

Modulo 5: Técnicas Avanzadas de Predicción

Modelos Lineales Generalizados

Leandro Gutierrez

19/10/2024

Descripción de la tarea

Dentro del paquete de R “MPV”, se encuentra una base de datos de gasto en combustible de diferentes coches con una serie de características:

- y - Miles/gallon.
- x1 - Displacement (cubic in).
- x2 - Horsepower (ft-lb).
- x3 - Torque (ft-lb).
- x4 - Compression ratio.
- x5 - Rear axle ratio.
- x6 - Carburetor (barrels).
- x7 - No. of transmission speeds.
- x8 - Overall length (in).
- x9 - Width (in).
- x10 - Weight (lb).
- x11 - Type of transmission (1=automatic, 0=manual).

1. Proponed una especificación que a vuestra intuición sea un buen modelo para explicar la variable y en base a las x que tenemos anteriormente.
2. Utilizar la técnica STEPWISE para elegir el modelo de tal forma que minimicemos el BIC.
3. Programad vuestro propio STEPWISE (Backward o Forward) para decidir cuál sería el mejor modelo minimizando la siguiente función:
4. Probad a variar el 0.05 para elegir un modelo según vuestra visión.
5. En función de los modelos anteriores, ¿cuál de ellos en el caso de que difieran recomendaríais?

Solución

Carga de los datos

```
# cargamos el dataset
df_org <- as_tibble(table.b3[-c(23,25),])

# creamos una copia del dataframe original
df <- df_org

# renombramos las columnas
colnames(df) <- c("response", "displacement", "horsepower", "torque", "compression", "rearXratio", "carburetor")
```

```
# visualizamos los datos
summary(df)
```

```
##      response      displacement      horsepower      torque
## Min.   :11.20   Min.    : 85.3   Min.     : 70.0   Min.     : 81.0
## 1st Qu.:16.43   1st Qu.:226.5   1st Qu.:106.0   1st Qu.:171.2
## Median :19.30   Median :318.0   Median :141.5   Median :243.0
## Mean   :20.04   Mean    :286.0   Mean     :137.0   Mean     :217.9
## 3rd Qu.:21.49   3rd Qu.:351.0   3rd Qu.:165.0   3rd Qu.:258.8
## Max.   :36.50   Max.    :500.0   Max.     :223.0   Max.     :366.0
## compression      rearXratio      carburetor      transmissions
## Min.     :8.000   Min.     :2.450   Min.     :1.000   Min.     :3.000
## 1st Qu.:8.000   1st Qu.:2.710   1st Qu.:2.000   1st Qu.:3.000
## Median :8.325   Median :3.000   Median :2.000   Median :3.000
## Mean    :8.313   Mean     :3.059   Mean     :2.567   Mean     :3.333
## 3rd Qu.:8.500   3rd Qu.:3.243   3rd Qu.:4.000   3rd Qu.:3.000
## Max.    :9.000   Max.     :4.300   Max.     :4.000   Max.     :5.000
##      length      width      weight      type
## Min.     :155.7   Min.     :61.80   Min.     :1905   Min.     :0.0000
## 1st Qu.:173.4   1st Qu.:65.78   1st Qu.:3028   1st Qu.:0.2500
## Median :196.1   Median :72.00   Median :3760   Median :1.0000
## Mean    :192.3   Mean     :71.42   Mean     :3626   Mean     :0.7333
## 3rd Qu.:207.1   3rd Qu.:76.30   3rd Qu.:4215   3rd Qu.:1.0000
## Max.    :231.0   Max.     :79.80   Max.     :5430   Max.     :1.0000
```

```
glimpse(df)
```

```
## Rows: 30
## Columns: 12
## $ response      <dbl> 18.90, 17.00, 20.00, 18.25, 20.07, 11.20, 22.12, 21.47, ~
## $ displacement <dbl> 350.0, 350.0, 250.0, 351.0, 225.0, 440.0, 231.0, 262.0, ~
## $ horsepower    <dbl> 165, 170, 105, 143, 95, 215, 110, 110, 70, 75, 155, 80, ~
## $ torque        <dbl> 260, 275, 185, 255, 170, 330, 175, 200, 81, 83, 250, 83, ~
## $ compression   <dbl> 8.00, 8.50, 8.25, 8.00, 8.40, 8.20, 8.00, 8.50, 8.20, 9.~
## $ rearXratio    <dbl> 2.56, 2.56, 2.73, 3.00, 2.76, 2.88, 2.56, 2.56, 3.90, 4.~
## $ carburetor     <dbl> 4, 4, 1, 2, 1, 4, 2, 2, 2, 2, 4, 2, 2, 1, 2, 2, 4, 4, 4, ~
## $ transmissions <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5, 3, 4, 4, 3, 4, 3, 3, 3, ~
## $ length        <dbl> 200.3, 199.6, 196.7, 199.9, 194.1, 184.5, 179.3, 179.3, ~
## $ width         <dbl> 69.9, 72.9, 72.2, 74.0, 71.8, 69.0, 65.4, 65.4, 64.0, 65~
## $ weight        <dbl> 3910, 3860, 3510, 3890, 3365, 4215, 3020, 3180, 1905, 23~
## $ type          <dbl> 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, ~
```

