Regresión

Una herramienta que nos permite observar las relaciones entre variables, teniendo en cuenta que, por una parte, tendremos a lo que llamaremos una “variable de respuesta”, la cual será la variable objetivo que querremos predecir, y por otro lado, tendremos a las llamadas “variables predictoras”, las cuales, en base a su valor, intentarán dar con el valor de la variable de respuesta. Es por esto que, muchas veces, diremos que vamos a predecir la variable objetivo con algunas variables del conjunto de datos.

De esta forma, tendremos 2 clasificaciones de regresiones, las que dependerán de la distribución que sigue la variable de respuesta (en estas tendremos las regresiones lineales y logísticas) y las que dependerán del número de variables predictoras (para 1 variable predictora la llamaremos simple, y para 2 o más, la llamaremos múltiple).

Regresión Lineal Simple (RLS):

Buscamos observar el comportamiento entre una variable de respuesta junto a una variable predictora, las cuales tendrán una relación lineal, es decir, tendremos a lo que conoceríamos una ecuación lineal del tipo y = m*x + b (según el libro, se indica como y = beta_0 + beta_1*x), donde m es la pendiente y b el intercepto.

Residuos:

Nos referiremos a “residuos” a la diferencia entre el valor esperado de “y” en la ecuación lineal frente al valor obtenido a partir de los datos, por tanto, si el residuo es igual a 0, se refiere que ambos valores serán iguales, en caso de no ser 0, significa que existe una discrepancia. De esta forma, podríamos observar todos los posibles residuos desde todos los valores que se puedan llegar a obtener, y de lo cual, podríamos observar su distribución y su variabilidad.

Correlación:

Otro concepto clave es la “correlación”, donde se define como la fuerza de una relación lineal entre dos variables, por lo cual podemos cuantificar esta magnitud. Existen diversas formas, las cuales dependerán del tipo de datos con los que trabajaremos, pero para este curso sólo necesitan conocer el coeficiente de correlación de Pearson y la denotaremos como “R”, la cual trabaja sobre variables numéricas continuas. En R, éste valor lo podemos conocer utilizando la función cor(data), la cual entregará una matriz con todas las correlaciones entre todas las combinaciones de pares posibles.

Siguiendo con la RLS:

La relación lineal la construiremos mediante la línea de mínimos cuadrados, la cual minimiza la suma de los cuadrados de los residuos. Para utilizar esto, tendremos que tener presente ciertas condiciones a cumplir:

1. Los datos deben presentar una relación lineal.
2. La distribución de los residuos debe ser cercana a la normal.
3. La variabilidad de los puntos en torno a la línea de mínimos cuadrados debe ser aproximadamente constante.

4. Las observaciones deben ser independientes entre sí. Esto significa que no se puede usar regresión lineal con series de tiempo (tema que va más allá de los alcances de este texto).

La primera se cumple, siempre y cuando, en la gráfica de las dos variables (respuesta y predictora), éstas sigan una relación lineal, es decir, forman una recta. La segunda se cumple aplicando un test de Shapiro a la gráfica de los residuos. La tercera condición es viendo la gráfica de los residuos, los cuales deben de estar variando en torno a la línea de mínimos cuadrados, y ésta debe ser constante (debe formar un cuadrado o rectángulos, los casos en que se formen conos, parábolas, formas raras, inmediatamente deja de tener una variabilidad constante en torno a la línea). Finalmente, las observaciones deben ser independientes entre sí, es decir, cada observación no tiene nada que ver y no influye en otra (esto se observa por contexto).

La función básica con la cual construiremos un modelo RLS, es con la función

modelo <- lm(fórmula, data), donde fórmula tendrá la forma “variable respuesta ~ variables predictoras”. Luego, para ver cómo predice el modelo, ésto se realiza con la función predict(modelo, new_data). Cabe mencionar que para la predicción, ésto tiene que funcionar sobre un conjunto nuevo de datos, ya que al utilizar los mismos datos con los cuales se construyó el modelo, éste podría estar “sobreajustado” (overfitting), con lo cual, su modelo queda totalmente invalidado para ser generalizable.

