INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

Algoritmos y análisis de datos

Evidencia 2

Team 1

Andrés G. Treviño Leal A01177504

Julius Döbelt A01760777

Profesor

Doc. Rodolfo Gameros

Fecha de entrega

4 de Septiembre del 2022

En este documento se estarán analizando modelos que más se adaptan a la situación de Fojal que en este caso es la otorgación de créditos para sus clientes, nos han pedido crear un modelo que les pueda ayudar a determinar a quiénes de sus clientes se les debería de otorgar el crédito y a quienes no. En base a la información recolectada de los clientes que sería Capital de Trabajo/ Ingresos Retenidos/ Ingresos antes de intereses e impuestos/ Ventas/ Divididas entre Activos por último el valor de mercado.

```{r}
fondo[!complete.cases(fondo),]

```

El principio de cualquier base de datos es analizar o checar de cualquier incoherencia datos, esto se puede ver como un NA. En nuestro caso no se obtuvieron NA para esto utilizamos la función de “complete.cases”.

```{r}
fondo$aprobado <- ifelse(fondo$Default==0, 1, 0)
fondo$no_aprobado <- ifelse(fondo$Default==1, 1, 0)
mean(fondo$aprobado)
```

Lo que se nos pidió hacer es crear dos nuevas columnas(dummy variables), para esto utilizamos la función de “ifelse" donde de la columna default, se iba a obtener la columna aprobado o no aprobado en este caso del crédito.

En los resultados de la función “ifelse” pudimos notar que el 98.2% de los créditos fueron aprobados. Lo que podemos inferir de la estadística descriptiva es que la empresa de Fojal ha otorgado más créditos a sus clientes que negárselo.

```{r}
fondo <- select(fondo, ID, Year, Default, `WC/TA`, `RE/TA`,
`EBIT/TA`, `ME/TL`, `S/TA`)
```

En la función de “select” la utilizamos para obtener las variables que iban a estar más relacionadas con el modelo que queremos crear, no pusimos las dummies ya que no tenían mucha correlación sólo las hicimos para identificar cuántos créditos se otorgaban.

```r
set.seed(11)
train = createDataPartition(y = fondo$Default, p=0.8, list=FALSE,
times = 1)
datos_train = fondo[train,]
datos_test = fondo[-train,]
```

En esta parte decidimos dividir nuestra base de datos en dos, donde "train" se quedó con 80% de la base de datos y "test" con el otro 20%. Esto lo hicimos con la finalidad de entrenar nuestro modelo teórico primero con el 80% de los datos y el 20% lo testeamos para averiguar si el modelo estaba correcto.

En las próximas páginas construimos tres modelos para predecir los resultados: el modelo Forward, el modelo Backward y el modelo Lasso. De los modelos Forward y Backward elegimos ese que tiene un valor de AIC más pequeño y entrenamos lo con la base de datos Train y predecimos los resultados de la base de datos Test y comparamos los con los observaciones de la base de datos real. Adicionalmente, comparamos los resultados con los del modelo Lasso y decidimos que modelo es mejor para predecir las probabilidades de incumplimiento de los clientes.

Como fundación para nuestros modelos de predicción usamos el modelo "probit". Usamos el modelo "probit" porque en la Evidencia 1 tuvo mejores resultados en comparación con el modelo "logit".

```r
modelo_probit = glm(Default~., data=fondo, family = binomial(link="probit"))
```