Podemos observar que nuestro dataset sanitizado cuenta con 30 columnas y 12 observaciones. Todas las variables son de tipo cuantitativas continuas y sus tipos de datos intrínsecos son `numeric`. No se observan valores nulos.

Apartado 1

Comenzaremos analizando un modelo lineal con todas las variables predictoras incorporadas, ello nos servirá para notar los efectos marginales de cada variable independiente y sus niveles de significancia

```
# creamos nuestro modelo lineal
modelo_1 <- lm(response ~ ., df)
```

```
# visualizamos el summary del modelo
```

```
pander(summary(modelo_1))
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	17,34	30,36	0,5712	0,5749
displacement	-0,07559	0,05635	-1,341	0,1964
horsepower	-0,06916	0,08779	-0,7878	0,4411
torque	0,1151	0,08811	1,306	0,2078
compression	1,495	3,101	0,4819	0,6357
rearXratio	5,843	3,148	1,856	0,0799
carburetor	0,3176	1,289	0,2464	0,8082
transmissions	-3,205	3,109	-1,031	0,3162
length	0,1808	0,1303	1,388	0,1822
width	-0,3979	0,3235	-1,23	0,2344
weight	-0,005115	0,005896	-0,8675	0,3971
type	0,6385	3,022	0,2113	0,835

Table 2: Fitting linear model: response ~ .

Observations	Residual Std. Error	R^2	Adjusted R^2
30	3,227	0,8355	0,7349

```
# obtenemos el rss del modelo
rss <- sum(residuals(modelo_1)^2)

print(rss)
```

```
## [1] 187.4007
```

```
# graficamos matriz de correlaciones
# cr <- cor(dplyr::select(df, - response), use="complete.obs")
# ggcorrplot(cr, hc.order = TRUE, type = "lower", lab = TRUE)
```

Podemos ver que nuestro modelo parece no estar capturando de manera correcta la naturaleza de nuestra variable dependiente **response**. Lo primero que observamos es bajos niveles de significancia en nuestros predictores, salvo por la variable **rearXratio** que presenta un nivel de significancia de **0.0799**, aún éste lejos del 0.05 que solemos buscar. Además se destaca el efecto marginal del intercepto demasiado alto en comparación con los demás coeficientes, un error standard alto en comparación con su estimación y con un p-value de **0.5749**, indicándonos que su incorporación en el modelo no aporta valor alguno al momento de explicar la variable dependiente.

De mismo modo notamos que obtenemos RSS de **187.4007** y un $R^2_{ajustado}$ de **0.7349**, lo que nos dice nuestro modelo es capaz de explicar el 73.49% de la variabilidad total de **response**.

A continuación intentaremos mejorar nuestro modelo incorporando efectos no lineales de las variables independientes, para ello utilizaremos la función **earth** con un threshold de **0.01**

