Análisis de regresión

Capítulo 10: Regresión y fundamentos del aprendizaje automático

Mg. Sc. J. Eduardo Gamboa U.

2026-10-02

Introducción

Cambio de paradigma

- Regresión clásica
  - Interés en parámetros
  - Inferencia
  - Intervalos de confianza
  - Pruebas de hipótesis
  - Verificación de supuestos
- Machine Learning
  - Interés en predicción
  - Generalización
  - Error **fuera de la muestra**
  - Robustez

Partición de datos

Holdout

- División generalmente 70 - 30, 80 -20 o 90-10
- Condiciones: aleatoria, reproducible, sin fuga de información (data leakage)

Métricas de evaluación

Error cuadrático medio

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Función de pérdida estándar en muchos algoritmos
- Mayor penalización
- Derivable
- Sensible a outliers
- Poco interpretable

Raíz del error cuadrático medio

$$RMSE = \sqrt{MSE}$$

- Misma unidad que la variable respuesta, facilita la comunicación
- Muy usada como métrica de comparación en validación/test.
- Mantiene penalización fuerte de errores grandes
- Sensible a outliers7
- Variación aceptable train - test de 10% a 15% como máximo

Error absoluto medio

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- Robusto frente a outliers
- Alineada con objetivos del tipo “error promedio real”
- No es derivable
- Penalización lineal
- Variación aceptable train - test de 10% a 15% como máximo

R^2 predictivo

$$R^2_{pred} = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \overline{y}_{train})^2}$$

- En training se utiliza el R^2 clásico (ajustado), mientras que en testing se determina el R^2 predictivo.
- Valores cercanos a 1 implican una buena generalización.
- Valores positivos cercanos a 0 implican que el desempeño es similar a predecir con la media del train.
- Valores negativos implican que el desempeño es peor que predecir con la media del train. El modelo falla.
- Aceptable dentro de 0.05–0.10 del train

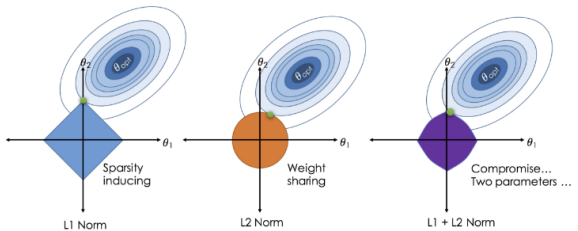
Sobreajuste

- Un modelo demasiado complejo lleva a sobreajuste del mismo.
- Pérdida de la capacidad de generalización.
- Bajo error en train y alto en testing
- Causas:
  - Modelos polinomiales mal especificados
  - Variables no relevantes en el modelo
  - Interacciones sin sustento

- Prevención
  - Mayor tamaño de muestra
  - Selección de variables
  - Uso de polinomios ortogonales
  - Regularización

Regularización

- Controla la complejidad directamente en el proceso de estimación.



Métodos de regularización

- Ridge: reduce la varianza de los coeficientes, no los llega a eliminar.
- Lasso: induce coeficientes exactamente nulos, realizando así una selección automática.
- Elastic net: combina Ridge y Lasso mediante un coeficiente α. Es útil cuando hay alta correlación entre predictores.

Regresión Ridge

- El vector de coeficientes de regresión se estima mediante $\widehat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$
- La matriz $\mathbf{X}'\mathbf{X}$ puede presentar problemas de inversión si las variable están muy correlacionadas (y es necesario descartar alguna(s))
- Se propone el estimador $\widehat{\beta}_\lambda$, conocido como "estimador Ridge", el cual es sesgado pero menos variable que $\widehat{\beta}$.

$$\widehat{\beta}_\lambda = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{Y}$$

- Este estimador se obtiene al minimizar:

$$\sum_{i=1}^{n}(y_i - \mathbf{x}_i'\beta)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

Regresión Lasso

- Se realiza un cambio en la penalidad, de modo que el estimador Lasso se obtiene al minimizar

$$\sum_{i=1}^{n} (y_i - \mathbf{x}_i'\beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

- Tanto en la regresión Ridge como Lasso, los coeficientes de regresión no son interpretables (son sesgados) sin embargo sí es de interés obtener las predicciones.
- La regresión Lasso automáticamente realiza una selección de variables

