

Modelo para la predicción de la intención de compra de compradores online

Juárez Jacobo León¹, Gallego Franco A.²

Departamento de Informática

Facultad de Ciencias Exactas

Universidad Nacional de Salta

Salta, Argentina

¹leonjuarez777@gmail.com, ²francoalex947@gmail.com

Resumen

Abordaremos técnicas de minería de datos para saber si una persona comprará o no en un sitio web de compras, basado en el comportamiento del mismo a través del sitio. Esto tendría un gran impacto económico para las empresas al darle información a cerca de sus clientes y como pueden mejorar sus ventas. En nuestro estudio logramos alcanzar una precisión de hasta el 90%.

Palabras clave: Predicción, Regresión Multinomial, Árbol de decisión, Compras Online.

I. INTRODUCCIÓN

La creciente demanda por las organizaciones de poder manejar grandes volúmenes de información hace posible que puedan impactar más en el mercado, ya que hoy en día, la información es un bien económico que tiene que tratarse como tal. Gracias a la ciencia de datos, muchas de las grandes organizaciones pueden adelantarse a sus competidores y atraer hacia ellos más clientes potenciales, por lo que hacer un buen análisis de la información puede hacer la diferencia entre que una empresa gane dinero o vaya a la quiebra.

En el presente informe desarrollamos el estudio basado en predicciones sobre los datos obtenidos a través de **Google Analytics**¹. En base a las observaciones realizadas, contamos con los datos de diferentes usuarios ubicados en distintas partes del mundo que navegan a través de Internet y sus acciones realizadas como por ejemplo: la cantidad de páginas web visitadas, las compras online realizadas, entre las cuestiones más importantes.

II. DATASET: ONLINE SHOPPERS

El dataset **Online Shoppers** consta de 12.330 sesiones y el comportamiento del usuario en las mismas. Los datos fueron tomados a lo largo del período de un año. El mismo fue realizado en base a los datos obtenido por **Google Analytics**.

A. Descripción del dataset

El dataset se compone de 18 variables (10 variables numéricas, y 8 variables categóricas).

1) **Variables numéricas:** Las variables numéricas "*Administrative*", "*Administrative Duration*", "*Informational*", "*Informational Duration*", "*Product Related*" y "*Product Related Duration*" representan el número de las diferentes páginas visitadas por usuarios en esa sesión y el tiempo total de sesión.

Las categorías de páginas se obtienen a partir de la información de **URL** de las páginas visitadas por el usuario y se actualizan en tiempo real cuando un usuario realiza una acción, por ejemplo, ir de una página a otra (gracias al registro de cookies que tiene los diferentes tipos de navegadores web).

Las variables "*Bounce Rate*", "*Exit Rate*" y "*Page Value*" representan las métricas medidas por Google Analytics para cada página de los sitios de comercio electrónico. En particular, *Bounce Rate* se refiere al porcentaje de usuarios que entran a un determinado sitio y luego navegan hacia otro sitio sin realizar ninguna acción que active la recolección de datos por parte de Google.

¹**Google Analytics** es una herramienta de analítica web de la empresa Google. Ofrece información agrupada del tráfico que llega a los sitios web según la audiencia, la adquisición, el comportamiento y las conversiones que se llevan a cabo en el sitio web.

2) **Variables categóricas:** La variable "*Special Day*" indica la proximidad de tiempo de visita del sitio a un día particular (por ejemplo, Día de la madre, Día de San Valentín, etc) en el que más probable que las sesiones finalicen con la transacción (la compra por comercio electrónico). El valor de esta variable se determina considerando la dinámica del comercio electrónico, como la duración entre la fecha del pedido y la fecha de entrega. Por ejemplo, para el día de San Valentín, este valor toma un valor distinto de cero entre el 2 de febrero y el 12 de febrero, cero antes y después de esta fecha a menos que esté cerca de otro día especial, y su valor máximo es del 1 al 8 de febrero.