El sobreajuste lo definiremos como un aprendizaje de memoria, pero que, a la hora de la verdad, si le preguntamos acerca de algún conocimiento fuera de lo aprendido, no tendrá la autonomía de responder, donde muchas veces arrojará resultados erróneos de predicción.

En el caso de que una variable predictora sea categórica (en concreto, una variable discreta), ésta la tendremos que clasificar en torno a números, es decir, si la variable A toma los valores “b” y “c”, le tendremos que asignar los valores “0” y “1”, respectivamente (se pueden utilizar otros números, pero se recomienda que se realice desde 0 hasta n).

Para evaluar el poder predictivo de un modelo, tendremo que revisar ciertas situaciones, las cuales van de observar valores atípicos hasta comprobar que el modelo no esté sobreajustado. Para el caso de los valores atípicos se detallará más adelante. Para observar qué tan bien está un modelo de RLS, una buena opción es observar la Bondad de Ajuste, la cual podremos observar mediante el coeficiente de determinación, que no es más que el coeficiente de correlación de Pierson al cuadrado R^2, y ésta nos indicará el porcentaje de explicación de la variabilidad de los datos respecto a la variable predictora.

Una herramienta para poder evitar el sobreajuste, es dividir nuestro conjunto de datos, donde podremos establecer ciertos porcentajes de división. Por una parte, tendremos el conjunto de entrenamiento, el cual utilizaremos para la construcción del modelo, mientras que el conjunto de prueba lo utilizaremos para evaluar el poder predictivo de nuestro modelo. La lógica detrás de esto es que, como construimos el modelo con un conjunto de entrenamiento, cuando evaluemos el poder predictivo con el conjunto de prueba, este último conjunto serán datos totalmente desconocidos para cuando se construyó el modelo, por tanto, si su precisión logra un aprox del 100%, quiere decir que su modelo va bien encaminado. Finalmente, el porcentaje recomendado para el split del conjunto es 70% para entrenamiento y 30% para pruebas.

Una versión mejorada a lo anterior, es la validación cruzada de k-pliegues, la cual dividirá el conjunto de datos en k subconjuntos con igual tamaño. Ésto se puede realizar mediante la función modelo <- train(formula, method = “lm”, trControl = trainControl(method = “cv”, number)), y es la función que les recomiendo usar, ya que les construye el modelo a la vez de que realiza la evaluación del modelo (realiza la validación cruzada ó “Crossfield Validation”). Otra opción es realizar una validación cruzada dejando uno fuera, la cual crea una cantidad de subconjuntos iguales a la cantidad de observaciones, luego, para cada subconjunto, se va dejando una observación fuera para con las demás construir el modelo. Ésto se lleva a cabo con la misma función train, pero en el argumento trainControl, method debe llevar el valor “LOOCV”.

Regresión Lineal Múltiple (RLM):

Para la construcción de éste modelo, se realiza de la misma manera que para un modelo RLS, sólo que para fórmula, ésta debe ir de la forma “variable respuesta ~ variable predictora 1 + variable predictora 2 + ... + variable predictora K”, luego se observa que con “+” se agregan variables, mientras que con un “-“, podremos quitar variables.

En principio, las condiciones serán las mismas, pero al tener más variables involucradas, debemos tener cuidado con ciertas situaciones, por tanto, se agregan más condiciones, las cuales son:

1. Las variables predictoras deben ser cuantitativas o dicotómicas (de ahí la necesidad de variables indicadoras para manejar más de dos niveles).
2. La variable de respuesta debe ser cuantitativa y continua, sin restricciones para su variabilidad.
3. Los predictores deben tener algún grado de variabilidad (su varianza no debe ser igual a cero). En otras palabras, no pueden ser constantes.
4. No debe existir multicolinealidad. Esto significa que no deben existir relaciones lineales fuertes entre dos o más predictores (coeficientes de correlación altos).
5. Los residuos deben ser homocedásticos (con varianzas similares) para cada nivel de los predictores.
6. Los residuos deben seguir una distribución cercana a la normal centrada en cero.
7. Los valores de la variable de respuesta son independientes entre sí.
8. Cada predictor se relaciona linealmente con la variable de respuesta.

