    Good     J     SI2    63.3     58   335  4.34  4.35  2.75
## 6  0.24  Very Good J     VVS2   62.8     57   336  3.94  3.96  2.48
#Entonces, para hacer la regresion lineal tengo que agarrar los #valores numericos que son
diamonds_num<-diamonds[,c(1,5:10)]

reg_precio=lm(price~carat+depth+table+x+y+z, data=diamonds_num)

summary(reg_precio)

##
## Call:
## lm(formula = price ~ carat + depth + table + x + y + z, data = diamonds_num)
##
## Residuals:
##       Min        1Q        Median        3Q        Max
## -23878.2    -615.0     -50.7     347.9    12759.2
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20849.316    447.562  46.584 < 2e-16 ***
## carat       10686.309    63.201 169.085 < 2e-16 ***
## depth       -203.154    5.504 -36.910 < 2e-16 ***
## table       -102.446    3.084 -33.216 < 2e-16 ***
## x           -1315.668   43.070 -30.547 < 2e-16 ***
## y            66.322    25.523   2.599  0.00937 **
## z            41.628    44.305   0.940  0.34744
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1497 on 53933 degrees of freedom
## Multiple R-squared:  0.8592, Adjusted R-squared:  0.8592
## F-statistic: 5.486e+04 on 6 and 53933 DF, p-value: < 2.2e-16

```

Que tan bien ajusta el modelo depende de la correlacion que tenga cada una de las variables explicativas sobre el precio, esto lo podemos ver a trav'és de:

```

diamonds_num<- diamonds_num[,c(4,1,2,3,5,6,7)]


cor(diamonds_num)

##          price      carat      depth      table        x        y
## price  1.0000000 0.92159130 -0.01064740  0.1271339  0.88443516  0.86542090

```

```

## carat  0.9215913 1.00000000  0.02822431  0.1816175  0.97509423  0.95172220
## depth -0.0106474 0.02822431  1.00000000 -0.2957785 -0.02528925 -0.02934067
## table  0.1271339 0.18161755 -0.29577852  1.0000000  0.19534428  0.18376015
## x      0.8844352 0.97509423 -0.02528925  0.1953443  1.00000000  0.97470148
## y      0.8654209 0.95172220 -0.02934067  0.1837601  0.97470148  1.00000000
## z      0.8612494 0.95338738  0.09492388  0.1509287  0.97077180  0.95200572
##
##          z
## price  0.86124944
## carat  0.95338738
## depth  0.09492388
## table  0.15092869
## x      0.97077180
## y      0.95200572
## z      1.00000000

```

Como podemos ver en la tabla de correlaciones, precio est\'a correlacionada altamente con "carat", "x", "y", "z" y tienen cierta correlacion con death y table, esto me dice que las variables escogidas sirven para explicar el precio. Podemos ver esta relacion de manera grafica a continuacion en el siguiente Diagrama de dispersion :

```
library(GGally)
```

```

## Warning: package 'GGally' was built under R version 3.2.5
## Warning: replacing previous import by 'utils::capture.output' when loading
## 'GGally'
## Warning: replacing previous import by 'utils::head' when loading 'GGally'
## Warning: replacing previous import by 'utils::installed.packages' when
## loading 'GGally'
## Warning: replacing previous import by 'utils::str' when loading 'GGally'
library(ggplot2)

```

```

my_fn <- function(data, mapping, ...){
  p <- ggplot(data = data, mapping = mapping) +
    geom_point() +
    geom_smooth(method=loess, fill="red", color="red", ...) +
    geom_smooth(method=lm, fill="blue", color="blue", ...)
  p
}

```

```
g = ggpairs(diamonds_num, columns = 1:7, lower = list(continuous = my_fn))
g
```