Tabla I: Variables del dataset - Online Shoppers

	Característica	Descripción
1	Administrative	Numero de paginas visitadas relacionadas con administración
2	Administrative Duration	Tiempo total en segundos que el usuario se mantuvo en paginas relacionadas con administración
3	Informational	Numero de paginas visitadas con relación a información de la pagina web, comunicación, dirección, etc
4	Informational Duration	Tiempo total en segundos que el usuario se mantuvo en paginas relacionadas con información
5	ProductRelated	Numero de paginas visitadas relacionadas a productos
6	ProductRelated Duration	Tiempo total en segundos que el usuario se mantuvo en paginas relacionadas a productos
7	BounceRates	Valor promedio de la tasa de rebote de las páginas visitadas
8	ExitRates	Valor promedio de la tasa de salida de las páginas
9	PageValues	Valor promedio de página de las páginas visitadas
10	SpecialDay	Cercanía a un día especial
11	Month	Mes del año
12	OperatingSystems	Sistema operativo del usuario
13	Browser	Navegador web del usuario
14	Region	Región geográfica del usuario
15	TrafficType	Indica como llego el usuario al sitio web (por ejemplo banner, SMS, direct)
16	VisitorType	El tipo de visitante puede ser "New Visitor", "Returning Visitor" and "Other"
17	Weekend	Indica si es fin de semana
18	Revenue	Indica si se realizó una compra

III. ENTENDIENDO LOS DATOS

Antes de realizar nuestro estudio sobre el dataset es muy importante entender el comportamiento de los datos, es decir, necesitamos hacer un análisis a priori para saber a que nos enfrentamos y poder sacar conclusiones rápidamente antes de aplicar todas las técnicas necesarias para los modelos de predicción. A continuación presentamos el análisis univariado y multivariado de los datos.

A. Análisis Univariado de datos

Primeramente realizamos boxplot de todas las variables numéricas del data set para ver si existen algunas anomalías como por ejemplo (*outliers*, explicado más adelante en secciones posteriores). Para ello realizamos un boxplot de todas las variables originales del dataset (18 características). En R se utiliza el siguiente comando:

```
boxplot(online_shoppers)
```

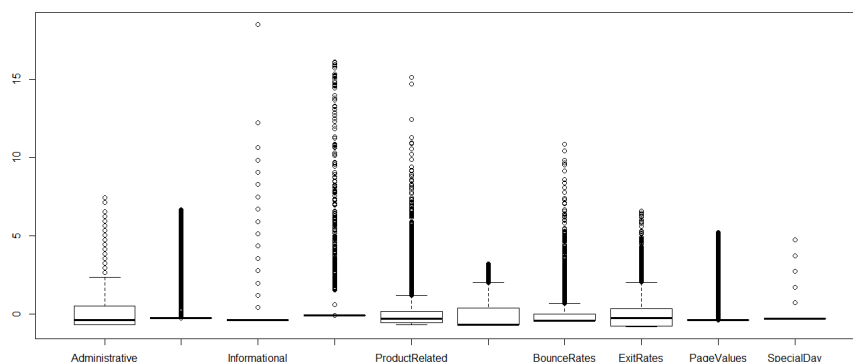


Fig. 1: Boxes plots de las 10 variables numéricas

Con la primera aproximación de los *box plot* podemos ver claramente que este dataset tiene muchos valores atípicos(*outliers*).

Luego analizamos mediante el siguiente gráfico, la relación que existe entre aquellos usuarios que entran a un sitio web para comprar y el tiempo necesario que el usuario tarda en llenar los formularios correspondientes (por ejemplo, para validar una venta). En *R* este gráfico se obtiene de la siguiente manera:

```
library(gridExtra)
library(ggplot2)

plot1 = ggplot(data = online_shoppers) +
  geom_point(mapping = aes(x = c(1:12330), y = Informational, color = Revenue))

plot3 = ggplot(data = online_shoppers) +
  geom_point(mapping = aes(x = c(1:12330), y = BounceRates, color = Revenue))
```

El resultado obtenido es el siguiente:

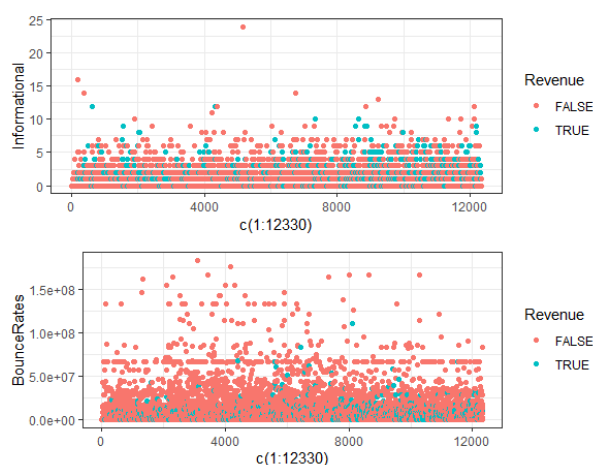


Fig. 2: Correspondencias entre las variables

B. Análisis multivariado de datos

Ahora procedemos a realizar el análisis multivariado de datos, para poder comprender si existen correlaciones entre las variables, y como influye una variable en otra. Para obtener una primera impresión, realizamos un gráfico de la matriz de correlación de las variables numéricas.

Los comandos necesarios en *R* son:

```
library(GGally)
ggcorr(online_shoppers_NM, method = c("everything", "pearson"))
```

Vemos que existen mayoritariamente correlaciones positivas entre las variables. Algunas mas fuertes que otras.

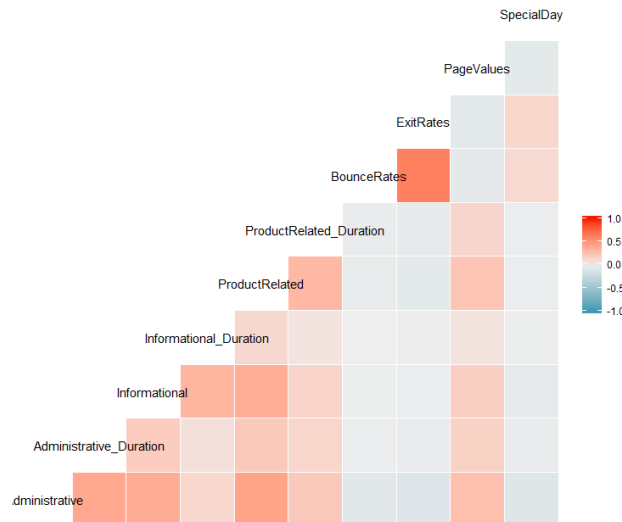


Fig. 3: Plot de la matriz de correlación de las variables numéricas

C. Análisis de componentes principales

Luego procedemos a realizar un análisis de las componentes principales del dataset. Esta técnica se utiliza para describir un conjunto de datos en términos de nuevas variables (*componentes*) no correlacionadas. Los componentes se ordenan por la cantidad de varianza original que describen, por lo que la técnica es útil para reducir la dimensionalidad de un conjunto de datos.

Para aplicar este método, es necesario analizar la dependencia entre las variables. Para ello utilizamos el *Test de Bartlett*.

El *Test de Bartlett* permite contrastar si más de 2 muestras presentan igualdad de varianzas (homocedasticidad). Las hipótesis que se plantean son:

- H_0 : las muestras presentan varianzas iguales (es decir que las variables son independientes).
- H_1 : las muestras presentan varianzas distintas (es decir que las variables son dependientes entre sí).

Para aceptar la hipótesis nula el p – valor tiene que ser menor 0.05

Para aplicar el test, recurrimos a los siguientes comandos en *R*:

```
cortest.bartlett(cor(online_shoppers_NM), nrow(online_shoppers_NM))
```

En este caso el p – valor es igual a cero, luego, rechazamos la hipótesis nula (hay variables dependientes) y por lo tanto el conjunto de variables del dataset se puede reducir.

Procedemos a calcular las componentes principales y posteriormente analizamos las contribuciones de cada componente según corresponda:

```

PCAOnlineShoppers = prcomp(online_shoppers_NM, scale. = TRUE )
summary(PCAOnlineShoppers)

#Contribuciones
PCAcont = PCAOnlineShoppers$rotation %*% diag(PCAOnlineShoppers$sdev)

```

Las contribuciones nos indican que tomando 6 componentes principales la contribución será del 0.78543. Luego podemos analizar las 6 componentes y sus coordenadas de los individuos, relacionados con la variable que explica si un visitante de una página web comprará o no.