Las más importantes a mencionar son desde la condición 4 a la 6, las cuales se pueden obtener mediante funciones en R. Algo que mencionar es que NO debe existir multicolinealidad, ya que las variables predictoras no deben tener una fuerte relación lineal. Ésto se puede observar con una matriz de correlación.

Selección de Predictores:

Si bien podríamos utilizar todas las variables para poder predecir la de respuesta, PERO no todas aportan el mismo grado de información, por lo cual, tendremos que seleccionar las mejores, es decir, las que aporten mayor información. Una forma de realizar esto es mediante las métricas AIC y BIC, las cuales buscaremos SIEMPRE los menores valores de ambos.

A continuación, se presentan diferentes formas de obtener las mejores variables predictoras, pero tener en cuenta que ésto sólo funciona cuando queramos explorar datos, ya que, para estudios científicos, debemos de realizar profundos estudios previos para considerar las variables. Considerar que las 3 primeras utilizan AIC, mientras que la última utiliza BIC.

Los modelos nulos se entienden como modelos sin variables, mientras que los modelos completos contemplan todas las variables.

```
# Ajustar modelo nulo.
nulo <- lm(mpg ~ 1, data = datos)
cat("=== Modelo nulo ===\n")
print(summary(nulo))

# Ajustar modelo completo.
completo <- lm(mpg ~ ., data = datos)
cat("=== Modelo completo ===\n")
print(summary(completo))
```

1. Selección hacia adelante: Partimos desde un modelo nulo, al cual le iremos agregando variables hasta conseguir las mejores. En cada momento iremos midiendo el AIC.

   ```
   # Ajustar modelo con selección hacia adelante.
   adelante <- step(nulo, scope = list(upper = completo), direction = "forward",
                    trace = 0)
   ```

2. Eliminación hacia atrás: Partimos desde un modelo con todas las variables de estudio, al cual le iremos eliminando variables hasta conseguir las mejores.

   ```
   atras <- step(completo, scope = list(lower = nulo), direction = "backward",
                 trace = 0)
   ```

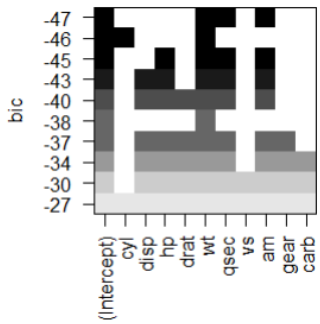
3. Regresión Escalonada: En palabras simples, es la combinación de los dos métodos anteriores.

   ```
   escalonado <- step(nulo, scope = list(lower = nulo, upper = completo),
                      direction = "both", trace = 0)
   ```

4. Método de Todos los Subconjuntos: Es la que más demora, pero la que puede entregar las mejores variables, ya que realiza una búsqueda exhaustiva de entre todos los posibles subconjuntos que se puedan generar.

   ```
   modelos <- regsubsets(mpg ~ ., data = datos, method = "exhaustive",
                         nbest = 1, nvmax = 10)

   print(plot(modelos))
   ```



En la anterior gráfica, las que tengan el cuadradito negro más arriba, serán nuestras variables predictoras. En este caso, “wt”, “qsec” y “am”.

La 4 es la más recomendada para su estudio, pero si les surge alguna complicación, usen las opciones 1 ó 2, ya que todas las selecciones son válidas.

Para la evaluación de un modelo RLM, tendremos que revisar valores atípicos y comprobar que se cumplan las condiciones, todo mientras se construya mediante alguna validación cruzada.