Forward Model:

```
Call:
glm(formula = Default ~ ID + Year + `WC/TA` + `RE/TA` + `EBIT/TA` +
    `ME/TL` + `S/TA`, family = binomial(link = "probit"), data = fondo)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7182  -0.1690  -0.0661  -0.0188   4.4288

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.052e+02  4.578e+01   2.299 0.021533 *
ID           2.714e-03  3.847e-04   7.054 1.73e-12 ***
Year        -5.407e-02  2.289e-02  -2.362 0.018156 *
`WC/TA`      2.143e-01  2.950e-01   0.727 0.467498
`RE/TA`     -8.568e-01  1.278e-01  -6.702 2.06e-11 ***
`EBIT/TA`   -5.458e+00  1.446e+00  -3.774 0.000161 ***
`ME/TL`     -3.699e-01  1.022e-01  -3.620 0.000295 ***
`S/TA`       3.087e-01  1.976e-01   1.562 0.118317
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 721.20  on 3999  degrees of freedom
Residual deviance: 471.91  on 3992  degrees of freedom
AIC: 487.91

Number of Fisher Scoring iterations: 12
```

El modelo “Forward” tiene un valor de AIC de 487.91 y las variables más importantes para la predicción del resultado son ID, Year, WC/TA, RE/TA, EBIT/TA, ME/TL y S/TA. En otras palabras, el modelo incluye todas las variables para su predicción. Normalmente, modelos no requieren todas las variables para las predicciones porque algunas variables no tienen una influencia en el resultado final (se llamaban “noise”). Es un primer indicador que el modelo “forward” no es óptimo.

Backward Model:

```
Call:
glm(formula = Default ~ ID + Year + `RE/TA` + `EBIT/TA` + `ME/TL` +
    `S/TA`, family = binomial(link = "probit"), data = fondo)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6994  -0.1721  -0.0676  -0.0192   4.4429

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.078e+02  4.574e+01   2.357 0.018432 *
ID           2.696e-03  3.829e-04   7.042  1.9e-12 ***
Year        -5.535e-02  2.287e-02  -2.420 0.015509 *
`RE/TA`     -8.355e-01  1.255e-01  -6.657  2.8e-11 ***
`EBIT/TA`   -5.402e+00  1.450e+00  -3.725 0.000196 ***
`ME/TL`     -3.604e-01  1.015e-01  -3.549 0.000386 ***
`S/TA`       3.245e-01  1.953e-01   1.662 0.096501 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 721.20  on 3999  degrees of freedom
Residual deviance: 472.43  on 3993  degrees of freedom
AIC: 486.43

Number of Fisher Scoring iterations: 12
```

En el modelo “Backward”, las variables más importantes para la asignación de crédito son ID, Year, RE/TA, EBIT/TA, ME/TL y S/TA. El modelo “Backward” no incluye la variable `WC/TA` porque no tiene una influencia en el resultado final. El valor de AIC es 486.43 que significa que el modelo "Backward" es mejor (AIC Backward = 486.43 < 487.91 = AIC Forward). Por eso solo entrenamos el modelo “Backward” porque ese modelo hará mejores predicciones.

Entrenamos nuestro modelo “Backward” con los datos de Training que representa 80% de la base de datos. Con una “Accuracy” de 98,61% es muy bueno. Pero sabemos que la mayoría de los clientes pueden pagar el crédito y lo más interesante es que cuántos de los clientes que no pueden pagar el crédito encontramos con el modelo. La matriz de confusión ayúdanos con esa pregunta (Explicaciones debajo)..

```
datos_train$Default <- as.factor(datos_train$Default)

modelo_crédito<-train(Default ~ ID + `RE/TA` + `EBIT/TA` + `ME/TL` + `S/TA`, data=datos_train, method="glm",
family=binomial(link="logit"))
modelo_crédito
modelo_crédito$finalModel
```

```
Generalized Linear Model

3200 samples
   5 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 3200, 3200, 3200, 3200, 3200, 3200, ...
Resampling results:

  Accuracy  Kappa
  0.986083  0.4568863


Call:  NULL

Coefficients:
    (Intercept)              ID    `\\`RE/TA\\``  `\\`EBIT/TA\\``    `\\`ME/TL\\``      `\\`S/TA\\``
      -6.086912         0.007596        -1.285149        -8.615108        -1.799045         0.525633

Degrees of Freedom: 3199 Total (i.e. Null);  3194 Residual
Null Deviance:      604
Residual Deviance: 358.4        AIC: 370.4
```