IV. PREPROCESAMIENTO

Los datos del mundo real muchas veces contiene puntos de datos incompletos e inconsistentes. También pueden carecer de ciertos comportamientos o tendencias, y es probable que contenga muchos errores. Convertiremos estos datos en un formato que permita utilizar las distintas técnicas de minería de datos, lo que se puede entender como procesamiento previo. Todos los científicos de datos dedican la mayor parte de su tiempo a las operaciones de preprocesamiento.

A. Valores faltantes

Pueden haber datos faltantes en el junto de datos, que si no se manejan adecuadamente pueden generar errores más adelante o pueden dar lugar a inferencias inexactas. Primero, descubrimos si faltan valores. En la Tabla 2 podemos ver las características y a su lado la cantidad de valores ausentes que tienen.

Hay dos formas de manejar los valores faltantes. La primera es eliminar la fila con los datos faltantes, la segunda es completar los valores faltantes con la media, la mediana, la moda o el valor que aparece con más frecuencia en la columna correspondiente si se trata de una variable categórica. Como no tenemos datos faltantes, no tomamos acción.

```

sapply(onlineShoppers, function(x) sum(is.na(x)))

```

Tabla II: Datos Faltantes

Característica	Datos Faltantes
Administrative	0
Administrative Duration	0
Informational	0
Informational Duration	0
ProductRelated	0
ProductRelated Duration	0
BounceRates	0
ExitRates	0
PageValues	0
SpecialDay	0
Month	0
OperatingSystems	0
Browser	0
Region	0
TrafficType	0
VisitorType	0
Weekend	0
Revenue	0

B. Outliers

¿Por qué es importante identificar los valores extremos(outliers)?

Es necesario identificarlos ya que puede sesgar(bias) o cambiar drásticamente las estimaciones y predicciones que se realizan sobre el modelo que en secciones posteriores será expuesto.

1) **Aspectos importantes:** Concluir que una observación es un outlier basado sólo en una característica podría llevar a que el modelo de predicción realice inferencias erróneas.

Cuando se tiene que decidir si una entidad individual (representada por una fila u observación) del dataset, es un valor extremo o no, es mejor considerar en conjunto a las características \mathbf{X} .

Cook Distance Esta distancia es una medida calculada con respecto a un modelo de regresión dado y, por lo tanto, sólo se ve afectada por las variables X incluidas en el modelo. Mide la influencia ejercida por cada punto de datos (fila) el resultado predicho.

La *Cook distance* para cada observación i mide los cambios en \hat{Y} (ajuste de Y) para todas las observaciones incluyendo aquellas que no presentan el valor i . Gracias a esto podemos saber cuánto impactó la observación en los valores ajustados.

Matemáticamente la *Cook distance* se calcula como:

$$D_i = \frac{\sum_{j=1}^n \left(\hat{Y}_j - \hat{Y}_{j(i)} \right)^2}{p \times MSE} \quad (1)$$

donde,

- \hat{Y}_j es el valor de j_{th} ajustado de respuesta cuando todas las observaciones son incluidas.
- $\hat{Y}_{j(i)}$ es el valor de j_{th} ajustado de respuesta, donde el ajuste no incluye la observación i .
- MSE es el error cuadrático medio.
- p es el número de coeficientes en el modelo de regresión.

Esto en R se traduce de la siguiente manera:

```
cooks_d <- cooks.distance(model_regression)
```

En general, aquellas observaciones que tienen una *Cook distance* mayor a 4 veces la media, pueden clasificarse como influyentes.

2) **Tratamiento de outliers:** Una vez que los valores extremos fueron identificados y verificamos la naturaleza del problema, tenemos que tener en cuenta los siguientes enfoques:

- 1) **Imputación de valores:** puede elegir reemplazar los valores extremos por algún valor conocido (media, mediana y la moda).
- 2) **Limitar valores extremos:** para valores extremos que se encuentre fuera los límites del rango intercuartílico, podríamos limitarlo reemplazando esas observaciones fuera del límite inferior con el valor del 5% y las que se encuentran por encima del límite superior, con el valor del 95%. A continuación se muestra el código en R :

```
qnt=quantile(x, probs=c(.25, .75), na.rm=T)
caps=quantile(x, probs=c(.05, .95), na.rm=T)
H <- 1.5 * IQR(x, na.rm = T)
x[x < (qnt[1] - H)] <- caps[1]
x[x > (qnt[2] + H)] <- caps[2]
```

- 3) **Predicción:** dependiendo el contexto, los valores atípicos se pueden reemplazar con valores faltantes (NA) y luego se pueden predecir al considerarlos como una variable de respuesta.