En primera instancia, si nos encontramos frente a valores atípicos, tendremos que corroborar si podremos quitar esos valores, ya que hay veces en las que los valores atípicos aportan mucha más información que los valores normales. Para ello tendremos ciertos estadísticos, las que se encuentran detalladas en las siguientes 8 temas:

1. Residuo estandarizado.
2. Valor predicho ajustado.
3. Residuo estudiantizado.
4. Diferencia en ajuste (DFFit).
5. Diferencia en betas (DFBeta).
6. Distancia de Cook.
7. Apalancamiento (Leverage).
8. Razón de Covarianzas.

Éstas se encuentran observadas en el siguiente script:

```r
resultados[["residuos_estandarizados"]] <- rstandard(modelo)
resultados[["residuos_estudiantizados"]] <-rstudent(modelo)
resultados[["distancia_Cook"]] <- cooks.distance(modelo)
resultados[["dfbeta"]] <- dfbeta(modelo)
resultados[["dffit"]] <- dffits(modelo)
resultados[["apalancamiento"]] <- hatvalues(modelo)
resultados[["covratio"]] <- covratio(modelo)

cat("Identificación de valores atípicos:\n")
# Observaciones con residuos estandarizados fuera del 95% esperado.
sospechosos1 <- which(abs(
  resultados[["residuos_estandarizados"]]) > 1.96)

cat("- Residuos estandarizados fuera del 95% esperado:",
    sospechosos1, "\n")

# Observaciones con distancia de Cook mayor a uno.
sospechosos2 <- which(resultados[["cooks.distance"]] > 1)

cat("- Residuos con una distancia de Cook alta:",
    sospechosos2, "\n")

# Observaciones con apalancamiento mayor igual al doble del
# apalancamiento promedio.

apal_medio <- (ncol(datos) + 1) / nrow(datos)
sospechosos3 <- which(resultados[["apalancamiento"]] > 2 * apal_medio)

cat("- Residuos con apalancamiento fuera de rango:",
    sospechosos3, "\n")

# Observaciones con DFBeta mayor o igual a 1.
sospechosos4 <- which(apply(resultados[["dfbeta"]] >= 1, 1, any))
names(sospechosos4) <- NULL

cat("- Residuos con DFBeta >= 1:",

    sospechosos4, "\n")

# Observaciones con razón de covarianza fuera de rango.
inferior <- 1 - 3 * apal_medio
superior <- 1 + 3 * apal_medio
sospechosos5 <- which(resultados[["covratio"]] < inferior |
                        resultados[["covratio"]] > superior)

cat("- Residuos con razón de covarianza fuera de rango:",
    sospechosos5, "\n")

# Resumen de valores sospechosos.
sospechosos <- c(sospechosos1, sospechosos2, sospechosos3,
                 sospechosos4, sospechosos5)

sospechosos <- sort(unique(sospechosos))

cat("\nResumen de valores sospechosos:\n")
cat("Apalancamiento promedio:", apal_medio, "\n")
```

```
cat("Intervalo razón de covarianza: [", inferior, "; ",
    superior, "]\n\n", sep = "")

print(round(resultados[sospechosos, c("distancia_Cook", "apalancamiento",
                                      "covratio")], 3))
```

Para todos los modelos es igual, por tanto es algo que pueden copypastear, teniendo en cuenta que deben cambiar los valores de “datos” y “modelo” con los que utilicen en su script.