Matriz de confusión para el modelo “Backward”:

```
predicciones1 <- ifelse(pred_credit$`1` > 0.22, yes = 1, no = 0)
matriz_confusion <- table(predicciones1,datos_test$Default,dnn = c( "predicciones","observaciones"))
matriz_confusion
```

```
            observaciones
predicciones   0   1
           0 786   7
           1   3   4
```

Para un valor de Cutoff de 0.22 tenemos los mejores resultados.

```
Recall <- matriz_confusion[2,2]/(matriz_confusion[2,2]+matriz_confusion[1,2])
Recall
Precision <- matriz_confusion[2,2]/(matriz_confusion[2,2]+matriz_confusion[2,1])
Precision
```

```
[1] 0.3636364
[1] 0.5714286
```

Para evaluar nuestro modelo creamos la matriz de confusión y calculamos los valores para Recall y Precision. Los valores para las dos métricas son buenos si notas solo existen 3200 puntos de datos para entrenar el modelo y por eso los valores no son perfectos. La mayoría de las personas pueden pagar el crédito y el modelo predice lo. Un Recall de 0.36 y Precision de 0.57 no son espectaculares pero el modelo de predicción definitivamente ayuda con la decisión a que clientes no debemos dar un crédito. Un Recall de 57% significa que podemos

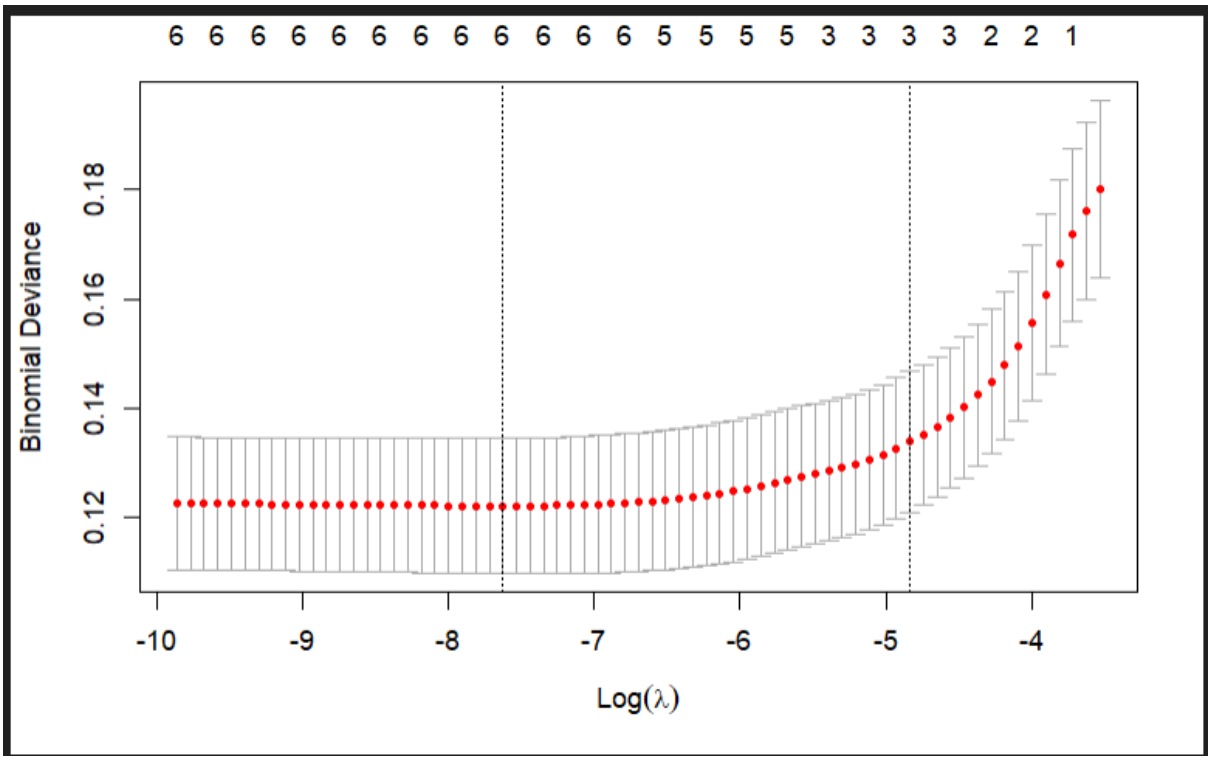
predecir 57 de 100 clientes que no pueden pagar el crédito. Precision de 36% significa que en 36 de 100 casos cuando nuestro modelo predice que los clientes no van a pegar el crédito, los clientes no pueden pagar el crédito en serio. Con más datos nuestro modelo sería mejor.