C. Manejo de datos categóricos

Como en los modelos que vamos a implementar las operaciones matemáticas, está claro que no podemos dar entradas como Meses; 'Enero', 'Febrero', etc. para el modelo. La forma más fácil de manejar este tipo de datos es la codificación de etiquetas, donde cada categoría en un atributo particular está codificada por un número único; Enero = 0, febrero = 1, etc.

Si bien los modelos con este tipo de codificación arrojan resultados aceptables, el modelo podría estar sesgado hacia algunas categorías que se han codificado con un valor alto. (Por ejemplo, diciembre = 11 y enero = 0). Para

evitar este efecto, utilizamos la codificación Onehot para nuestro conjunto de datos. Después de la codificación, las 18 características iniciales de entrada aumentaron a 57. Este paso es muy importante para permitir un entrenamiento más rápido y evitar complicaciones innecesarias del modelo.

```
library(dummies)
library(data.table)

shopMod = dummy.data.frame(online_shoppers, names = c("Month", "Region",
  "VisitorType", "OperatingSystems", "Browser", "Weekend"), sep = "_")
View(shopMod)
```

BounceRates	ExitRates	PageValues	SpecialDay	Month_Aug	Month_Dec	Month_Feb
0.20	0.20	0	0.0	0	0	1
0.00	0.10	0	0.0	0	0	1
0.20	0.20	0	0.0	0	0	1
0.05	0.14	0	0.0	0	0	1
0.02	0.05	0	0.0	0	0	1
15789474.00	24561404.00	0	0.0	0	0	1
0.20	0.20	0	0.4	0	0	1
0.20	0.20	0	0.0	0	0	1

Fig. 4: Resultados de la codificación one hot

V. SELECCIÓN DE LAS MEJORES CARACTERÍSTICAS

Al tener una gran cantidad de variables (57) se hace necesario seleccionar aquellas que tengan una mayor efecto sobre la variable "Revenue" y eliminar las que no tienen un efecto considerable sobre ella.

Los métodos de selección automática de características se pueden usar para construir muchos modelos con diferentes subconjuntos de un conjunto de datos e identificar aquellos atributos que son y no son necesarios para construir un modelo preciso.

Un método automático popular para la selección de funciones proporcionado por el paquete caret de R se llama Recursive Feature Elimination o RFE. Un algoritmo de bosque aleatorio es usado en cada iteración para evaluar el modelo. El algoritmo está configurado para explorar todos los subconjuntos posibles de los atributos. Se seleccionaron 13 atributos, aunque en el gráfico muestra la precisión de los diferentes tamaños de subconjunto de atributos, podemos ver que solo 13 atributos dan resultados casi comparables a los que daría un modelo con todas los atributos.

```
# Librerias
library(mlbench)
library(caret)

# Para que sea repetible
set.seed(7)
control=rfeControl(functions=rfeFuncs, method="cv", number=10)

# Corremos el algoritmo RFE
testing=rfe(shopMod[,1:56], shopMod$Revenue, sizes=c(1:56), rfeControl=control)

# Lista de las caracteristicas elegidas
predictors(testing)

# Grafico de los resultados
plot(testing, type=c("g", "o"))
```

En base al análisis anterior, seleccionamos las mejores 13 características. Estas son Administrative, Administrative_Duration, Informational, Informational_Duration, ProductRelated, ProductRelated_Duration, BounceRates,

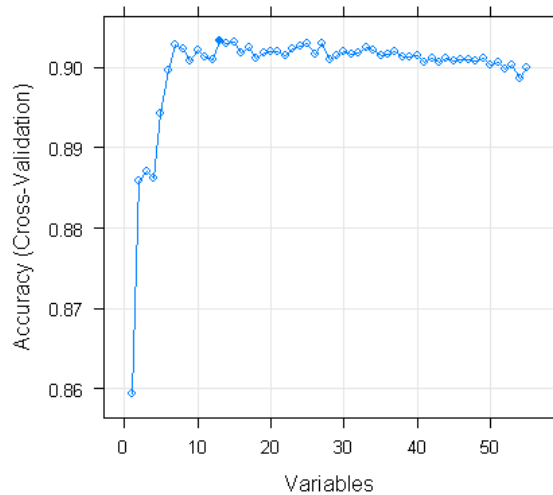


Fig. 5: Cantidad de variables seleccionadas

ExitRates, PageValues, Month_Nov, TrafficType, VisitorType_New_Visitor, VisitorType_Returning_Visitor y la característica de clasificación Revenue.