```

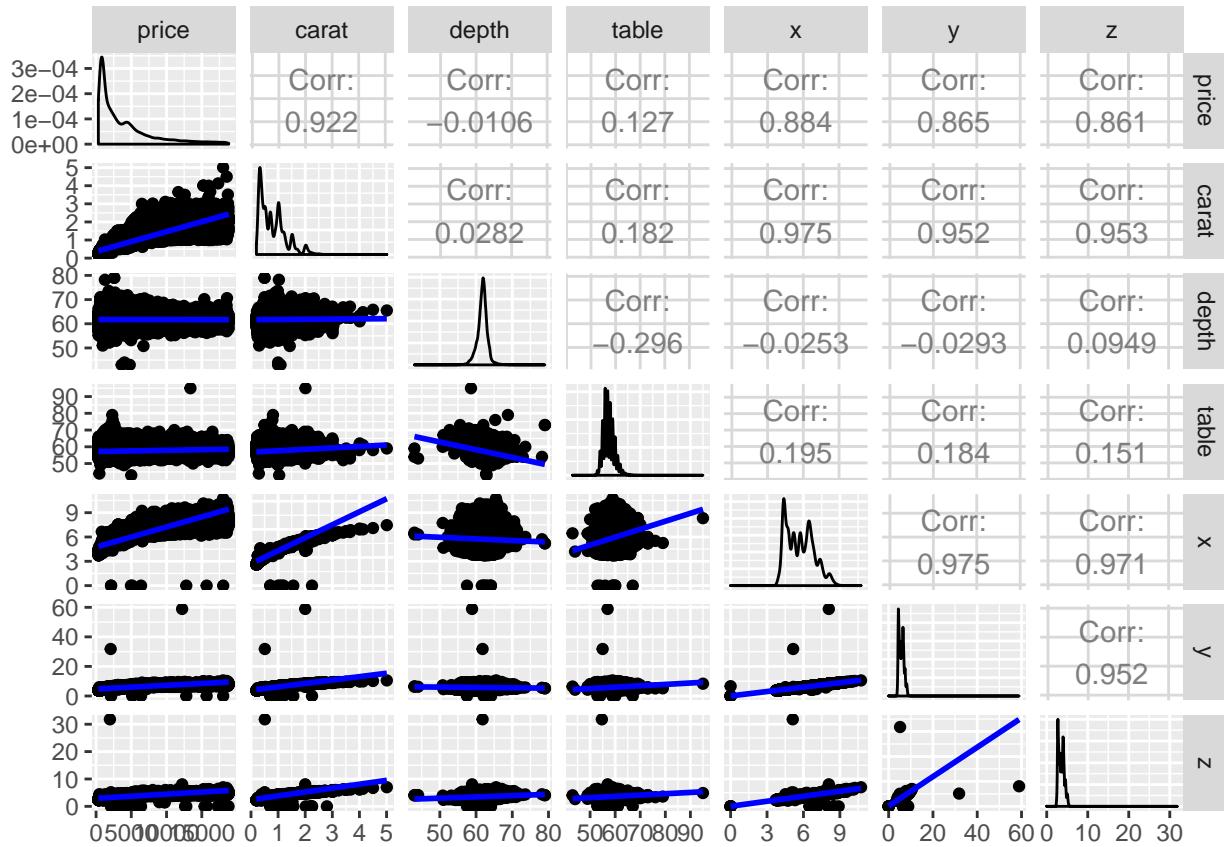
## Warning: Computation failed in `stat_smooth()`:
## 'Calloc' could not allocate memory (18446744071597158400 of 4 bytes)

## Warning: Computation failed in `stat_smooth()`:
## 'Calloc' could not allocate memory (18446744071597158400 of 4 bytes)

## Warning: Computation failed in `stat_smooth()`:
## 'Calloc' could not allocate memory (18446744071597158400 of 4 bytes)

## Warning: Computation failed in `stat_smooth()`:
## 'Calloc' could not allocate memory (18446744071597158400 of 4 bytes)

```

Graficamente así se ven los valores del modelo contra los observados. Podemos ver que no forman una linea de 45 grados por lo que posiblemente hay alg\'un problema dada la correlacion entre las variables explicativas. Esto tambien lo podemos ver en la tabla de correlaciones.

#Grafica de los valores del modelo contra los reales

```
library(plotly)

data<-as.data.frame(cbind(reg_precio$fitted.values,diamonds_num$price))
p <- plot_ly(data = data, x = ~data[,1], y = ~data[,2]) %>%
  layout(title="Valores ajustadosvs Valores observados")

## Warning: replacing previous import by 'shiny::includeHTML' when loading
## 'crosstalk'

## Warning: replacing previous import by 'shiny::knit_print.shiny.tag' when
## loading 'crosstalk'

## Warning: replacing previous import by 'shiny::code' when loading
## 'crosstalk'

## Warning: replacing previous import by 'shiny::includeScript' when loading
## 'crosstalk'

## Warning: replacing previous import by 'shiny::includeMarkdown' when loading
## 'crosstalk'

## Warning: replacing previous import by 'shiny::tags' when loading
## 'crosstalk'

## Warning: replacing previous import by 'shiny::is.singleton' when loading
```

```
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::withTags' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::img' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::tagAppendAttributes' when  
## loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::knit_print.shiny.tag.list'  
## when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::knit_print.html' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::tagAppendChild' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::includeCSS' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::br' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::singleton' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::span' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::a' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::tagList' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::strong' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::tag' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::p' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::validateCssUnit' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::HTML' when loading  
## 'crosstalk'  
## Warning: replacing previous import by 'shiny::h1' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::h2' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::h3' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::h4' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::h5' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::h6' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::tagAppendChildren' when  
## loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::em' when loading 'crosstalk'  
## Warning: replacing previous import by 'shiny::div' when loading 'crosstalk'
```