Modelo Lasso

El modelo Lasso es otro modelo para predecir los resultados. Para eso, necesitamos la librería "glmnet" en que el "glm" significa regresión logística.

```
library("glmnet")
x = model.matrix(Default ~ ., fondo)[,-3]
y = fondo$Default
```

En el código de arriba borramos la tercera columna porque es la variable de Default.



Para el modelo Lasso es de mayor importancia de que hay variables estandarizadas porque las variables tienen que ubicar una misma escala. Por eso declaramos “standardize = T”. Con esa configuración calculamos nuestro modelo Lasso best_lasso lo que use las variables ID, `WC/TA`, `RE/TA`, `EBIT/TA`, `ME/TL`, `S/TA`. El modelo no necesita la variable ‘Year’ para sus predicciones

Matriz de confusión:

Para el modelo Lasso se requiere un valor de Cutoff alto porque las probabilidades son casi 1 para la mayoría de las predicciones.

```
predicciones2 <- ifelse(predLprobs > 0.99995, yes = 1, no = 0)
matriz_confusion2 <- table(predicciones2, datos_test$Default)
matriz_confusion2
```

```
predicciones2   0   1
            0 738   4
            1  51   7
```

```
ze = T)
```

```
8 x 1 sparse Matrix of class "dgCMatrix"
                     s0
(Intercept) -6.644548519
(Intercept)  .
ID           0.007524463
`WC/TA`      0.356301815
`RE/TA`     -1.615346174
`EBIT/TA`   -9.182746172
`ME/TL`     -0.788710365
`S/TA`       0.456015103
```

Con esa configuración recibimos una gran Recall de 63,64% pero solo un valor de Precision de 12,07%. Un valor más grande para Precision es casi imposible porque si aumentamos Precision un poco, Recall baja a 0-20%.

```
Recall <- matriz_confusion2[2,2]/(matriz_confusion2[2,2]+matriz_confusion2[1,2])
Recall
Precision <- matriz_confusion2[2,2]/(matriz_confusion2[2,2]+matriz_confusion2[2,1])
Precision
```

```
[1] 0.6363636
[1] 0.1206897
```

Resumen

Primero, construimos los modelos “Forward” y “Backward” que tenían resultados similares. Sin embargo, el valor de AIC fue más pequeño para el modelo “Backward” que es la razón porque usamos ese modelo para entrenar nuestro modelo de predicción. Entonces, comparamos los resultados de nuestro modelo de predicción con los valores reales de la base de datos Test y evaluamos el modelo con la matriz de confusión y métricas como Recall y Precision.

Segundo, creamos el modelo Lasso para predecir los resultados. Ese modelo también analizamos con la matriz de confusión y las métricas Recall y Precision.

En nuestro caso, el modelo “Backward” tiene las mejores predicciones porque tenemos altos valores para Precision y Recall. En consecuencia, con el modelo “Backward” podemos predecir precisamente los clientes que no pueden pagar el crédito. El modelo no es perfecto porque hay solo 3200 datos para entrenar el modelo que es casi nada. Si tuviéramos una base de datos con más de un millón de filas, el modelo sería mejor.