```
# creamos el modelo
modelo_earth <- earth(response ~ ., df, thresh=0.01)

# visualizamos el summary
summary(modelo_earth)
```

```
## Call: earth(formula=response~., data=df, thresh=0.01)
```

```
##
##              coefficients
## (Intercept)    15.8105232
## h(185.2-length) 0.3747693
## h(length-185.2) 0.3117515
## h(69.4-width)   1.1780878
## h(weight-3510) -0.0084423
##
## Selected 5 of 8 terms, and 3 of 11 predictors
## Termination condition: RSq changed by less than 0.01 at 8 terms
## Importance: length, weight, width, displacement-unused, horsepower-unused, ...
## Number of terms at each degree of interaction: 1 4 (additive model)
## GCV 7.00508    RSS 102.9747    GRSq 0.8276052    RSq 0.9096004
```

```
# calculamos R^2 ajustado
rs <- 0.9096004
p <- length(modelo_earth$coefficients) - 1 # quitamos intercept
n <- nrow(df)
rsa <- 1 - (1 - rs) * ((n - 1) / (n - p - 1))
print(rsa)
```

```
## [1] 0.8951365
```

```
# numero de params
k <- length(coef(modelo_earth))

# numero de observaciones
n <- nrow(df)

# calculamos el BIC
bic <- n * log(rss / n) + k * log(n)

print(bic)
```

```
## [1] 71.96754
```

La función `earth` nos entrega un modelo basado splines, donde seleccionó 5 terminos (los observados en el summary) a partir de los 11 predictores originales. Obtuvimos un *RSS* (**Suma de los Cuadrados de los Residuos**) de **102.9747** disminuyendo respecto al modelo original con todas las variables predictoras incorporadas. Además, nos entrega un $R^2_{ajustado}$ **0.8951** también mejorando el valor obtenido por el modelo original. Y obtuvimos un *BIC* de **71.9675**

Apartado 2

Para este apartado utilizaremos la función `stepAIC` del paquete `MASS` el cual realiza una simplificación de nuestro modelo descartando las variables que generan la menor perdida de información posible, en este caso utilizaremos el metodo hibrido (**both**) para la selección

```
# utilizamos stepAIC para encontrar un modelo simplificado
modelo_aic <- stepAIC(modelo_1, trace=TRUE, direction="forward", scope=respuesta~., k = log(n))

## Start:  AIC=95.78
## response ~ displacement + horsepower + torque + compression +
##          rearXratio + carburetor + transmissions + length + width +
##          weight + type
```

```
# visualizamos la formula del modelo propuesto
pander(formula(modelo_aic))
```

response ~ displacement + horsepower + torque + compression + rearXratio + carburetor + transmissions + length + width + weight + type

```
# vemos summary del modelo propuesto
pander(summary(modelo_aic))
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	17,34	30,36	0,5712	0,5749
displacement	-0,07559	0,05635	-1,341	0,1964
horsepower	-0,06916	0,08779	-0,7878	0,4411
torque	0,1151	0,08811	1,306	0,2078
compression	1,495	3,101	0,4819	0,6357
rearXratio	5,843	3,148	1,856	0,0799
carburetor	0,3176	1,289	0,2464	0,8082
transmissions	-3,205	3,109	-1,031	0,3162
length	0,1808	0,1303	1,388	0,1822
width	-0,3979	0,3235	-1,23	0,2344
weight	-0,005115	0,005896	-0,8675	0,3971
type	0,6385	3,022	0,2113	0,835

Table 4: Fitting linear model: response ~ displacement + horsepower + torque + compression + rearXratio + carburetor + transmissions + length + width + weight + type

Observations	Residual Std. Error	R^2	Adjusted R^2
30	3,227	0,8355	0,7349

```
AIC(modelo_aic)
```

```
## [1] 166.0979
```

```
BIC(modelo_aic)
```

```
## [1] 184.3134
```

Apartado 3

En primer lugar haremos unos pequeños cambios al algoritmo que fué brindado para poder entender como funciona y poder interpretar como afecta en la elección del modelo óptimo el parámetro **k** el cual representa una **penalización por la inclusión de nuevas variables predictoras**.