Regresión Elastic Net

- Es una mezcla de las regresiones Ridge y Lasso, ya que busca minimizar:

$$\sum_{i=1}^{n}(y_i - \mathbf{x}_i'\beta)^2 + \alpha\lambda_1\sum_{j=1}^{p}|\beta_j| + (1-\alpha)\lambda_2\sum_{j=1}^{p}\beta_j^2$$

Solo con la finalidad de aplicar lo visto en clase, vamos a generar o simular un conjunto de datos:

```
set.seed(10)
n <- 200

area       <- runif(n,40,200) |> round(2)
equipos    <- (0.25*area + rnorm(n,0,5)) |> round(0)
horas      <- runif(n,6,16) |> round(1)
eficiencia <- runif(n,0,1) |> round(2)
antig      <- (runif(n,1,30) |> round(0)) - 10*eficiencia
antig      <- ifelse(antig<0,0,antig)

y <- 50 + 0.8*area + 3*equipos + 2.5*horas - 10*eficiencia + rnorm(n,0,15)

datos      <- data.frame(y,area,equipos,horas,eficiencia,antig)
```

```r
set.seed(1)
id    <- sample(1:n,160)
train <- datos[id,]
test  <- datos[-id,]
```

```
train |> head()
```

```
           y   area equipos horas eficiencia antig
68  237.8184 111.60      21  11.4       0.89   0.0
167 315.6165 162.11      41   8.9       0.80   4.0
129 346.4940 177.10      47   7.7       0.49   8.1
162 275.1718 122.62      33  11.0       0.45   1.5
43  110.6787  42.31       1  14.3       0.89   5.1
14  267.1220 135.35      32  11.9       0.68   1.2
```

```
test |> head()
```

```
         y   area equipos horas eficiencia antig
3 206.4967 108.31      22  13.2       0.88   0.0
4 325.4285 150.90      41  11.5       0.18  14.2
5 155.4318  53.62      10   8.6       0.99   7.1
6 174.8376  76.07      22   9.1       0.66   4.4
8 148.9692  83.57      13   7.8       0.31   3.9
9 315.0447 138.53      40  11.5       0.23  21.7
```

```r
mod1 <- lm(y ~ ., train)
mod1 |> summary()
```

```
Call:
lm(formula = y ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-43.810  -8.632  -1.668   8.974  37.081

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  50.956685   5.815923   8.762 3.28e-15 ***
area          0.801696   0.063355  12.654  < 2e-16 ***
equipos       2.952934   0.230197  12.828  < 2e-16 ***
horas         2.439309   0.370542   6.583 6.84e-10 ***
eficiencia  -10.677379   3.886344  -2.747  0.00672 **
antig        -0.007581   0.134004  -0.057  0.95496
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.81 on 154 degrees of freedom
Multiple R-squared:  0.9626,    Adjusted R-squared:  0.9614
F-statistic: 793.4 on 5 and 154 DF,  p-value: < 2.2e-16
```

```r
library(car)
mod1 |> vif()
```

```
      area    equipos      horas eficiencia      antig
  6.363948   6.445116   1.007467   1.077785   1.113653
```

```r
pred_mod1 <- predict(mod1,test)
pred_mod1 |> as.vector()
```

```
 [1] 225.5557 319.0254 133.8266 192.0235 172.0296 305.5647 298.6336 213.7393
 [9] 339.5655 215.7780 171.8059 382.9219 140.3886 358.8467 245.7746 159.1674
[17] 250.5458 167.7351 258.5965 320.6813 300.0730 190.5805 254.6541 296.0384
[25] 122.1684 183.0899 211.0512 278.2194 245.1004 216.7238 304.1779 249.9813
[33] 375.6827 131.8134 183.6351 350.7597 218.3396 262.5787 180.4514 331.9510
```

```r
rtrain1 <- data.frame(real = train$y,
                      pred = predict(mod1, train))

(rmse_train <- sqrt(mean((rtrain1$real - rtrain1$pred)^2)))
```

```
[1] 13.54508
```

```r
(mae_train  <- mean(abs(rtrain1$real - rtrain1$pred)))
```

```
[1] 10.75111
```

```r
(R2_pred_train = 1 - sum((rtrain1$real - rtrain1$pred)^2) / sum((rtrain1$real - mean(train$y))^2))
```

```
[1] 0.9626322
```

```r
rtest1 <- data.frame(real  = test$y,
                     pred  = predict(mod1,test))

(rmse_ols <- sqrt(mean((rtest1$real - rtest1$pred)^2)))
```

```
[1] 18.72747
```

```r
(mae_ols  <- mean(abs(rtest1$real - rtest1$pred)))
```

```
[1] 14.88706
```

```r
(R2_pred = 1 - sum((rtest1$real - rtest1$pred)^2) / sum((rtest1$real - mean(train$y))^2))