VI. MODELOS PROPUESTOS

Para llegar a nuestro objetivo (predecir si un cliente va a comprar o no), construimos dos modelos, el primero usando **Árbol de Decisión** y el segundo usando **Regresión Multinomial**.

A. Regresión Multinomial

Antes de explicar el modelo de regresión multinomial, es necesario entender qué es la regresión lineal.

En pocas palabras, la regresión lineal consiste en generar un modelo de regresión (ecuación de una recta) que permita explicar la relación lineal que existe entre dos variables (en el caso de la regresión lineal simple). A la variable dependiente o respuesta se le conoce como Y y a la variable predictora o independiente como X . El modelo se describe como sigue:

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

Uno de los parámetros más importantes es el error aleatorio ϵ , que representa la diferencia entre el valor ajustado por la recta y el valor real. Recoge el efecto de todas aquellas variables que influyen en Y pero que no se incluyen en el modelo como predictores.

Condiciones para la regresión lineal

- 1) **Linealidad:** la relación entre ambas variables debe ser lineal.
- 2) **Distribución normal de los residuos:** los residuos se tienen que distribuir de forma normal, con media igual a 0. Los valores extremos suelen ser una causa frecuente por la que se viola la condición de normalidad.
- 3) **Varianza de residuos constante (homocedasticidad):** la varianza de los residuos ha de ser aproximadamente constante a lo largo del eje X .
- 4) **Valores atípicos y de alta influencia:** estos valores pueden generar una falsa correlación u ocultar una existente.
- 5) **Independencia:** las observaciones deben ser independientes unas de otras.

Es importante notar que las condiciones se verifican a partir de los residuos. Es por ello que primero generamos el modelo y después lo validamos mediante un proceso iterativo en el que se ajusta el modelo inicial, se evalúa mediante los residuos y se mejora.

1) **Modelo:** Para el modelo multinomial, utilizamos las 13 variables que en las secciones anteriores fueron calificadas como las mejores para realizar predicciones. Es importante realizar el entrenamiento del modelo con una parte del mismo y no utilizar todos los datos del dataset ya que de esta manera lograremos mejores predicciones.

La variable de respuesta es *Revenue* que representa si un usuario compra o no en una página web, y el resto de las 12 variables son las predictoras en este modelo.

Para los datos de entrenamiento usamos el 70% de los datos del dataset original y el 30% restante para las predicciones. Los comandos *R* necesarios para estos pasos se describen a continuación:

```
#establecer una semilla
set.seed(1649)

#definir el conjunto de datos a entrenar
trainData=sample_frac(shopMejor,0.7)

#definir el set de datos para predecir
testData=as.data.frame(setdiff(shopMejor, trainData))

#realizar el modelo de regresión multinomial m1
m1= multinom(Revenue ~ ., data=trainData, trace=FALSE)

#obtener un resumen del modelo
summary(m1, cor=FALSE, Wald=TRUE)
```

El resumen nos arroja el valor de cada variable predictora y el error estándar.

Luego procedemos a probar el modelo con los datos que apartamos para este momento.

```
#creamos la tabla que nos permite ver las predicciones (matriz de confusión)
tc<-table(predict(m1,testData),testData$Revenue)
tc
#calculamos el porcentaje de aciertos de nuestro modelo

sum(diag(tc))/nrow(testData)
#porcentaje de aciertos: 0.8487106
```

Luego realizamos un segundo modelo para ver si se puede mejorar la precisión del modelo anterior. Para ello utilizamos la función *step* que selecciona el modelo a partir del criterio de información de Akaike (**AIC**). Creamos un estadístico que permite decidir el orden de un modelo. **AIC** toma en consideración tanto la medida en que el modelo se ajusta a las series observadas como el número de parámetros utilizados en el ajuste.

Buscamos el modelo que describa adecuadamente las series y tenga el mínimo **AIC**.