Agregamos unos prints para poder observar como se van formando los modelos candidatos y como se realiza la selección final en función del criterio elegido

```
stepwise <- function(df, k){
  #Inicializo las variables
  n <- nrow(df)
  m <- ncol(df) - 1
  m_name <- colnames(dplyr::select(df, -response))
  old_m <- rep(NA, length(m_name))
```

```

modelos <- data.frame()
formula_min <- ""
# Bucle para recorrer las posibles variables
for (i in 1:m) {
  U <- c(0)
  ncol <- length(m_name)

  for (m_var in 1:ncol){
    remaining_var <- paste0(m_name[m_var], collapse="+")
    formula_str <- remaining_var

    if (formula_min != "") {
      formula_str <- paste(formula_min, remaining_var, sep = "+")
    }

    # Creo un modelo
    formula_i <- as.formula(paste0("response~", formula_str))
    mod_i <- glm(formula=formula_i, data=df, family = gaussian)
    m_num <- length(mod_i$coefficients) - 1
    pred <- predict(mod_i, df, type="response")

    # Formula a minimizar
    U[m_var] <- (sum((df$response - pred)**2))**0.5 / ((sum(df$response**2))**0.5 + (sum(pred**2))**0.5)

  }

  # Almaceno el resultado
  Umin <- which.min(U)
  old_m[i] <- m_name[Umin]
  m_name <- m_name[-Umin]
  formula_min <- paste0(old_m[(!is.na(old_m))], collapse="+")

  modelos[i,1] <- formula_min
  modelos[i,2] <- U[Umin]

  cat(paste0("Mejor modelo con p = ", i, ":\n ", formula_min, "\n"))
  U[Umin]

}

return(modelos)
}

```

```
k=0.005
```

```
modelos <- stepwise(df, k)
```

```

## Mejor modelo con p = 1:
## displacement
## Mejor modelo con p = 2:
## displacement+compression
## Mejor modelo con p = 3:
## displacement+compression+width
## Mejor modelo con p = 4:
## displacement+compression+width+length
## Mejor modelo con p = 5:

```

```
## displacement+compression+width+length+weight
## Mejor modelo con p = 6:
## displacement+compression+width+length+weight+rearXratio
## Mejor modelo con p = 7:
## displacement+compression+width+length+weight+rearXratio+transmissions
## Mejor modelo con p = 8:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque
## Mejor modelo con p = 9:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
## Mejor modelo con p = 10:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor
## Mejor modelo con p = 11:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor+type
pander(modelos, split.table=TRUE)
```

Table 5: Table continues below

V1
displacement
displacement+compression
displacement+compression+width
displacement+compression+width+length
displacement+compression+width+length+weight
displacement+compression+width+length+weight+rearXratio
displacement+compression+width+length+weight+rearXratio+transmissions
displacement+compression+width+length+weight+rearXratio+transmissions+torque
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor+type

V2
0,07727
0,07974
0,08344
0,08748
0,09152
0,09427
0,09788
0,1011
0,105
0,1099
0,1148

```
cat(paste0("El mejor modelo con k = ", k, ": ", modelos[which.min(modelos$V2), c("V1")]))
```

```
## El mejor modelo con k = 0.005: displacement
```

Con una penalización por la inclusión de nuevas variables (k) igual a **0.005** obtenemos un modelo muy selectivo, con solo una variable predictora en su fórmula, **displacement**, y un valor para la función a minimizar de **0.07727**.

