Análisis de Regresión con R
Introducción - Grado en Inteligencia Artificial

Índice

En esta práctica repasaremos lo comandos básicos de R para ajustar un modelo de regresion lineal. Además de la estimación de los parámetros del modelo, realizaremos los contrastes estadísticos oportunos para verificar la validez del modelo propuesto. Utilizaremos como ejemplo los datos del fichero Advertising.csv.

El conjunto de datos de publicidad consiste en las ventas de ese producto en diferentes mercados, junto con los publicidad del producto en cada uno de esos mercados para tres medios diferentes: TV, radio y prensa. Supongamos que un cliente nos contrata para asesoramiento sobre cómo mejorar las ventas producto.

Puesto que el problema de estimación de los parámetros de un modelo de regresión lineal es un problema de optimización convexa sin restricciones, compararemos los valores de los parámetros ajustados con los que obtendríamos aplicando un método iterativo de optimización como el método de descenso de gradiente. También veremos un método alternativo al método de descenso de gradiente, denominado método de descenso de gradiente estocástico (Stochastic Gradient Descent).

1. Ajuste de un modelo de regresión lineal con R

El fichero Advertising.csv contiene información sobre las ventas de un producto en 200 mercados diferentes, junto con la inversión en publicidad de dichos mercados en distintos medios. En primer lugar, importa los datos y realiza una representación gráfica que te permita visualizar la posible relación entre las ventas y la inversión en publicidad en distintos medios.

Vamos a asumir que la variable Y (`Sales`) depende linealmente de tres variables X_1 (`TV`), X_2 (`Radio`)

y X_3 (`Newspaper`), es decir, planteamos el modelo

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon.$$

Para realizar un ajuste lineal en R, utilizaremos la función `lm`. El argumento principal de la función `lm` es una **fórmula** de R. En nuestro caso escribiremos:

```
> z <- lm(Advertising$Sales ~ Advertising$TV + Advertising$Radio + Advertising$Newspaper)
```

De forma equivalente, podríamos escribir:

```
> z <- lm(Sales ~ TV + Radio + Newspaper, data = Advertising)
```

Observa que no necesitamos especificar en la fórmula el término independiente, ya que éste se incluye por defecto. La función `lm` devuelve un objeto de tipo `lm` con varias componentes.

```
> class(z)
```

```
## [1] "lm"
```

```
> names(z)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

Podemos resumir los resultados del ajuste con la funcion `summary`

```
> summary(z)
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio + Newspaper, data = Advertising)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8277 -0.8908  0.2418  1.1893  2.8292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.938889   0.311908   9.422   <2e-16 ***
## TV           0.045765   0.001395  32.809   <2e-16 ***
## Radio        0.188530   0.008611  21.893   <2e-16 ***
## Newspaper   -0.001037   0.005871  -0.177     0.86
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.686 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

1.1. Estimación de los parámetros del modelo

Para obtener los coeficientes estimados del modelo, podemos utilizar la función `coef`

```
> coef(z)
```

```
##  (Intercept)           TV        Radio    Newspaper
##  2.938889369  0.045764645  0.188530017 -0.001037493
```

El resultado es equivalente a:

```
> z$coefficients
```

```
##  (Intercept)           TV        Radio    Newspaper
##  2.938889369  0.045764645  0.188530017 -0.001037493
```

Comprueba que los coeficientes ajustados por el método de mínimos cuadrados se obtienen como[1]:

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{X}^t\mathbf{X}\right)^{-1}\mathbf{X}^t\mathbf{y}$$

donde, en nuestro ejemplo,

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \ \mathbf{X} = \begin{pmatrix} 1 & x_{11} & x_{12} & x_{13} \\ 1 & x_{21} & x_{22} & x_{23} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & x_{n3} \end{pmatrix}.$$

Para obtener los valores ajustados por el modelo de regresión lineal, $\hat{y}_i$, usaremos la función `fitted`

```
> fitted(z)
```

Podemos obtener la misma información escribiendo `z$fitted.values`. Los residuos del modelo, $\hat{\epsilon}_i$, serán por lo tanto

```
> Advertising$Sales - z$fitted.values
```

Equivalentemente,

[1] La función `solve` calcula la inversa de una matriz y la función `t` calcula la traspuesta

```
> residuals(z)
```

En el modelo de regresión lineal múltiple, el RSE (residual standard error) se calcula como:

$$RSE = \sqrt{\frac{RSS}{n-p-1}}.$$

Se trata de una estimación de la desviación típica del error. Calcula el RSE a partir de los residuos del modelo y comprueba que obtienes el mismo valor que se muestra al usar la función `summary`.

Podremos obtener intervalos de confianza para los coeficientes del modelo con la función `confint`. Si no se especifica el argumento `level`, los intervalos se calculan con una confianza $1 - \alpha = 0.95$.

```
> confint(z, level = 0.9)
```

```
##                    5 %        95 %
## (Intercept)  2.42340953 3.454369213
## TV           0.04345935 0.048069943
## Radio        0.17429853 0.202761502
## Newspaper   -0.01074031 0.008665319
```

1.2. Contrastes sobre los parámetros del modelo

Analiza los resultados de los contrastes sobre los parámetros del modelo que devuelve la función `summary`. Observa que de los resultados se desprende (test F) que al menos una de las variables predictoras es útil para predecir la respuesta Y. Además, si nos fijamos en los contrastes individuales, parece que la variable `Newspaper` no es significativa. Ajusta un modelo lineal que explique la variable `Sales` como función de `TV` y `Radio` y analiza si se observa una pérdida importante en el ajuste del modelo (coeficiente R^2). Como verás, parece que el modelo que explica la variable `Sales` como función de `TV` y `Radio` es más razonable.

```
> z2 <- lm(Sales ~ TV + Radio, data = Advertising)
```