El modelo obtenido tras aplicar este proceso es el siguiente: *Revenue* *Administrative* + *ProductRelated* + *ProductRelatedDuration* + *BounceRates* + *PageValues* + *MonthNov* + *VisitorTypeReturningVisitor*

Luego probamos el modelo con los datos, y vemos que obtenemos un acierto del 0.8424069%. En este caso no hay mucha variación con respecto al primer modelo en cuanto a la eficiencia en la predicción.

Las matrices de confusión de cada modelo son:

Tabla III: Matriz de Confusión - Modelo m2

testPred	FALSE	TRUE
FALSE	2816	463
TRUE	87	124

B. Árbol de Decisión

Un árbol de decisión es un modelo de predicción utilizado en diversos ámbitos que van desde la inteligencia artificial hasta la Economía. Es un método analítico que a través de una representación esquemática de alternativas facilita la toma de mejores decisiones.

Tabla IV: Matriz de Confusión - Modelo m1

testPred	FALSE	TRUE
FALSE	2797	422
TRUE	106	165

1) **Modelo:** Como es costumbre en estos tipos de modelos empezamos separando parte de los datos para el entrenamiento del modelo y el resto para el testeo, esto nos permitirá saber el nivel de precisión del modelo y además, comparar entre distintos modelos que utilizan mas o menos variables a la hora de predecir si un cliente realizará una compra. Se recomienda usar el 70% de los datos originales para el entrenamiento y el resto para la prueba.

Luego creamos el modelo con **ctree** dándole como entrada los datos de entrenamiento y la variable que queremos predecir a partir del resto, en nuestro caso **"Revenue"**.

Una vez creado el modelo lo usamos para predecir los resultados del conjunto de testeo. Usamos estos resultados para crear la matriz de confusión y saber cual es el grado de precisión de nuestro modelo. Obtenemos mas características del modelo usando el comando **confusionMatrix**, los resultados nos dejan ver que nuestro modelo tiene una exactitud del 90%, lo que quiere decir que de 10 clientes que entren al sitio, podremos predecir con exactitud si 9 de ellos compraran o no.

```
library(party)

set.seed(8356)
indice=sample(2,nrow(shopMejor), replace=TRUE, prob=c(0.7,0.3))
trainData=shopMejor[indice==1,]
testData=shopMejor[indice==2,]

shopTree=ctree(formula=Revenue~., data=trainData)

table(predict(shopTree),trainData$Revenue)

plot(shopTree,type="simple")

testPred=predict(shopTree,newdata=testData)

table(testPred, testData$Revenue)

confusionMatrix(testPred,testData$Revenue)
```

Tabla V: Matriz de Confusión

testPred	FALSE	TRUE
FALSE	3015	182
TRUE	165	368

2) Matriz de confusión y estadísticos:

Accuracy : 0.907
 95% CI : (0.8972, 0.9161)
 No Information Rate : 0.8525
 P-Value [Acc > NIR] : <2e-16
 Kappa : 0.6252
 Mcnemar's Test P-Value : 0.3904
 Sensitivity : 0.9481
 Specificity : 0.6691

Pos Pred Value : 0.9431
 Neg Pred Value : 0.6904
 Prevalence : 0.8525
 Detection Rate : 0.8083
 Detection Prevalence : 0.8571
 Balanced Accuracy : 0.8086
 'Positive' Class : FALSE

3) **Representación gráfica:** El siguiente gráfico muestra las reglas lógicas que se siguen para realizar la predicción.