Los resultados que arroja el script anterior es:

```
Identificación de valores atípicos:
- Residuos estandarizados fuera del 95% esperado: 17 18
- Residuos con una distancia de Cook alta:
- Residuos con apalancamiento fuera de rango: 9 16
- Residuos con DFBeta >= 1: 3 5 6 9 25 28 32
- Residuos con razón de covarianza fuera de rango: 15 16

Resumen de valores sospechosos:
Apalancamiento promedio: 0.125
Intervalo razón de covarianza: [0.625; 1.375]

                  distancia_Cook apalancamiento covratio
Datsun 710                 0.058          0.095     0.919
Hornet Sportabout          0.013          0.093     1.182
Valiant                    0.038          0.097     1.045
Merc 230                   0.162          0.297     1.313
Cadillac Fleetwood         0.008          0.227     1.474
Lincoln Continental        0.001          0.264     1.570
Chrysler Imperial          0.348          0.230     0.724
Fiat 128                   0.146          0.128     0.715
Pontiac Firebird           0.045          0.068     0.856
Lotus Europa               0.088          0.161     1.050
Volvo 142E                 0.063          0.124     1.016
```

Figura 14.16: identificación de valores atípicos.

Donde se eliminarán las observaciones (acá se indican con sus índices en el conjunto de datos) que aparezcan en todas las opciones de la parte de “Identificación de valores atípicos”, teniendo en cuenta que la que manda es la distancia de Cook. Una forma de concluir para este caso sería “Como se presentan que existen ciertas observaciones como valores atípicos, ninguna de aquellas posee una distancia de Cook mayor a 1, por lo cual no son razón para eliminar”.

Luego, para verificar las condiciones, se debe de realizar lo siguiente:

1. Independencia de los Residuos: Se utiliza la función durbinWatsonTest(modelo), la cual entregará un p-valor, donde H0 será que los residuos son independientes, Ha no son independientes.
2. Distribución normal de los residuos: Se utiliza una prueba de Shapiro sobre los residuos.

3. Homocedasticidad de los residuos: Se utiliza la función ncvTest(modelo), obteniendo un p-valor, donde H0 se cumple el supuesto de homocedasticidad, Ha no se cumple.
4. Multicolinealidad: Ésto se observa con el valor del VIF de cada variable, donde no puede sobrepasar 10, 5 ó 2.5 (se recomienda el intermedio, 5). Además, si el VIF promedio es mayor a 1, podría haber sesgo en el modelo. Otra forma de ver la multicolinealidad es la Tolerancia, que no es más que el inverso multiplicativo del VIF. Todo ésto se puede observar mediante la función vif(modelo), obteniendo lo siguiente:

```
Verificar la multicolinealidad:
- VIFs:
      wt     qsec       am
2.482952 1.364339 2.541437
- Tolerancias:
      wt      qsec        am
0.4027465 0.7329556 0.3934781
- VIF medio: 2.129576
```

Donde se observa que los vifs son menores a 5, por tanto se puede concluir que no existe multicolinealidad entre las presentes variables, pero como el vif medio es mayor a 1, existe el riesgo de sesgo en el modelo.

Finalmente, la validación cruzada para una RLM será de la misma forma que para una RLS.

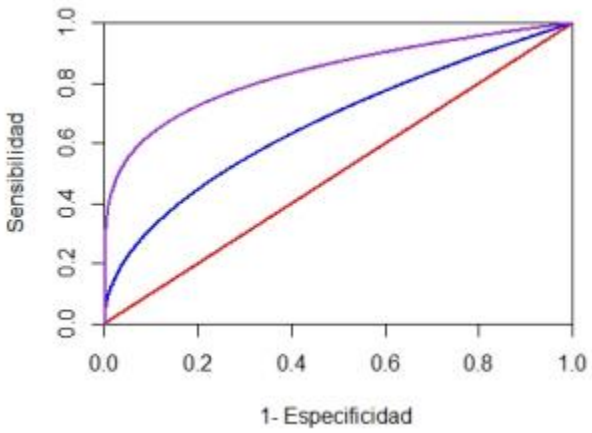
Regresión Logística:

RLM y RLS nos permiten predecir una variable numérica continua, la cual sigue una distribución normal. Pero no todas las variables son de las mismas características. El mayor problema surge cuando queramos predecir una variable que no sigue una distribución normal, como pueden ser las variables discretas, las que toman valores puntuales. Un ejemplo de ésto son las variables dicotómicas (toman sólo 2 valores), ya que no podremos utilizar RLS ó RLM para éste propósito. Nuestra solución para este problema es utilizar una Regresión Logística. La función base a utilizar para construir un modelo de RLog es glm(formula, family = binomial(link = “logit”), data).
Para evaluar un modelo de regresión logística se realizará mediante una matriz de confusión, en el que, para una variable dicotómica, la matriz será de la siguiente manera:

		Real		
		1 (+)	**0 (−)**	**Total**
Clasificación	**1 (+)**	VP	FP	$VP + FP$
	0 (−)	FN	VN	$FN + VN$
	Total	$VP + FN$	$FP + VN$	n

Se construye mediante la función confusionMatrix(data, reference), donde data es la respuesta predicha y reference la observada.
Aquí, VP son los verdaderos positivos, es decir, correctamente clasificados como 1 (positivo), VN verdaderos negativos (correctamente clasificados como 0), FP son los erróneamente clasificados como positivos cuando deberían ser negativos y FN son los erróneamente clasificados como negativos cuando deberían ser positivos. Luego, en el libro, aparecen ciertas métricas en base a lo

anterior, pero las más importantes serían la exactitud ([VP + VN] / n), la sensibilidad (VP / [VP + FN]) y la especificidad (VN / [FP + VN]). Estos valores se pueden graficar mediante una curva de calibración, llamada curva ROC, que será de la forma:



Mientras más lejos se encuentre nuestra curva respecto de la diagonal roja, mayor exactitud tendrá nuestro modelo.

Para utilizar este tipo de regresiones, debemos verificar ciertas condiciones:

1. Debe existir una relación lineal entre los predictores y la respuesta transformada.
2. Los residuos deben ser independientes entre sí.

Aparte de las condiciones anteriores, pueden ocurrir ciertas situaciones donde el método de optimización no converja:

1. Multicolinealidad entre los predictores, que en este caso se aborda del mismo modo que para RLM (por ejemplo, mediante el VIF o la tolerancia).
2. Información incompleta, que se produce cuando no contamos con observaciones suficientes para todas las posibles combinaciones de predictores.
3. Separación perfecta, que ocurre cuando no hay superposición entre las clases, es decir, ¡cuando los predictores separan ambas clases completamente!. Para ésto, R muestra un warning:

```
Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

En el caso en que usemos una regresión logística múltiple, debemos de buscar las mejores variables predictoras, para lo cual se procede de la misma manera que para RLM, teniendo en cuenta que los modelos nulo y completo se construyen con glm y no con lm.

Finalmente, el proceso para observar los valores atípicos, verificar independencia de residuos y multicolinealidad, se realizan de la misma manera que lo explicado en RLM, sólo cambian las funciones para construir el modelo de regresión logística.

Se dan a conocer dos tipos para construir modelos, además del glm:

Mediante train y RFEcontrol:

```
modelo <- train(am ~ wt, data = entrenamiento, method = "glm",
                family = binomial(link = "logit"),
                trControl = trainControl(method = "cv", number = 5,
                                         savePredictions = TRUE))
```

En este caso, no considerar la variable modelo como el modelo construido por train, éste se encuentra utilizando lo siguiente: modelo[[“finalModel”]], en el que modelo es lo que arroja el train.

Por otro lado, RFEControl se realizará de la siguiente manera (ejemplo sacado del ejemplo solución del EP 15):

```
control.seleccion <- rfeControl(functions = lrFuncs, method = "LOOCV",
                                number = 1, verbose = FALSE)

control.entrenamiento <- trainControl(method = "none", classProbs = TRUE,
                                      summaryFunction = twoClassSummary)

modelo.en <- rfe(datos.en, EN, metric = "ROC", rfeControl = control.seleccion,
                 trControl = control.entrenamiento, sizes = 2:6)

print(modelo.en)
```

Una cosa importante del RFE es que realiza internamente el proceso de seleccionar las variables, donde se definen las cantidas en la función rfe, en el parámetro sizes, donde se establecen los límites.