1.3. Predicción

Una vez fijado el modelo que explica la variable `Sales` como función de `TV` y `Radio`, podemos hacer predicciones como se muestra a continuación. Por ejemplo, si queremos predecir las ventas en un comercio que invierte 20000$ en publicidad en radio y 100000$ en publicidad en TV, escribiremos:

```
> newdata = data.frame(TV = 100, Radio = 20)
> predict(z2, newdata)
```

```
##        1
## 11.25647
```

A continuación, se muestra el intervalo de confianza para la venta media en mercados con una inversión de 20000$ en publicidad en radio y 100000$ en publicidad en TV. En segundo lugar se muestra el intervalo de predicción para la venta en un mercado que invierte 20000$ en publicidad en radio y 100000$ en publicidad en TV. Observa que el intervalo para la predicción es mayor.

```
> newdata = data.frame(TV = 100, Radio = 20)
> predict(z2, newdata, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 11.25647 10.98525 11.52768
```

```
> predict(z2, newdata, interval = "predict")
```

```
##        fit      lwr      upr
## 1 11.25647 7.929616 14.58332
```

2. Ajuste de los parámetros de un modelo de regresión lineal mediante métodos de optimización

Consideremos de nuevo el modelo de regresión lineal

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \epsilon.$$

Recordamos que, dada una muestra de entrenamiento $(y_1, x_{11}, \ldots, x_{1p}), \ldots, (y_n, x_{n1}, \ldots, x_{np})$, los parametros estimados del modelo se obtienen minimizando la función:

$$RSS(\beta_0, \beta_1, \ldots, \beta_p) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_{i1} - \ldots - \beta_p x_{ip})^2. \tag{2.1}$$

El problema de estimación de los parámetros se puede ver, por lo tanto, como un problema de optimización convexa sin restricciones. Podremos aproximar su solución mediante un método de optimización como, por ejemplo, el método de descenso de gradiente.

2.1. Método de descenso de gradiente

Para simplificar el problema, consideraremos un modelo de regresión lineal simple:

$$Y = \beta_0 + \beta_1 X_1 + \epsilon.$$

En primer lugar, vamos a simular una muestra con $n = 100$ observaciones, (y_i, x_i), de un modelo de regresión lineal simple con parámetros β_0 y β_1. Para ello, generamos en primer lugar los valores x_i a

partir de una distribución uniforme. A continuación generamos los errores del modelo, ϵ_i, a partir de una distribución normal de media 0 y varianza σ^2. Los valores y_i se calcularán entonces como

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i.$$

En R, haremos:

```
> n <- 100
> x <- runif(n, min = 0, max = 5)  # x_i: n puntos aleatorios en el itervalo [min,max]
> beta0 <- 2  # Parámetro beta0 del modelo
> beta1 <- 5  # Parámetro beta1 del modelo
> epsilon <- rnorm(n, sd = 1)  # error (con desviación típica sd=1)
> y <- beta0 + beta1 * x + epsilon  # y_i
```

Puedes hacer un diagrama de dispersión de la muestra de entrenamiento generada y ver el efecto de diferentes valores del parámetro `sd`. Calcula el valor de los parámetros estimados con la función `lm`.

Veremos ahora como aproximar los parámetros del modelo mediante el método de descenso de gradiente (también conocido como batch gradient descent). Para ello debemos minimizar la función:

$$J(\beta_0, \beta_1) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2.$$

Observa que este problema es equivalente a (2.1). El factor $1/2$ es únicamente para simplificar la notación del método de descenso de gradiente.

Comprueba que el método de gradiente para minimizar la función $J(\beta_0, \beta_1)$ se puede escribir:

Algoritmo: Método de descenso de gradiente para $J(\beta_0, \beta_1)$

Dado $(\hat{\beta}_0, \hat{\beta}_1)$ y $t > 0$

repite

$\hat{\beta}_0 = \hat{\beta}_0 + t\sum_{i=1}^{n}(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)$

$\hat{\beta}_1 = \hat{\beta}_1 + t\sum_{i=1}^{n} x_i(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)$

hasta que se cumpla el criterio de parada

Prográmalo en R y compara los resultados obtenidos con los que devuelve la función `lm`. Recuerda que el método de descenso de gradiente converge (siempre que el paso t no sea demasiado grande).

2.2. Método de descenso de gradiente estocástico (Stochastic Gradient Descent)

Observa que en el método de descenso de gradiente, para cada actualización del valor de los parámetros estimados, se necesita la muestra de entrenamiento al completo. Existe una alternativa al método de descenso de gradiente que también proporciona buenos resultados. El algoritmo para el caso particular del modelo de regresión lineal simple es el siguiente:

Algoritmo: Método de descenso de gradiente estocástico para $J(\beta_0, \beta_1)$

Dado $(\hat{\beta}_0, \hat{\beta}_1)$ y $t > 0$

repite

  repite para cada $i = 1, \ldots, n$

  $\hat{\beta}_0 = \hat{\beta}_0 + t(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)$

  $\hat{\beta}_1 = \hat{\beta}_1 + t x_i(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)$

hasta que se cumpla el criterio de parada

Observa que con este método se actualiza el valor de los parámetros con cada observación de la muestra de entrenamiento. Mientras que el método de descenso de gradiente (batch) necesita leer todos los datos para hacer una única iteración (lo cual puede ser costoso si n es muy grande), el método de descenso de gradiente estocástico puede empezar a iterar desde la primera observación. Programa este método y compara los resultados con los obtenidos por el método de descenso de gradiente. Puedes representar gráficamente las iteraciones en un gráfico de contorno (curvas de nivel).