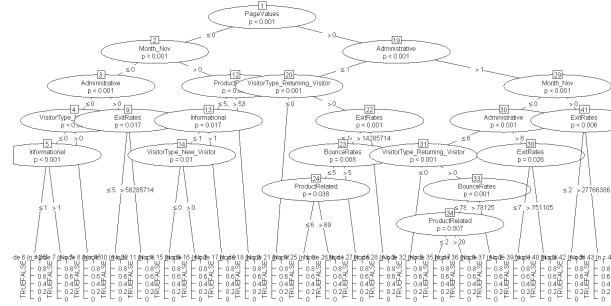


Fig. 6: Árbol de decisión

```
Model formula:
Revenue ~ Administrative + Administrative_Duration + Informational +
Informational_Duration + ProductRelated + ProductRelated_Duration +
BounceRates + ExitRates + PageValues + Month_Nov + TrafficType +
VisitorType_New_Visitor + VisitorType_Returning_Visitor

Fitted party:
[1] root
[2] PageValues <= 0
[3] Month_Nov <= 0
[4] Administrative <= 0
[5] VisitorType_New_Visitor <= 0
[6] Informational <= 1: FALSE (n = 2535, err = 0.2%)
[7] Informational > 1: FALSE (n = 74, err = 2.7%)
[8] VisitorType_New_Visitor > 0: FALSE (n = 152, err = 2.6%)
[9] Administrative > 0
[10] ExitRates <= 58285714: FALSE (n = 2239, err = 3.1%)
[11] ExitRates > 58285714: FALSE (n = 123, err = 9.8%)
[12] Month_Nov > 0
[13] ProductRelated <= 53
[14] Informational <= 1
[15] VisitorType_New_Visitor <= 0: FALSE (n = 943, err = 6.9%)
[16] VisitorType_New_Visitor > 0: FALSE (n = 210, err = 13.8%)
[17] Informational > 1: FALSE (n = 96, err = 18.8%)
[18] ProductRelated > 53: FALSE (n = 314, err = 26.8%)
[19] PageValues > 0
[20] Administrative <= 1
[21] VisitorType_Returning_Visitor <= 0: TRUE (n = 153, err = 2.0%)
[22] VisitorType_Returning_Visitor > 0
[23] ExitRates <= 14285714
[24] BounceRates <= 5
[25] ProductRelated <= 69: TRUE (n = 140, err = 12.1%)
[26] ProductRelated > 69: FALSE (n = 8, err = 37.5%)
[27] BounceRates > 5: TRUE (n = 58, err = 37.9%)
[28] ExitRates > 14285714: TRUE (n = 248, err = 45.2%)
[29] Administrative > 1
[30] Month_Nov <= 0
[31] Administrative <= 6
[32] VisitorType_Returning_Visitor <= 0: TRUE (n = 62, err = 19.4%)
[33] VisitorType_Returning_Visitor > 0
[34] BounceRates <= 78125
[35] ProductRelated <= 20: TRUE (n = 78, err = 16.7%)
[36] ProductRelated > 20: TRUE (n = 141, err = 49.6%)
[37] BounceRates > 78125: FALSE (n = 277, err = 32.5%)
[38] Administrative > 6
[39] ExitRates <= 751105: TRUE (n = 19, err = 31.6%)
[40] ExitRates > 751105: FALSE (n = 323, err = 24.1%)
[41] Month_Nov > 0
[42] ExitRates <= 27766386: TRUE (n = 359, err = 33.1%)
[43] ExitRates > 27766386: FALSE (n = 48, err = 31.2%)

Number of inner nodes: 21
Number of terminal nodes: 22
```

Fig. 7: Condiciones del Árbol de decisión

VII. CONCLUSIONES Y TRABAJO FUTURO

Como vimos a lo largo de las diferentes secciones, la influencia de los outliers en el dataset es un tema muy importante. En nuestro caso teníamos muchos datos atípicos pero decidimos no eliminarlos debido a que en este contexto influían mucho, ya que muchos de los individuos observados que son atípicos terminaron comprando a través de la página web visitada.

Realizamos pruebas pertinentes sobre eliminarlos o reemplazarlos según corresponda el valor del rango intercuartílico pero no ayudaba mucho al modelo de regresión.

En el futuro se realizarán las gestiones para trabajar con datasets sobre ventas online de orden nacional y usar otras técnicas como Clustering para caracterizar a posibles clientes.

REFERENCES

- [1] <https://medium.com/analytics-vidhya/ospi-mul-randomforests-156acdb73fd9>
- [2] <https://www.kaggle.com/roshansharma/online-shopper-s-intention>
- [3] A. Schaerf, Scheduling sport tournaments using constraint logic programming, *Constraints* 4 (1), 1999, pp. 43–65.
- [4] <https://machinelearningmastery.com/feature-selection-with-the-caret-r-package/>
- [5] <https://www.rdocumentation.org/packages/caret/versions/6.0-84/topics/rfe>
- [6] https://rpubs.com/Joaquin_AR/226291