```

## Warning: replacing previous import by 'shiny::pre' when loading 'crosstalk'
## Warning: replacing previous import by 'shiny::htmlTemplate' when loading
## 'crosstalk'
## Warning: replacing previous import by 'shiny::suppressDependencies' when
## loading 'crosstalk'
## Warning: replacing previous import by 'shiny::tagSetChildren' when loading
## 'crosstalk'
## Warning: replacing previous import by 'shiny::includeText' when loading
## 'crosstalk'
## Warning: replacing previous import by 'shiny::hr' when loading 'crosstalk'
##Si dejo la grafica no se compila bien al hacer el knit

```

Podemos ver que como el modelo tiene una $R^2 = .8592$ por lo que podemos decir que el modelo explica 85% de la varianza con las variables que estamos usando pero hay algunos problemas entre ellas por lo que los estimadores deben de estar sesgados. nuestro error estandar residual $\sigma = 1497$ por lo que $\sigma^2 = 1497^2$

¿Cual es el angulo entre Y y \hat{Y}

$$\arccos(R^2) = \theta$$

En este caso en particular $R^2 = .8592$ por lo que

```
acos(sqrt(.8592))
```

```
## [1] 0.3846484
```

Haciendo una funcion de logverosimilitud

En caso de que podamos aproximar con una normal tenemos lo siguiente:

```

lik1<-function(par,X,y){
  beta<-par[1:ncol(X)]
  sigma2<-par[ncol(X)+1]
  n<-nrow(y)
  X<-as.matrix(X)
  beta<-as.matrix(beta)

  mu<-X%*%beta

  logl= -.5*n*log(2*pi) -.5*n*log(sigma2) - (sum((y-mu)**2)* .5/sigma2)

  return(-logl)
}

```

En nuestro caso en particular:

```

y_reg<-diamonds_num[,1]
X_reg<-cbind(rep(1,length(y_reg)), diamonds_num[,2:7])

```

```

betasigma<-c(1,.79,61.75,57.46,5.731,5.735,35.5, 1)

optim(par=betasigma,lik1, x=X_reg, y=y_reg,method="BFGS")

## $par
## [1] 18205.0228 10894.5542 -182.0883 -79.3253 -963.5743 165.5249
## [7] -732.0578 150309.6275
##
## $value
## [1] 777459.2
##
## $counts
## function gradient
##      156      100
##
## $convergence
## [1] 1
##
## $message
## NULL

reg_precio$coefficients

```

```

## (Intercept)      carat      depth      table        x        y
## 20849.3164 10686.3091 -203.1541 -102.4457 -1315.6678 66.3216
##          z
##      41.6277

```

Ahora, como z es “no significativa” vamos a sacarla del modelo y a correr todo con el nuevo conjunto de variables

```

diamonds_red<-diamonds_num[,-c(7)]

reg_red<-lm(price~carat+depth +table+x+y, data=diamonds_red)

summary(reg_red)

##
## Call:
## lm(formula = price ~ carat + depth + table + x + y, data = diamonds_red)
##
## Residuals:
##      Min      1Q      Median      3Q      Max
## -23872.1   -614.8    -50.5    347.5  12759.4
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20702.947    419.575  49.343 < 2e-16 ***
## carat       10686.707    63.199 169.095 < 2e-16 ***
## depth       -200.718     4.855 -41.344 < 2e-16 ***
## table       -102.490     3.084 -33.234 < 2e-16 ***
## x           -1293.542    36.063 -35.869 < 2e-16 ***
## y            69.575     25.287    2.751  0.00594 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 

```

```
## Residual standard error: 1497 on 53934 degrees of freedom
## Multiple R-squared:  0.8592, Adjusted R-squared:  0.8592
## F-statistic: 6.583e+04 on 5 and 53934 DF,  p-value: < 2.2e-16
```

Ahora con esto vuelvo a definir mis variables a usar en la funcion de max verosimilitud

```
y_reg<-diamonds_red[,1]
X_reg<-cbind(rep(1,length(y_reg)), diamonds_num[,2:6])
betasigma<-c(1,1,1,1,1,1,100)
optim(par=betasigma,lik1, X=X_reg, y=y_reg, method = "BFGS")

## Warning in log(sigma2): NaNs produced

## Warning in log(sigma2): NaNs produced

## Warning in log(sigma2): NaNs produced

## $par
## [1] 27260.5528 10906.0130 -271.6052 -134.3417 -1586.6727 271.2380
## [7] 140486.4120
##
## $value
## [1] 802048.1
##
## $counts
## function gradient
##      159      100
##
## $convergence
## [1] 1
##
## $message
## NULL
```

Podemos ver como los estimadores obtenidos por maxima verosimilitud se parecen mas a los que tienen mayor significancia en el modelo de regresion lineal.