```

```
[1] 0.9379136
```

```r
library(yardstick)
rtrain1 |> metrics(real, pred)
```

```
# A tibble: 3 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard      13.5
2 rsq     standard       0.963
3 mae     standard      10.8
```

```r
rtest1 |> metrics(real, pred)
```

```
# A tibble: 3 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard      18.7
2 rsq     standard       0.938
3 mae     standard      14.9
```

```r
library(ModelMetrics)

(mse_train  <- mse(rtrain1$real, rtrain1$pred))
```

```
[1] 183.4693
```

```r
(rmse_train <- rmse(rtrain1$real, rtrain1$pred))
```

```
[1] 13.54508
```

```r
(mae_train  <- mae(rtrain1$real, rtrain1$pred))
```

```
[1] 10.75111
```

```r
(mse_test  <- mse(rtest1$real, rtest1$pred))
```

```
[1] 350.7181
```

```r
(rmse_test <- rmse(rtest1$real, rtest1$pred))
```

```
[1] 18.72747
```

```r
(mae_test  <- mae(rtest1$real, rtest1$pred))
```

```
[1] 14.88706
```

```r
library(caret)

postResample(rtrain1$pred, rtrain1$real)
```

```
      RMSE   Rsquared        MAE
13.5450832  0.9626322 10.7511063
```

```r
(cor(rtrain1$pred, rtrain1$real))^2
```

```
[1] 0.9626322
```

```r
postResample(rtest1$pred, rtest1$real)
```

```
      RMSE   Rsquared        MAE
18.7274692  0.9375058 14.8870566
```

```r
(cor(rtest1$pred, rtest1$real))^2
```

```
[1] 0.9375058
```

```r
mod2 <- lm(y ~ area + horas + eficiencia + antig, train)
mod2 |> summary()
```

```
Call:
lm(formula = y ~ area + horas + eficiencia + antig, data = train)

Residuals:
   Min     1Q Median     3Q    Max
-51.80 -10.10   1.63  13.90  50.33

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 49.67643    8.33644   5.959 1.65e-08 ***
area         1.54733    0.03613  42.827  < 2e-16 ***
horas        2.09231    0.52979   3.949 0.000119 ***
eficiencia  -9.55018    5.57001  -1.715 0.088423 .
antig        0.31446    0.18871   1.666 0.097649 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 19.79 on 155 degrees of freedom
Multiple R-squared:  0.9227,    Adjusted R-squared:  0.9207
F-statistic: 462.6 on 4 and 155 DF,  p-value: < 2.2e-16
```

```
rtrain2 <- data.frame(real = train$y,
                      pred = predict(mod2, train))
postResample(rtrain2$pred, rtrain2$real)
```

```
      RMSE   Rsquared        MAE
19.4810891  0.9227033 15.4683283
```

```
rtest2    <- data.frame(real  = test$y,
                        pred  = predict(mod2,test))
postResample(rtest2$pred, rtest2$real)
```

```
      RMSE   Rsquared        MAE
21.9153883  0.9153784 16.9787321
```

```
library(olsrr)
mod1 |> ols_step_both_p()
```

```r
mod3 <- lm(y ~ area + equipos + horas + eficiencia, train)
mod3 |> summary()
```

```
Call:
lm(formula = y ~ area + equipos + horas + eficiencia, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-43.834  -8.589  -1.620   8.962  37.156

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  50.85244    5.49852   9.248  < 2e-16 ***
area          0.80225    0.06238  12.860  < 2e-16 ***
equipos       2.95049    0.22539  13.090  < 2e-16 ***
horas         2.43907    0.36932   6.604 6.04e-10 ***
eficiencia  -10.61984    3.73878  -2.840  0.00511 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.76 on 155 degrees of freedom
Multiple R-squared:  0.9626,    Adjusted R-squared:  0.9617
F-statistic: 998.2 on 4 and 155 DF,  p-value: < 2.2e-16
```

```
rtrain3 <- data.frame(real = train$y,
                      pred = predict(mod3, train))
postResample(rtrain3$pred, rtrain3$real)
```

```
      RMSE   Rsquared        MAE
13.5452239  0.9626314 10.7474439
```

```
rtest3    <- data.frame(real  = test$y,
                        pred  = predict(mod3,test))
postResample(rtest3$pred, rtest3$real)
```

```
     RMSE  Rsquared       MAE
18.726266  0.937517 14.884317
```