Apartado 4

A continuación modificaremos los valores de k para ver como afecta éste parámetro a la selección del modelo óptimo

```
k=0.002

modelos <- stepwise(df, k)

## Mejor modelo con p = 1:
## displacement
## Mejor modelo con p = 2:
## displacement+compression
## Mejor modelo con p = 3:
## displacement+compression+width
## Mejor modelo con p = 4:
## displacement+compression+width+length
## Mejor modelo con p = 5:
## displacement+compression+width+length+weight
## Mejor modelo con p = 6:
## displacement+compression+width+length+weight+rearXratio
## Mejor modelo con p = 7:
## displacement+compression+width+length+weight+rearXratio+transmissions
## Mejor modelo con p = 8:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque
## Mejor modelo con p = 9:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
## Mejor modelo con p = 10:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor
## Mejor modelo con p = 11:
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor+type

pander(modelos, split.table=TRUE)
```

Table 7: Table continues below

V1
displacement
displacement+compression
displacement+compression+width
displacement+compression+width+length
displacement+compression+width+length+weight
displacement+compression+width+length+weight+rearXratio
displacement+compression+width+length+weight+rearXratio+transmissions
displacement+compression+width+length+weight+rearXratio+transmissions+torque
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor+type

V2
0,07427
0,07374
0,07444

V2
0,07548
0,07652
0,07627
0,07688
0,07708
0,07798
0,0799
0,08182

```
print(paste0("El mejor modelo con k = ", k, ": ", modelos[which.min(modelos$V2), c("V1")]))
```

```
## [1] "El mejor modelo con k = 0.002: displacement+compression"
```

Notamos que al tomar el valor $k = 0.002$ el mejor modelo lo obtenemos con dos variables predictoras `displacement+compression` y un valor de **0.07374** para el criterio U (resultado de la función a minimizar).

Si por último elegimos una penalización muy pequeña por la inclusión de variables predictoras obtendremos los siguientes resultados

```
k=0.001
```

```
modelos <- stepwise(df, k)
```

```
## Mejor modelo con p = 1:
```

```
## displacement
```

```
## Mejor modelo con p = 2:
```

```
## displacement+compression
```

```
## Mejor modelo con p = 3:
```

```
## displacement+compression+width
```

```
## Mejor modelo con p = 4:
```

```
## displacement+compression+width+length
```

```
## Mejor modelo con p = 5:
```

```
## displacement+compression+width+length+weight
```

```
## Mejor modelo con p = 6:
```

```
## displacement+compression+width+length+weight+rearXratio
```

```
## Mejor modelo con p = 7:
```

```
## displacement+compression+width+length+weight+rearXratio+transmissions
```

```
## Mejor modelo con p = 8:
```

```
## displacement+compression+width+length+weight+rearXratio+transmissions+torque
```

```
## Mejor modelo con p = 9:
```

```
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
```

```
## Mejor modelo con p = 10:
```

```
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor
```

```
## Mejor modelo con p = 11:
```

```
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor+
```

```
pander(modelos, split.table=TRUE)
```

Table 9: Table continues below

V1
displacement
displacement+compression

V1
displacement+compression+width
displacement+compression+width+length
displacement+compression+width+length+weight
displacement+compression+width+length+weight+rearXratio
displacement+compression+width+length+weight+rearXratio+transmissions
displacement+compression+width+length+weight+rearXratio+transmissions+torque
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor
displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower+carburetor+type

V2
0,07327
0,07174
0,07144
0,07148
0,07152
0,07027
0,06988
0,06908
0,06898
0,0699
0,07082

```
cat(paste0("El mejor modelo con k = ", k, "\n", modelos[which.min(modelos$V2), c("V1")]))
```

```
## El mejor modelo con k = 0.001:
```

```
## displacement+compression+width+length+weight+rearXratio+transmissions+torque+horsepower
```

Para el valor de $k = 0.001$ el mejor modelo contiene en su formula 9 de las 11 variables predictoras disponibles y la función a minimizar toma un valor de **0.06898**.

A esta altura es necesario destacar que cuanto menos se penaliza la inclusión de variables explicativas nuestro algoritmo de selección tiende a agregar todas los predictores disponibles en búsqueda del modelo que minimice la diferencia entre la respuesta real y la predicha por nuestra regresión lineal multiple. Al mismo tiempo se nota una mejora en el criterio seleccionado (función U) a medida que mas variables forman parte del modelo.

Apartado 5