Ridge: Construcción del modelo en training

```
library(glmnet)
X <- model.matrix(y~.,train)[,-1]
ytr <- train$y
cv_ridge <- cv.glmnet(X,ytr,alpha=0)
```

Ridge: Evaluación training vs testing

```
Xtrain  <- model.matrix(y~.,train)[,-1]
rtrain4 <- data.frame(real = train$y,
                      pred = predict(cv_ridge, Xtrain))
postResample(rtrain4$lambda.1se, rtrain4$real)
```

```
      RMSE   Rsquared        MAE
14.7015112  0.9624073 11.5626441
```

```
Xtest     <- model.matrix(y~.,test)[,-1]
rtest4    <- data.frame(real  = test$y,
                        pred  = predict(cv_ridge, Xtest))
postResample(rtest4$lambda.1se, rtest4$real)
```

```
     RMSE  Rsquared       MAE
19.972784  0.937703 15.766761
```

Lasso: Construcción del modelo en training

```
X         <- model.matrix(y~.,train)[,-1]
ytr       <- train$y
cv_lasso <- cv.glmnet(X,ytr,alpha=1)
```

Lasso: Evaluación training vs testing

```
Xtrain    <- model.matrix(y~.,train)[,-1]
rtrain5   <- data.frame(real = train$y,
                        pred = predict(cv_lasso, Xtrain))
postResample(rtrain5$lambda.1se, rtrain5$real)
```

```
      RMSE   Rsquared        MAE
13.9963365  0.9610603 10.8901951
```

```
Xtest     <- model.matrix(y~.,test)[,-1]
rtest5    <- data.frame(real  = test$y,
                        pred  = predict(cv_lasso, Xtest))
postResample(rtest5$lambda.1se, rtest5$real)
```

```
     RMSE  Rsquared       MAE
19.233899  0.936381 15.612121
```

Elastic Net: Construcción del modelo en training

```
X         <- model.matrix(y~.,train)[,-1]
ytr       <- train$y
cv_enet   <- cv.glmnet(X,ytr,alpha=0.5)
```

Lasso: Evaluación training vs testing

```
Xtrain    <- model.matrix(y~.,train)[,-1]
rtrain6   <- data.frame(real = train$y,
                        pred = predict(cv_enet, Xtrain))
postResample(rtrain6$lambda.1se, rtrain6$real)
```

```
      RMSE   Rsquared        MAE
14.3318335  0.9605474 11.1200072
```

```
Xtest     <- model.matrix(y~.,test)[,-1]
rtest6    <- data.frame(real  = test$y,
                        pred  = predict(cv_enet, Xtest))
postResample(rtest6$lambda.1se, rtest6$real)
```

```
     RMSE  Rsquared       MAE
19.595944  0.936061 15.827343
```

```
postResample(rtest1$pred, rtest1$real)
```

```
      RMSE   Rsquared        MAE 
18.7274692  0.9375058 14.8870566 
```

```
postResample(rtest2$pred, rtest2$real)
```

```
      RMSE   Rsquared        MAE 
21.9153883  0.9153784 16.9787321 
```

```
postResample(rtest3$pred, rtest3$real)
```

```
     RMSE  Rsquared       MAE 
18.726266  0.937517 14.884317 
```

```
postResample(rtest4$lambda.1se, rtest4$real)
```

```
     RMSE  Rsquared       MAE 
19.972784  0.937703 15.766761 
```

```
postResample(rtest5$lambda.1se, rtest5$real)
```

```
     RMSE  Rsquared       MAE 
19.233899  0.936381 15.612121 
```

```
postResample(rtest6$lambda.1se, rtest6$real)
```

```
     RMSE  Rsquared       MAE 
19.595944  0.936061 15.827343 
```